مروری بر آزمون نفوذ وب

مهین السادات میرجلیلی[1]، میترا علیدوستی[2] و علیرضا نوروزی[3]

[1] کارشناسی ارشد، گروه امنیت اطلاعات، دانشگاه مالک اشتر، تهران
Mahin.Mirjalili@gmail.com

[2] دانشجوی دکتری، گروه امنیت اطلاعات، دانشگاه مالک اشتر، تهران
Mitra.Alidoosti@gmail.com

[3] استادیار، گروه امنیت اطلاعات، دانشگاه صنعی مالک اشتر، تهران
Nowroozi@mut.ac.ir

چکیده

در این مقاله مروری بر آزمون نفوذ و به‌طور خاص در زمینهٔ وب خواهیم داشت. در جهت این امر، ابتدا مقالاتی که به‌طورکلی به آزمون نفوذ و روش‌های انجام آن اشاره کرده‌اند پرداخته شده و سپس مقالات موجود در زمینهٔ آزمون نفوذ وب از سه جهت مقایسهٔ ابزارهای انجام‌شده آزمون نفوذ به‌صورت خودکار، معرفی روش یا ابزار جدیدی برای انجام آزمون نفوذ به‌صورت دستی و مقالاتی که محیط آزمونی را جهت آموزش و یا امکان بررسی ابزارها و روش‌های مختلف ارائه کرده‌اند، تقسیم‌بندی کرد. در این مقاله چهار روش مختلف برای انجام آزمون نفوذ وب، سیزده مقاله در زمینهٔ مقایسهٔ پویش‌گرهای آسیب‌پذیری وب، ده مقاله که روش یا ابزار جدیدی برای انجام آزمون نفوذ ارائه داده‌اند و چهار محیط آزمون، مورد بررسی قرارگرفته‌اند.

واژگان کلیدی: آزمون نفوذ، آسیب‌پذیری‌های تحت وب، پویش‌گر

مقدّمه

آزمون نفوذپذیری فرآیند ارزیابی امنیتی شبکه یا سامانه‌های رایانه‌ای است که به‌صورت شبیه‌سازی یک حمله توسط یک نفوذگر اخلاقی صورت می‌پذیرد. مهم‌ترین تفاوت بین نفوذگر و انجام‌دهندهٔ آزمون نفوذ در این است که آزمون نفوذپذیری با مجوز و قراردادی که با سازمان یا شرکت امضا شده است انجام و درنهایت خروجی به‌صورت یک گزارش تهیه می‌شود. هدف از آزمون نفوذپذیری، افزایش ضریب امنیتی داده‌ها است. اطلاعات و ضعف‌های امنیتی که در آزمون نفوذپذیری مشخص می‌شود، محرمانه تلقی شده و نباید تا برطرف شدن کامل افشا شود.

در این مقاله، با توجه به اهمیت امنیت در برنامه‌های کاربردی تحت وب، به‌مرور پژوهش‌های صورت‌گرفته در زمینهٔ آزمون نفوذ و به‌طور خاص آزمون نفوذ وب می‌پردازیم.

سؤالاتی که ذهن پژوهش‌گران را در زمینهٔ آزمون نفوذ درگیر می‌کند، می‌توان به‌صورت زیر بیان کرد:

- آزمون نفوذ چگونه انجام می‌شود؟
- انواع آزمون نفوذ چیست؟
- آزمون نفوذ به‌صورت خودکار چگونه صورت می‌گیرد؟
- از چه ابزارهایی می‌توان برای انجام آزمون نفوذ به‌صورت خودکار استفاده نمود؟
- مقایسه ابزارها و کارآمدی هر یک از آنها؟
- روش‌ها و ابزارهای جدید چه هستند و چه قابلیت‌هایی دارند؟
- چگونه می‌توان ابزارها و روش‌های مختلف را مورد بررسی قرار داد؟

در این مقاله سعی شده است با بررسی چهار روش‌شناسی مختلف برای انجام آزمون نفوذ، سیزده مقاله در زمینهٔ مقایسهٔ پویش‌گرهای آسیب‌پذیری وب، ده مقاله که روش یا ابزار جدیدی برای انجام آزمون نفوذ ارائه داده‌اند و نگاهی به چهار محیط آزمون، به این سؤالات پاسخ داده شود.

با توجه به حجم زیاد مقالات در زمینهٔ مورد بررسی، در انتخاب مقالات سعی شده است معیارهایی از قبیل پوشش گسترهٔ وسیع زمانی از سال ۲۰۰۶ تا ۲۰۱۴ و تعداد ارجاعات

به هر مقاله در نظر گرفته شود. لازم به ذکر است، بیشتر مقالات انتخاباتی، ازجمله مقالاتی هستند که در سال‌های گذشته مورد توجه بسیاری از نویسندگان قرارگرفته‌اند.

در این مقاله، ابتدا به مرور مقالاتی خواهیم پرداخت که به ارائهٔ روشی برای انجام آزمون نفوذ پرداخته‌اند؛ سپس مقالات موجود در زمینهٔ آزمون نفوذ وب را از سه نظر بررسی می‌کنیم:

مقالات موجود در زمینهٔ بررسی و مقایسهٔ روش‌ها و ابزارهای موجود برای انجام آزمون نفوذ وب؛

مقالاتی که روش و یا ابزار جدیدی را برای انجام آزمون نفوذ وب ارائه کرده‌اند؛

پژوهش‌هایی که محیط آزمونی را برای آزمایش ابزارها و نظریه‌های مختلف ارائه کرده‌اند.

۱-آزمون نفوذ

۱-۱- تاریخچه و اهمیت

یکی از قدیمی‌ترین روش‌ها برای دست‌یابی به امنیت در سامانه‌های رایانه‌ای استفاده از آزمون نفوذ است. در اوایل سال ۱۹۷۰ وزارت دفاع آمریکا از این روش برای تشخیص ضعف سامانه‌های رایانه‌ای و تولید برنامه‌هایی با سطح امنیت بالاتر استفاده کرد. با استفادهٔ سازمان‌ها از آزمون نفوذ برای اطمینان از امنیت خدمات و سامانه‌های اطلاعاتی می‌توان ضعف‌های امنیتی را پیش از مورد بهره‌برداری قرارگرفتن شناسایی کرد[۱].

شرکت‌های بزرگ، داده‌های مهمی دارند که حفاظت از آنها یکی از دغدغه‌های همیشگی بوده و هست؛ آزمون نفوذ با شبیه‌سازی حمله‌های متعدد، سازوکارهای امنیتی شرکت‌ها را مورد آزمون قرار می‌دهد. در مواردی از قبیل اضافه‌شدن زیرساخت جدید، نصب نرم‌افزار جدید، به‌روزرسانی سامانه‌ها، اضافه‌شدن وصله‌های امنیتی، تغییر سیاست‌های کاربران نیاز است که آزمون نفوذ اجرا شود. موارد متعددی ازجمله مسائل امنیتی، حفاظت اطلاعات، اولویت‌بندی خطرات امنیتی، جلوگیری از ضررهای مالی و غیره موجب اهمیت انجام آزمون نفوذ می‌شود[۲].

آزمون نفوذ را می‌توان به‌صورت دستی یا خودکار انجام داد. انجام آزمون نفوذ به‌صورت دستی نیازمند یک گروه آزمایش‌کنندهٔ ماهر و با تجربهٔ بالا جهت کنترل تمام کارها است که در تمام مدت آزمون باید حضور فیزیکی داشته باشند، این امر باعث می‌شود این گزینه مقرون‌به‌صرفه نباشد. انجام آزمون نفوذ به‌صورت خودکار راهی ساده و امن برای انجام تمام کارهای مرتبط با آزمون نفوذ است. همچنین به‌دلیل اینکه اکثر کارها به‌صورت خودکار انجام می‌شود، از لحاظ زمانی مقرون‌به‌صرفه‌تر است. یکی دیگر از مزایای این آزمون، امکان استفادهٔ مجدد از پارامترهای تنظیم‌شده برای آزمون است. در [۳] مقایسه‌ای بین این دو آزمون صورت گرفته است که در جدول ۱ قابل مشاهده است.

(جدول ۱): مقایسه آزمون نفوذ دستی و خودکار

	دستی	خودکار
فرآیند انجام آزمون	کار فشرده، ناهماهنگ و مستعد خطا، بدون استانداردهای کیفیت خاص، نیاز به ابزارهای متمایز، هزینه بالا و نیاز به کارکنان امنیتی باتجربه برای اجرا و تفسیر آزمون	سریع، راحت، امن، متمرکز و استانداردشده برای تولید نتایج سازگار و قابل تکرار، گزارش شفاف و قابل‌استفاده
تغییر شبکه	معمولاً روی تعدادی از سیستم‌ها تأثیر می‌گذارد	سیستم‌ها بدون تغییر باقی می‌مانند.
مدیریت و توسعه exploit	توسعه و مدیریت یک پایگاه داده‌ی exploit هزینه‌بر است و نیاز به تجربه بالا دارد، احتمال ناامن بودن exploit های عمومی وجود دارد.	فروشندگان محصول تمام exploitها را توسعه و مدیریت می‌کنند، Exploitها به‌صورت حرفه‌ای توسعه می‌یابند و برای اجرا امن هستند.
پاک‌سازی (*Cleaning Up*)	آزمون‌کننده باید تمام تغییرات را به حالت قبل برگرداند.	با استفاده از یک کلیک عملیات پاک‌سازی انجام می‌شود.
گزارش	نیاز به تلاش قابل‌توجه، ثبت و تلفیق همه‌ی نتایج به‌صورت دستی است.	گزارش‌ها به‌صورت خودکار تولید می‌شوند.
ثبت (Logging) / حسابرسی (Auditing)	آهسته، طاقت‌فرسا، غالباً فرآیند نادرست	به‌صورت خودکار تمام فعالیت‌ها ثبت می‌شود.
آموزش	نیاز است آزمون گر روش‌های موقت و غیراستاندارد را یاد بگیرد	نصب در کمتر از یک روز انجام می‌شود و کاربران می‌توانند کار با ابزار را یاد بگیرند.

C2 Network Security از دو دیدگاه به بررسی آزمون نفوذ می‌پردازد: مطلع‌بودن افراد و دسترسی به سامانه‌ها.

از لحاظ اطلاع افراد، آزمون نفوذ به دو دستهٔ گروه آبی[1] و گروه قرمز[2] تقسیم‌بندی می‌شود. گروه آبی با دانش و رضایت کارکنان فناوری اطلاعات سازمان صورت می‌گیرد. گروه قرمز شامل اجرای یک آزمون نفوذ بدون اطلاع کارکنان فناوری اطلاعات سازمان می‌شود؛ ولی سطوح بالاتر از این موضوع آگاه هستند. گروه آبی هزینهٔ کمتری دارد و به‌صورت متداول‌تری مورد استفاده قرار می‌گیرد. با استفاده از گروه قرمز علاوه‌بر امنیت، می‌توان واکنش کارکنان فناوری اطلاعات به مخاطرات امنیتی و دانش آنها را نیز مورد بررسی قرار داد[۴].

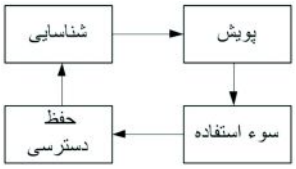
حمله‌هایی که برای آزمون نفوذ صورت می‌گیرد، می‌تواند به‌صورت داخلی یا خارجی باشد. درصورت اجرای هر دو مورد، به‌طورمعمول آزمون خارجی زودتر انجام می‌شود. در آزمون نفوذ خارجی، هیچ اطلاعی از داده‌های درونی سازمان در دسترس نیست و آزمون تنها با اطلاعات جمع‌آوری‌شده انجام می‌گیرد. آزمون داخلی، در درون سازمان و پشت دیوار آتش صورت می‌گیرد و در هنگام اجرای آن امکان دسترسی به اطلاعات سازمان وجود دارد[۴].

۱-۲- روش‌شناسی‌ها و روش‌های موجود

روش‌شناسی، طرحی است که برای رسیدن به مقصد از آن استفاده می‌شود. عدم استفاده از یک روش‌شناسی در آزمون نفوذ ممکن است باعث ناقص‌بودن آزمون، مصرف بالای زمان، نتیجه‌ندادن تلاش و مؤثرنبودن آزمون شود. با وجود تعداد زیاد روش‌شناسی‌ها چیزی به نام " روش‌شناسی درست" وجود ندارد و هر آزمون نفوذ می‌تواند روش‌شناسی متفاوتی داشته باشد؛ ولی استفاده از روش‌شناسی باعث می‌شود آزمون نفوذ حرفه‌ای، مؤثر و با هزینهٔ کمتری صورت گیرد. روش‌شناسی که برای آزمون نفوذ در نظر گرفته می‌شود، به‌طورمعمول دارای ۴ یا ۷ مرحله است. هرچند نام و یا تعداد مرحله‌ها در روش‌شناسی‌های مختلف متفاوت است، همهٔ آنها نمایی کلی از آزمون نفوذ را نشان می‌دهند. برای مثال بعضی از روش‌شناسی‌ها از عبارت "جمع‌آوری اطلاعات[3]" استفاده می‌کنند؛ درحالی‌که برخی همین فرآیند را "شناسایی[4]" می‌نامند.

[1] Blue-teaming
[2] Red-teaming
[3] Information gathering
[4] Reconnaissance

روش‌شناسی مطرح‌شده در منبع[۵] شامل چهار مرحله است: شناسایی، پویش[5] (پویش درگاه، پویش آسیب‌پذیری)، بهره‌برداری[6] و حفظ دسترسی[7]. نخستین مرحله در آزمون نفوذ شناسایی است که در آن به جمع‌آوری اطلاعات درباره هدف پرداخته می‌شود. هرچه اطلاعات بیشتری در این مرحله جمع‌آوری شود، موفقیت در گام‌های بعدی بیشتر خواهد بود. مرحلهٔ دوم در این روش‌شناسی به دو دسته تقسیم می‌شود: پویش درگاه که فهرستی از درگاه‌های باز و خدماتی که روی هرکدام از آن‌ها در حال اجراست به دست می‌آید و پویش آسیب‌پذیری که فرآیندی برای شناسایی ضعف‌هایی است که خدمات و نرم‌افزارهای مورد نظر دارند. با توجه به نتایج به‌دست‌آمده در گام دوم و آگاهی از اینکه چه درگاه‌هایی باز هستند، چه خدماتی بر روی این درگاه‌ها در حال اجراست و این‌که چه آسیب‌پذیری‌هایی دارند می‌توان هدف مورد نظر را مورد حمله قرار داد. آخرین مرحله حفظ دسترسی است. بیش‌تر دسترسی‌هایی که در مرحله حمله به دست می‌آید موقتی است؛ و با قطع‌شدن اتصال از بین می‌رود؛ در این مرحله سعی می‌شود این دسترسی محفوظ بماند. هرچند این منبع "گزارش" را یکی از مراحل آزمون نفوذ در نظر نگرفته است، ولی آخرین فعالیت یک آزمون نفوذ را ارائهٔ گزارش بیان کرده است. بر اساس [۵] گزارش باید شامل جزئیات چگونگی انجام آزمون، خلاصه‌ای از تهدیدات امنیتی پیداشده، مواردی که آزمون پوشش نمی‌دهد و غیره باشد.



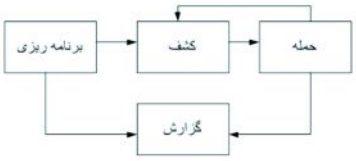
(شکل ۱): روش‌شناسی مطرح در [۵]

در روش‌شناسی آزمون نفوذ NIST آزمون نفوذ شامل چهار مرحله است: برنامه‌ریزی[8]، کشف[9]، حمله[10]، گزارش[11].

در مرحلهٔ برنامه‌ریزی قوانین شناخته می‌شود و اهداف آزمون تنظیم می‌شوند. مرحله کشف در دو مرحله انجام می‌شود، مرحلهٔ نخست شامل شروع آزمون و جمع‌آوری اطلاعات و مرحله‌ی دوم که بعد از مرحلهٔ حمله صورت

[5] Scanning
[6] Exploitation
[7] Maintaining Access
[8] Planning
[9] Discovery
[10] Attack
[11] Reporting

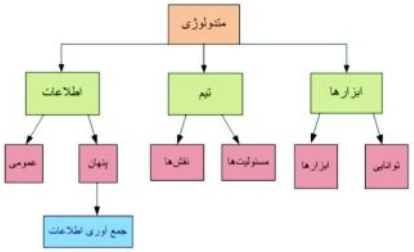
می‌گیرد، شامل تحلیل آسیب‌پذیری‌ها می‌شود. در مرحلهٔ حمله که به‌عنوان قلب آزمون نفوذ شناخته می‌شود، وجود آسیب‌پذیری‌های مختلف در هدف مورد بررسی قرار می‌گیرد. گزارش هم‌زمان با بقیهٔ مرحله‌ها تهیه می‌شود. در مرحلهٔ برنامه‌ریزی برنامه طرح ارزیابی به وجود می‌آید؛ در مرحله‌های کشف و حمله به‌طورمعمول وقایع ثبت و به‌صورت دوره‌ای به مدیر گزارش می‌شود. در انتهای آزمون یک گزارش برای توصیف شناسایی آسیب‌پذیری‌ها، رتبه‌بندی خطرات و راهنمایی برای چگونگی بهبود ضعف‌های شناخته‌شده ارائه می‌شود[۶].



(شکل ۲): روش‌شناسی مطرح در [۶]

روش‌شناسی ارائه‌شده در [۷] شامل سه قسمت اصلی اطلاعات، گروه و ابزارهاست. در بخش اطلاعات، با استفاده از روش‌های مختلف به جمع‌آوری اطلاعات دربارهٔ هدف پرداخته می‌شود. در این مقاله مرحله اطلاعات در چهار گام تعریف شده است: بررسی شبکه، شناسایی سیستم‌عامل، پویش درگاه‌ها، شناسایی خدمات. بخش دوم مطرح‌شده در این روش‌شناسی، تشکیل گروه است. درصورتی‌که گروه‌هایی با نقش‌ها و مسئولیت‌های مختلف شکل بگیرد، آزمون نفوذ به‌صورت مؤثرتری انجام خواهد شد. یکی دیگر از پارامترهای مهم در آزمون نفوذ استفاده از ابزارهاست. برای انجام یک آزمون نفوذ مؤثر بهتر است که به‌جای تسلط کمتر روی تعداد بیشتری ابزار روی تعداد کمتری از ابزارها تسلط بیشتری وجود داشته باشد.

مورد دیگری که در این مقاله مطرح شده تنظیم سیاست‌هایی است که باید توسط آزمون‌کننده و مشتری رعایت شود. در سیاست پیشنهادی مواردی از قبیل محفوظ‌ماندن اطلاعات به‌دست‌آمده توسط آزمون‌کننده و گزارش آنها به‌صورت کامل در انتهای فرآیند آزمون نفوذ، توافق زمان‌بندی، محرمانه‌ماندن تمام اطلاعات ازجمله قرارداد، استفاده از اطلاعات به‌دست‌آمده تنها برای آزمون، عدم مسئولیت آزمون‌کنندهٔ نفوذ درصورت وقوع یک حملهٔ واقعی و غیره بیان شده است.



(شکل ۳): روش‌شناسی مطرح در [۷]

یکی از روش‌شناسی‌های مطرح‌شده Open Source Security Testing Methodology Manual است که توسط ISECOM طراحی شده است. مرحله‌های مطرح‌شده در OSSTMM عبارت‌اند از: مقدمه[۱]، تعامل[۲]، پژوهش[۳]، مداخله[۴].

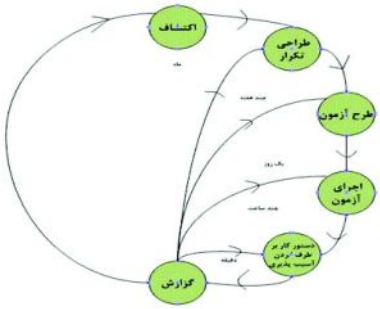
در مرحله مقدمه، محدودهٔ زمانی و نوع آزمون باید مشخص شود. مرحله تعامل مشخص می‌کند که چه هدف‌هایی در حوزهٔ آزمون نفوذ هستند. در مرحله پژوهش بیشترین دادهٔ ممکن دربارهٔ سامانه‌های هدف به دست می‌آید. در مرحله آخر عملکرد امنیتی سنجیده می‌شود. بعد از اتمام آزمون نفوذ، نتایج پردازش‌شده و گزارش تهیه می‌شود. OSSTMM از یک مجموعه ابزار به نام Security Test Audit and Reporting برای پردازش نتایج استفاده می‌کند[۸].

در پژوهش [۹] یک طرح آزمون نفوذ برای برنامه‌های کاربردی تحت وب، مبتنی بر طرح آزمون RUP[۵] ارائه شده است. هرکدام از روش‌شناسی‌های ذکرشده می‌تواند برای اهداف و آزمون‌های نفوذ مختلف مناسب باشد و همان‌طور که گفته شد، نمی‌توان گفت یک روش‌شناسی از دیگری بهتر است. این طرح روشی نظام‌مند، اصولی، مقرون‌به‌صرفه و به‌طورکامل یک‌پارچه‌شده را با چرخهٔ حیات تولید نرم‌افزار مبتنی بر امنیت برای آزمون نفوذ فراهم می‌سازد و دقت، کیفیت و کارایی چنین آزمون‌هایی را بهبود می‌دهد. در این پژوهش همچنین پایگاه داده‌ای از تکنیک‌ها و ابزارهای مورد نیاز برای انجام آزمون نفوذ برنامه‌های کاربردی تحت وب نیز ارائه شده است که در تدوین آن از منابع مختلف ازجمله راهنماها و استانداردهای معتبر و تکنیک‌های آزمون پراکنده در اینترنت بهره گرفته شده است.

در پایان‌نامه [۱۰] یک روش آزمون نفوذ مبتنی بر روش چابک ارائه‌شده که از مزایای روش چابک در فرآیند آزمون

[1] Induction
[2] Interaction
[3] Inquest
[4] Intervention
[5] Rational Unified Process

نفوذ استفاده شده است و توانسته مدلی بر اساس روش‌های اکس.پی و اسکرام را طراحی کرده و جریان‌های اطلاعات بین فعالیت‌ها را نشان دهد, بدین طریق در این روش, چرخهٔ آزمون نفوذ اصلاح شده و می‌تواند چارچوبی جهت بالابردن دقت, کارایی, کیفیت و رضایت شغلی آزمون‌گران شود. در این فرآیند، مدیریت تغییرات به‌راحتی صورت می‌پذیرد و اهداف فناوری اطلاعات با اهداف کسب‌وکار هم‌راستا می‌شود، تعامل با مشتری بالا رفته و باعث می‌شود که تعیین اولویت عمق و دامنه آزمون نفوذ راحت‌تر صورت پذیرد.



(شکل ۴): روش‌شناسی مطرح در [۱۰]

۲- آزمون نفوذ وب

امروزه با گسترش اینترنت و استفاده از برنامه‌های تحت وب در زمینه‌های گوناگون از قبیل نظامی، پزشکی، مالی و غیره، امنیت وب یکی از دغدغه‌های مهم به شمار می‌رود که برای تضمین امنیت از آزمون نفوذ استفاده می‌شود. این آزمون نفوذ می‌تواند به‌صورت دستی و یا خودکار انجام شود که این دو گزینه در جدول ۱ مقایسه شده‌اند.

ابزارهایی را که برای شناسایی آسیب‌پذیری‌ها استفاده می‌شوند، بر اساس اطلاعاتی که از برنامهٔ هدف در اختیار دارند، می‌توان به سه دستهٔ جعبه‌سفید[1]، جعبه‌سیاه[2] و جعبه‌خاکستری[3] تقسیم کرد.

یک ابزار جعبه‌سفید برای بررسی آسیب‌پذیری‌ها از کد برنامهٔ کاربردی هدف استفاده می‌کند. با تجزیه‌وتحلیل کد برنامه، یک ابزار جعبه‌سفید می‌تواند تمام مسیرهای پنهانی برنامه را پیدا کند که باعث پیداشدن آسیب‌پذیری‌های موجود در مسیرهای برنامه می‌شود. در این دسته از ابزارها ممکن است به‌دلیل دسترسی به کد برنامه، آسیب‌پذیری‌هایی گزارش شوند که وجود ندارند؛ یا به‌عبارت‌دیگر احتمال دارد امکان استفاده از آسیب‌پذیری شناخته‌شده وجود نداشته باشد. ایراد آزمون جعبه‌سفید وابسته‌بودن آن به زبان و چارچوبی خاص است.

برخلاف ابزارهای جعبه‌سفید در ابزارهای جعبه‌سیاه فرض می‌شود که هیچ دانشی از کد برنامهٔ کاربردی وجود ندارد و به‌جای استفاده از کد برنامه، آزمون‌گر مانند یک کاربر عادی با یک مرورگر از برنامهٔ کاربردی استفاده می‌کند. یک ابزار جعبه‌سیاه ابتدا قسمت‌های مختلف برنامهٔ کاربردی را برای پیداکردن تمام بردارهای تزریق[4] ممکن بررسی می‌کند. به هر مسیری که به‌وسیلهٔ آن، حمله‌کننده بتواند به‌صورت دستی یا با استفاده از ابزار به یک شبکه یا رایانه نفوذ و از طریق آن محتوای[5] مخرب مورد نظر خود را به سامانه وارد کند، بردار حمله گفته می‌شود. بعد از شناسایی بردارهای تزریق به برنامهٔ کاربردی، ورودی‌هایی برای شناسایی آسیب‌پذیری‌ها داده می‌شود؛ به این فرآیند fuzzing گفته می‌شود و برای fuzzing نوع بردار تزریق و زمان استفاده از آن در ابزارهای جعبه‌سیاه می‌تواند متفاوت باشد. در پایان این ابزارها پاسخ‌های http و html مربوط به fuzzing را مورد بررسی قرار می‌دهند و درصورت موفق‌بودن، آن را به‌عنوان یک آسیب‌پذیری گزارش می‌کنند.

از مزیت‌های ابزارهای جعبه‌سیاه نسبت به جعبه‌سفید می‌توان به وابسته‌نبودن آنها به کد برنامه و مثبت اشتباه[6] کمتر اشاره کرد. با توجه به اینکه یک ابزار جعبه‌سیاه تنها قابلیت شناسایی آسیب‌پذیری‌هایی را دارد که حمله‌ای مربوط به آن را اجرا کند، از عیب‌های آن می‌توان به عدم تضمین شناسایی تمام آسیب‌پذیری‌های برنامه نام برد.

ابزارهای جعبه‌خاکستری همان‌طور که از نام آن‌ها پیداست، ترکیبی از جعبه‌سیاه و جعبه‌سفید هستند. در این ابزارها از تکنیک‌های تجزیه‌وتحلیل ایستای جعبه‌سفید برای شناخت آسیب‌پذیری‌ها استفاده می‌شود؛ سپس به‌صورت واقعی سعی می‌شود روی آسیب‌پذیری‌های پیداشده، حمله اجرا شود تا وجود آنها تصدیق شود. درصورت موفق‌بودن این گام، آسیب‌پذیری گزارش داده می‌شود. ابزارهای جعبه‌خاکستری می‌توانند آسیب‌پذیری‌های موجود در تمام مسیرهای برنامه را با مثبت اشتباه کم پیدا کنند؛ ولی این ابزارها نیز مانند ابزارهای جعبه‌سفید وابسته به زبان یا چارچوب خاص هستند[۱۱].

1 White-Box
2 Black-Box
3 Grey-Box

4 Injection Vector
5 payload
6 False positive

۳- آسیب‌پذیری‌های تحت وب

با توجه به اینکه برنامه‌های کاربردی بسیار آسیب‌پذیرند، مهاجمان، مسیرها و روش‌های مختلفی را برای آسیب به سازمان‌های مختلف به کار می‌گیرند. این آسیب‌پذیری‌ها می‌توانند بسیار ساده یا پیچیده باشند؛ به‌طوری‌که کشف و بهره‌برداری از آنها برای مهاجمان بسیار سخت باشد. هر سازمان با توجه به ویژگی‌های مربوط به محیط و تجارت خود و ریسک‌های مربوط به آنها به‌ویژه عوامل تهدیدکننده، کنترل‌های امنیتی پیاده‌سازی شده و تأثیر آنها بر مسائل مالی و تجاری سازمان شناسایی کند. شرکت OWASP هرساله فهرستی از ده آسیب‌پذیری رایج را منتشر می‌کند. آخرین فهرست منتشرشده در سال ۲۰۱۳ شامل آسیب‌پذیری‌های زیر است[۱۲]:

۱) **تزریق**[1]: خطر تزریق (مانند تزریق OS، SQL و LDAP) زمانی رخ می‌دهد که داده‌های نامطمئن به‌عنوان بخشی از یک دستور به یک مترجم[2] ارسال شود. داده‌های مهاجمان، مترجم را به اجرای دستورهای ناخواسته و یا دسترسی به اطلاعات غیرمجاز وادار می‌کند.

۲) **احراز هویت و مدیریت نشست فروشکسته**[3]: بیشتر بخشی از عملیات برنامه‌های کاربردی که به احراز هویت و مدیریت نشست مرتبط است، صحیح پیاده‌سازی نمی‌شود؛ که باعث حملهٔ مهاجمان به کلمات عبور، کلیدها یا مجوزهای جلسه شوند یا از جریان‌های پیاده‌سازی برای شناسایی هویت کاربران دیگر استفاده کند.

۳) **Cross Site Scripting (XSS)**: این خطر زمانی که برنامهٔ کاربردی داده‌های غیر امنی را به‌کار می‌گیرد و آن را بدون اعتبارسنجی به یک مرورگر ارسال می‌کند، رخ می‌دهد. XSS به مهاجمان اجازه می‌دهد که اسکریپتی را در مرورگر قربانی اجرا کنند که می‌تواند باعث سرقت نشست، مخدوش‌کردن وب‌سایت‌ها و یا تغییر مسیر کاربر به یک وب‌سایت مخرب شود.

۴) **ارجاعات ناامن شئ مستقیم**[4]: این خطر زمانی رخ می‌دهد که برنامه‌نویس به مواردی نظیر پرونده، پوشه یا پایگاه داده بدون کنترل دسترسی یا حفاظت‌های دیگر ارجاع می‌دهد. که در این صورت مهاجمان می‌توانند از مراجع جهت دسترسی به داده‌های غیرمجاز استفاده کنند.

۵) **پیکربندی نادرست امنیتی**[5]: تأمین امنیت، نیازمند تنظیمات دقیق امنیتی برای برنامه‌های کاربردی، چارچوب‌ها، سرویس‌های وب، میزبان‌های پایگاه داده و میزبان‌های برنامه‌های کاربردی تحت وب است. با توجه به اینکه بسیاری از تنظیمات پیش‌فرض به‌اندازهٔ کافی ایمن نیستند، لازم است تا تمامی این تنظیمات به‌درستی تعریف، اجرا و نگهداری شوند.

۶) **در معرض قرار گرفتن داده‌های حساس**[6]: بسیاری از برنامه‌های کاربردی تحت وب به‌درستی از اطلاعات حساس حفاظت می‌کنند. مهاجمان ممکن است این اطلاعات حساس را تغییر دهند یا بدزدند. اطلاعات حساس باید شامل حفاظت‌های اضافی مانند رمزنگاری شوند.

۷) **کنترل سطح دسترسی توابع ازدست‌رفته**[7]: بیشتر برنامه‌های کاربردی تحت وب حقوق سطح دسترسی توابع را قبل از اینکه این عملکرد در UI قابل مشاهده باشد، بررسی می‌کنند. بااین‌حال، برنامه‌های کاربردی نیاز دارند زمانی که هر تابع در دسترس قرار می‌گیرد، بر روی آن کنترل دسترسی را بررسی کنند. درصورتی‌که درخواست‌ها مورد بررسی قرار نگیرند، امکان دارد مهاجمان بتوانند به برخی از قابلیت‌های برنامه، بدون داشتن مجوز مناسب دسترسی پیدا کنند.

۸) **Cross Site Request Forgery (CSRF)**: حمله‌های CSRF مرورگر را وادار به ارسال درخواست جعلی HTTP شامل کوکی نشست قربانی و دیگر اطلاعات هویتی به برنامه‌های کاربردی تحت وب آسیب‌پذیر می‌کند. این خطر به مهاجمان اجازه می‌دهد مرورگر قربانی را مجبور به ارسال درخواست کند و برنامهٔ کاربردی آسیب‌پذیر نیز تصور می‌کند که درخواست‌های ارسالی از طرف قربانی درخواست‌های قانونی و درست است.

۹) **استفاده از اجزا با آسیب‌پذیری‌های شناخته‌شده**[8]: اجزا مانند کتابخانه‌ها، چارچوب‌ها و پیمانه‌های نرم‌افزاری دیگر به‌طورمعمول با دسترسی کامل اجرا می‌شوند. اگر یک جزء

[1] injection
[2] Interpreter
[3] Broken Authentication and Session Management
[4] Insecure Direct Object References
[5] Security Misconfiguration
[6] sensitive data exposure
[7] Missing Function Level Access Control
[8] Using components with known vulnerabilities

آسیب‌پذیر، مورد بهره‌برداری قرار گیرد، می‌تواند موجب ازدست‌دادن داده‌های مهم شود. برنامه‌های کاربردی که از اجزای با آسیب‌پذیری‌های شناخته‌شده استفاده می‌کنند، ممکن است باعث تضعیف دفاع برنامه شود و احتمال برد وسیعی از حمله‌ها را بالا ببرد.

۱۰) **تغییر مسیرها و انتقال‌های نامعتبر**[1]: برنامه‌های کاربردی تحت وب به‌طور مرتب کاربران را به صفحات دیگر هدایت و از داده‌های غیر امن برای تعیین صفحات مقصد استفاده می‌کنند. بدون اعتبارسنجی مناسب، مهاجمان می‌توانند قربانیان را به وب‌سایت‌های مخرب یا صفحات غیرمجاز هدایت کنند.

۴- پویش‌گرهای آسیب‌پذیری جعبه‌سیاه

در سال‌های اخیر پویش خودکار یا نیمه‌خودکار برنامه‌های کاربردی برای پیداکردن آسیب‌پذیری‌ها مورد توجه قرارگرفته است. برای انجام آزمون نفوذ به‌صورت خودکار، پویش‌گرهای متعددی، هم از نوع تجاری و هم از نوع منبع باز وجود دارند. در ادامهٔ این بخش ابتدا به معرفی برخی از پویش‌گرهای تجاری و منبع باز پرداخته و سپس مروری بر پژوهش‌های موجود در حوزهٔ آزمون نفوذ وب خواهیم پرداخت. مقالات موجود در این حوزه سه موضوع را مورد نظر قرار داده‌اند: مقایسهٔ ابزارهای پویش‌گر موجود، ساخت یک پویش‌گر جدید و طراحی یک برنامهٔ کاربردی آسیب‌پذیر.

۴-۱- پویش‌گرهای آسیب‌پذیری منبع باز

در این قسمت تعدادی از پویش‌گرهای منبع باز را با ذکر نام شرکت توسعه‌دهنده، فناوری استفاده‌شده و نقشهٔ مورد استفاده، در جدول ۲ معرفی کرده و سپس در جدول ۳ با بیان مقیاس کاربرد، روش پویش نخستین و خروجی هرکدام به مقایسهٔ آنها می‌پردازیم.

۴-۲- پویش‌گرهای آسیب‌پذیری تجاری

پویش‌گرهای آسیب‌پذیری تجاری توسط سازمان‌های مختلفی توسعه می‌یابند و هرچند هزینهٔ بالایی دارند، ولی نسبت به پویش‌گرهای منبع باز مشکلات امنیتی کمتری دارند. به‌دلیل رقابتی که بین سازمان‌های مختلف برای توسعهٔ پویش‌گرها وجود دارد، محدودیت‌های آنها فاش نمی‌شود.

[1] Invalidated redirects and forwards

در جدول ۴، به معرفی برخی پویش‌گرهای تجاری معروف پرداخته و در (جدول ۵ آنها را مقایسه می‌کنیم.

۴-۳- مطالعات دانشگاهی در زمینهٔ پویش‌گرها

در ادامه، مقالات موجود در زمینهٔ آزمون نفوذ را در سه دسته مورد تحلیل قرار می‌دهیم. دسته‌ای از مقالات به مقایسهٔ پویش‌گرهای تجاری و منبع باز موجود پرداخته‌اند و آنها را در برابر برخی آسیب‌پذیری‌ها مورد آزمون قرار داده‌اند. دستهٔ دوم مقالات بررسی‌شده، مقالاتی هستند که به ارائهٔ روش و یا ابزار جدیدی برای انجام آزمون نفوذ به‌صورت خودکار پرداخته‌اند. همچنین برای ارزیابی روش‌ها و ابزارهای امنیتی نیاز است از برنامه‌های آسیب‌پذیر که آسیب‌پذیری‌های آنها مشخص است، استفاده شود که دستهٔ سوم بررسی‌ها را به این برنامه‌ها اختصاص داده‌ایم.

۴-۳-۱- مقایسهٔ ابزارها و روش‌های موجود

مقالات متعددی به مقایسه و بررسی پویش‌گرهای وب پرداخته‌اند. در برخی از این مقالات مقایسهٔ ابزارها هدف اصلی و در برخی دیگر از مقایسه با هدف ارائهٔ ابزار یا روشی جدید استفاده شده است. بیشترین آسیب‌پذیری‌هایی که در این مقایسه‌ها مورد نظر قرارگرفته است SQL injection و XSS بوده است. در بسیاری از این مقالات آمده است که ابزارهای موجود قابلیت شناسایی همهٔ آسیب‌پذیری‌ها را ندارند و در برخی دیگر نتیجه گرفته شده است که دارای مثبت اشتباه بالا هستند.

در جداول ۶ و ۷ پویش‌گرهای آزمون‌شده در هر مقاله، آسیب‌پذیری‌های مورد نظر و محیط آزمون استفاده‌شده آورده شده است.

در [۱۶] نام ابزارها به‌دلیل تجاری‌بودن و اعلام بی‌طرفی مقاله ذکر نشده است.

در [۱۷] نتایج نشان می‌دهد که ابزارهای متفاوت نتایج به‌طورکامل متفاوتی تولید می‌کنند، قادر به شناسایی بسیاری از آسیب‌پذیری‌ها نیستند و حدود ۲۰٪ تا ۷۰٪ مثبت اشتباه دارند.

(جدول ۲): معرفی پویش‌گرهای منبع باز [۱۳]

نام پویشگر	محصول شرکت	فناوری	پلت فرم
Websecurify	GNUCITIZEN	JavaScript	Windows, Mac, Linux
Skipfish	Micheal Zalewski	C (General)	Linux, FreeBSD, Mac OS X, Windows
Wapiti	Informatica Gesfor	Python (2.6.x)	Unix/Linux, FreeBSD, Mac OS X, Windows
Parosproxy	MileSCAN	Java 1.4x	Linux, Mac OS X, Windows
Arachni	Tasos Laskos	Ruby (1.9.x)	Windows
Open acunetix	John Martinelli	Java 1.6	Windows
Grendel-Scan	David Byrne	Java1.6	Linux, Mac OS X, Windows
W3AF	W3AF developers	Python (2.5.x)	Linux, Mac OS X, Windows
WebScarab	OWASP	Java (1.5.x)	Linux, Mac OS X, Windows
SQLmap	SQLmap developers	Python 2.6	Linux, Mac OS X, Windows
ZAP	OWASP	Java 1.6x	Linux, Mac OS X, Windows
Andiparos	Compass Security	Java 1.5x	Linux, Mac OS X Windows
Watabo	Andreas Schmidt	Ruby 1.8x	Linux, Mac OS X, Backtrack, Windows

(جدول ۳): مقایسه پویش‌گرهای منبع باز [۱۳]

مقیاس کاربرد							متد پویش اولیه		خروجی	
پویشگرها	GUI	یکربندی	کاربرد	پایداری	کارایی	اسپایدر	خزش دستی	تحلیل فایل	گزارش	ثبت وقایع
ZAP	✓	V.S	V.S	V.ST	F	✓	✓	×	✓	✓
Webscarab	✓	V.S	V.S	ST	F	✓	✓	×	✓	✓
Websecurify	✓	V.S	V.S	V.ST	F	✓	×	×	✓	×
Andiparos	✓	V.S	S	ST	F	✓	✓	×	✓	✓
Parosproxy	✓	V.S	V.S	UST	SL	✓	✓	×	✓	✓
GrendelScan	✓	S	S	ST	SL	✓	✓	×	✓	✓
Watabo	✓	V.S	V.S	UST	F	×	✓	×	×	✓
Skipfish	×	S	C	ST	F	✓	×	✓	✓	✓
Open Acunetix	✓	S	S	ST	F	✓	✓	×	×	×
Arachni	✓	S	C	ST	F	✓	✓	×	✓	×
SQLmap	✓	C	S	ST	SL	×	✓	✓	✓	✓
W3AF	✓	C	C	FG	SL	✓	✓	×	✓	✓
Wapiti	×	C	C	FG	F	✓	×	×	✓	✓

منظور از علائم استفاده‌شده در جدول ۳ در زیر آمده است:

V.S:	Very Simple		**UST:**	UnStable
V.ST:	Very Stable		**SL:**	Slow
F:	Fast		**C:**	Complex
ST:	Stable		**FG:**	Fragile
S:	Simple			

(جدول۴): معرفی پویش‌گرهای تجاری [۱۴]

نام پویشگر	ویژگی
Acunetix	نرخ مثبت اشتباه کم ، امکان اصلاح امنیتی برنامه کاربردی،
N-Stalker	مجموعه‌ای از برنامه‌های امنیتی برای ارزیابی امنیت برنامه کاربردی، امکان ایجاد سیاست‌های ارزیابی موردنیاز
Netsparker	تمرکز بر روی کاهش مثبت اشتباه، نخستین پویش‌گر فارغ از مثبت اشتباه
Burp Scanner	پلت فرمی برای اعمال حمله و آزمون امنیتی برنامه‌های کاربردی، قابلیت انجام عملیات هم در مد فعال و هم در مد منفعل، توانایی پویش دستی و همین‌طور خودکار
Rational AppScan	توانایی اصلاح پیشرفته آسیب‌پذیری و همین‌طور نشان دادن نتایج در Results Expert wizard
Falcove	هم عملیات پویش و هم آزمون نفوذ، بررسی امنیتی برنامه‌های کاربردی و بررسی محتویات پویا
HP WebInspect	قابلیت پویش سریع، ارزیابی امنیتی وسیع و نتایج دقیق پویش امنیتی برنامه کاربردی
NTOSpider	شناسایی آسیب‌پذیری برنامه کاربردی، طبقه‌بندی تهدید امنیتی، ارائه گزارش گرافیکی به‌صورت فایلHTML

(جدول ۵): مقایسه پویش‌گرهای تجاری [۱۵]

	مقیاس کاربرد					خروجی	
پویشگرها	GUI	پیکربندی	کاربرد	پایداری	کارایی	گزارش	ثبت وقایع
Acunetix	✓	V.S	V.S	V.ST	F	✓	✓
N-Stalker	✓	V.S	V.S	V.ST	V.F	✓	×
Netsparker	✓	V.S	V.S	V.ST	F	✓	✓
Burp Scanner	✓	V.S	V.S	ST	V.F	✓	✓
Rational AppScan	✓	S	S	V.ST	F	✓	✓
HP WebInspect	✓	V.S	V.S	S	F	✓	✓
NTOSpider	✓	V.S	V.S	S	F	×	✓

مقایسهٔ انجام‌شده توسط McAllister و همکارانش در سال ۲۰۰۸ نشان می‌دهد که قابلیت تشخیص آسیب‌پذیری توسط ابزارها پایین است؛ هرچند با استفاده از تکنیک پیشنهادی آسیب‌پذیری‌های بیشتری پیدا شده است[۱۸].

در مقالهٔ [۱۹] نویسندگان در انتهای مقاله با توجه به نرخ تشخیص آسیب‌پذیری‌ها ادعا کرده‌اند که مجموعهٔ آزمون[۳۲] آنها برای ابزارهای متفاوت مؤثر است. همچنین در این مقاله آمده است که هیچ‌کدام از ابزارها توانایی شناسایی آسیب‌پذیری‌های سطح دو یا بیشتر را ندارند.

در سال ۲۰۱۰ در پایان‌نامه‌ای که توسط Shelly انجام شده، نتیجه گرفته شده که استفاده از یک محیط آزمون با نسخه‌های امن و ناامن روش خوبی برای فهمیدن دلیل تولید مثبت اشتباه و منفی اشتباه توسط ابزارهاست. نتیجه‌ای که روی کیفیت ابزارها اعلام شده بیان‌گر این است که ابزارها قادر به تشخیص تزریق SQL و XSS ساده هستند؛ ولی برای شناسایی SQL injection و XSS غیر ساده، جریان‌های مدیریت نشست، اجرای فایل بدخواه و سرریز بافر کار بیشتری برای بهبود تکنیک‌های ابزارها مورد نیاز است[۱۴].

در [۲۰] اشاره شده است که ابزارها نیاز به فهم بالاتری نسبت به محتوای فعال و زبان‌های برنامه‌نویسی مانند Silverlight، Flash، Java Applet و JavaScript دارند.

در منبع [۲۱] یک ابزار جدید به نام CIVS-WS[۳۳] با روشی جدید برای شناسایی SQL/XPath injection ساخته شده است. در قسمت نتیجه‌گیری آمده است که ابزار پیاده‌سازی‌شده قدرت پوشش ۱۰۰٪ و مثبت اشتباه ۰٪ را دارد.

مقالهٔ [۲۲] بر اساس نتایج به‌دست‌آمده در [۱۶] است. نویسنده روشی را برای شناسایی SQL injection پیشنهاد داده و ابزاری به نام VS.WS را تولید کرده است. برای آزمایش این روش، آزمون [۱۶] دوباره تکرار شده است. تمام ابزارها در برابر ۲۶۲ سرویس وب عمومی و پیاده‌سازی جاوا از چهار سرویس وب مشخص‌شده توسط TPC-APP

[32] Test suite

[33] Command Injection Vulnerability Scanner for Web Services

benchmark اجرا شده است در قسمت نتیجهٔ این مقاله آمده است که عملکرد ابزار پیاده‌سازی‌شده در پوشش و مثبت اشتباه بهتر از ابزارهای تجاری است.

نتیجهٔ مقاله [۲۳] این است که خزش[۳۴] در یک برنامهٔ کاربردی تحت وب پیشرفته یک چالش جدی برای ابزارهای آزمون نفوذ است و نیازمند پشتیبانی آنها از فناوری‌هایی مانند Flash و Java ، الگوریتم‌های پیشرفتهٔ بیشتری برای اجرای خزش عمیق و پی‌گیری حالت برنامهٔ تحت آزمون و پژوهش‌های بیشتری در زمینهٔ خودکارسازی شناسایی آسیب‌پذیری‌های منطق برنامه است.

نویسندگان مقاله [۲۴] به این نتیجه رسیده‌اند که حتی زمانی که به پویش‌گرها یاد داده می‌شود که از آسیب‌پذیری‌ها بهره‌برداری کنند، قادر به تشخیص stored SQL injection نیستند. همچنین در این مقاله آمده است که بهبود بعضی از عملکردهای پویش‌گرهای جعبه‌سیاه، مانند پویش غیر حالتمند، انتخاب ورودی بر اساس نام و برچسب فیلد، تازگی بردار حمله، تجزیه‌وتحلیلی پاسخ میزبان و post scanning می‌تواند باعث بهبود نرخ کشف شود.

منبع [۲۵] که تعمیمی بر [۲۰] و [۲۳] است، با بررسی سه پویش‌گر به این نتیجه رسیده است که پویش‌گرها به‌دلیل وجود ضعف در مرحلهٔ سوم قادر به تشخیص آسیب‌پذیری نیستند.

در منبع [۲۶] نتیجه گرفته شده است که Iron WASP، NetSparker community edition، OWASP ZAP، Vega، N-Stalker و W3AF به‌ترتیب بیشترین تا کمترین تعداد آسیب‌پذیری را شناسایی کرده و از کمترین تا بیشترین منفی اشتباه آنها را دارند.

بر اساس نتایج [۲۷] W3AF، archani و skipfish به‌عنوان بهترین‌ها انتخاب شده‌اند. همچنین نشان داده شده است که بیشترین تفاوت پویش‌گرها در شناسایی آسیب‌پذیری‌های تزریق، cross-site scripting، مدیریت نشست وbroken authentication است. در این بخش مقالات متعددی را بررسی کردیم که به مقایسهٔ پویش‌گرهای مختلف پرداخته بودند. همان‌طور که در جدول ۷ مشاهده می‌شود، آسیب‌پذیری‌هایی که در OWASP top 10 مطرح شده‌اند در این مقالات مورد توجه قرارگرفته‌اند و همچنین برای بالابردن ارزش ارزیابی‌ها در اکثر مقالات از پویش‌گرهای معروفی همچون Acunetix Web Vulnerability Scanner، IBM Rational AppScan، Burp scanner استفاده شده است. در جدول ۸ خلاصه‌ای از تعداد استفاده از هر پویش‌گر در مقالات آمده است.

در جدول ۹ فهرستی از حمله به آسیب‌پذیری‌های مختلف آورده شده و پویش‌گرهایی که بیشتر از بقیه، در مقالات مورد مقایسه قرارگرفته‌اند، بررسی شده و بیان شده است که قادر به شناسایی چند درصد از آسیب‌پذیری‌ها خواهند بود[۲۸].

۴-۳-۲- ارائهٔ روش یا ابزاری جدید

در این قسمت تعدادی از مقالات را که به طراحی پویش‌گری جدید پرداخته‌اند، بررسی می‌کنیم.

در منبع [۲۹] ابزاری با قابلیت تحلیل ایستا به نام Pixy برای شناسایی آسیب‌پذیری‌های برنامه‌های تحت وب طراحی شده است. Pixy یک ابزار منبع باز است و هدف اصلی آن شناسایی آسیب‌پذیری cross-site scripting در اسکریپت‌های PHP است. دلیل انتخاب PHP پرکاربردبودن آن در طراحی برنامه‌های تحت وب و استفادهٔ تعداد زیادی از مشاوران امنیتی از برنامه‌های PHP بوده است. مبنای تشخیص آسیب‌پذیری‌ها در این ابزار، تجزیه‌وتحلیل جریان داده است. برای ارزیابی این ابزار از شش برنامهٔ PHP منبع باز استفاده شده است. در PhpNuke، PhpMyAdmin و Gallery ۳۶ آسیب‌پذیری شناخته‌شده با ۲۷ مثبت اشتباه بازسازی شد. در Simple PHP Blog، Serendipity و Yapig ۱۵ آسیب‌پذیری ناشناخته با ۱۶ مثبت اشتباه شناسایی شدند. از محدودیت‌های pixy می‌توان به عدم پشتیبانی از خصوصیت شئ‌گرایی نام برد.

Saner نام ابزاری است که با ترکیب تکنیک‌های ایستا و پویا برای شناسایی اشتباهات موجود در برنامه‌های کاربردی به زبان PHP در [۳۰] ارائه شده است. در این ابزار تکنیک‌های ایستا برای شناسایی ورودی‌های هر مسیر به‌کمک شیوه‌های مدل‌کردن رشته‌ها مورد استفاده قرارگرفته است. تکنیک دوم تحلیل پویا است که به‌صورت پایین-بالا عمل می‌کند و با استفاده از آن ابتدا کد برای شناسایی ورودی‌ها بازسازی و سپس با استفاده از مجموعهٔ عظیمی از ورودی‌های بدخواه برای شناسایی جریان‌های قابل بهره‌برداری اجرا می‌شود. هدف از مرحله پویا آزمون‌کردن تمام مسیرهای برنامه است که در مرحله ایستا مشکوک شناخته شده‌اند. ازآنجاکه مرحلهٔ تحلیل ایستا محتاطانه است، امکان تولید مثبت اشتباه را دارد که باید به‌صورت دستی ارزیابی شود که عملی خسته‌کننده است. هدف از تحلیل پویا

[34] Crawling

خودکارسازی این امر یا حداقل تشخیص خودکار این است که چه ورودی آسیب‌پذیری باید استفاده شود. برای ارزیابی، این ابزار روی پنج برنامهٔ Jetbox، MyEasyMarket، PBLGuestbook، PHP-Fusion و Sendcard که به زبان PHP نوشته شده مورد ارزیابی قرارگرفته است. هرچند کارایی اجرای این ابزار خیلی خوب نیست؛ ولی زمان مورد نیاز برای تحلیل بیشتر برنامه‌ها خوب است.

(جدول ۶): خلاصه‌ی مقالات با رویکرد مقایسه ابزارهای موجود(بخش اول)

منبع	پویشگرهای آزمون شده	برنامه‌های تحت آزمون
[۱۶]	HP WebInspect, IBM Rational AppScan, Acunetix	
[۱۷]	سه ابزار تجاری بدون ذکر نام	Online BooksStore MyReferences,
[۱۸]	Brup Spider, W3AF, Acunetix	Django-forum Django-basic-blog ,
[۱۹]	چهار ابزار تجاری و منبع باز بدون ذکر نام	یک برنامه‌ی شبیه به بانک برخط
[۱۴]	Grendel-Scan, Wapiti, W3AF, Hailstorm, N-stalker, Netsparker, Acunetix, Brup Scanner	برنامهٔ تغییریافتهٔ BuggyBank نسخه امن و ناامن Hokie Exchange
[۲۰]	Acunetix, Cenzic Hailstorm Pro,HP WebInspect, IBM Rational AppScan, Mcafee SECURE, N-Stalker, QualysGuard PCI, Rapid7 NeXpose	، یک محیط آزمون شخصی‌سازی‌شدهWordPress، phpBB، Drupal
[۲۱]	CIVS-WS, AppScan, Acunetix, VS.BB	عملیاتی که توسط ۹ سرویس وب پیاده‌سازی شده‌اند
[۲۲]	VS.WS	آزمون [۱۶] دوباره تکرار شده است
[۲۳]	Acunetix , IBM Rational AppScan, Burp scanner, Grendel-Scan, Hailstorm, Milescan, N-Stalker , NTOSpider, Paros, W3AF , HP WebInspect	WackoPicko
[۲۴]	Acunetix WVS AppScan , QualysGuard Suite	MatchIt WackoPicko, PCI,
[۲۵]	OWASP ZAP, N-Stalker WVS, Acunetix WVS, IBM Rational AppScan	PCI, WackoPicko, SimplifiedTB(در راستای همین مقاله طراحی شده است)
[۲۶]	به ترتیب افزارش تعداد بردارهای ورودی: Iron WASP، W3AF، N-Stalker، NetSparker، Vega و OWASP ZAP	WackoPicko
[۲۷]	IronWASP, ZAP, SQLmap, W3AF, arachni, Skipfish, Watobo, VEGA, Andiparos, ProxyStrike, Wapiti , ParosProxy, GrendelScan, PowerFuzzer, Oedipus, UWSS, Grabber, WebScarab, MiniMySQLat0r, WSTool, Crawlfish, Gamja, iScan, DSSS, Secubat, SQID, SQLiX, Xcobra	

(جدول ۷): خلاصهٔ مقالات با رویکرد مقایسهٔ ابزارهای موجود(بخش دوم)

منبع	آسیب‌پذیری‌های آزمون شده
[۱۶]	تزریق SQL، تزریق XPath، اجرای کد، پارامترهای ممکن بر پایه‌ی سرریز بافر، امکان کشف نام کاربری و کلمه‌ی عبور و امکان کشف مسیر میزبان
[۱۷]	تزریق SQL، XSS
[۱۸]	stored XSS ،reflected XSS
[۱۹]	XSS, SQL injection, blind SQL injection, file inclusion
[۱۴]	تزریق SQL، XSS، جریان‌های مدیریت نشست، اجرای بدخواهانه‌ی فایل و سرریز بافر
[۲۰]	انواع مختلف XSS، انواع مختلف SQL injection، Cross-Channel Scripting، session management flaws، CSRF، کشف اطلاعات، تنظیمات رمز شده و میزبان ، تشخیص بدافزار
[۲۱]	SQL injection, XPath injection
[۲۲]	SQL injection
[۲۳]	انواع XSS، SQL injection،command-line injection، file inclusion ، file exposure
[۲۴]	انواع مختلف SQL injection به‌خصوص stored SQL inaction
[۲۵]	Stored XSS
[۲۶]	reflected XSS behind reflected XSS behind JavaScript, stored XSS, reflected XSS, stored SQLI, reflected SQLI, parameter manipulation, directory file exposure, file inclusion, command line injection, predictable session ID, , flash weak passwords , traversal, logic flow, forceful browsing
[۲۷]	بر اساس OWASP TOP 10

(جدول ۸): فراوانی پویشگرها در مقالات بررسی‌شده

نام پویشگر	تعداد تکرار
Acunetix Web Vulnerability Scanner	۸
IBM Rational AppScan	۶
W3AF, N-stalker	۵
HP WebInspect , Brup Spider, Grendel-Scan, Hailstorm, Wapiti, OWASP ZAP	۳
Netsparker, VS.BB, Paros, Iron WASP, Vega	۲
بقیه موارد	۱

(جدول ۹): آسیب‌پذیری‌های قابل شناسایی توسط پویش‌گرهای پرکاربرد در مقالات مقایسه‌ای

vulnerability	AppScan	Acunetix	W3AF	N-stalker	vulnerability	AppScan	Acunetix	W3AF	N-stalker
Error based SQL injection	✓	✓	✓	✓	Format String Attack	✓	×	✓	×
Blind SQL injection	✓	✓	✓	✓	Code Injection	✓	✓	×	×
Server Side Java Script injection	×	✓	×	×	XML Injection	✓	×	×	×
Reflected Cross Site Scripting	✓	✓	✓	✓	Expression Language Injection	×	✓	✓	×
Persistent Cross Site Scripting	✓	✓	✓	×	Buffer Overflow	✓	×	✓	×
DOM Cross Site Scripting	✓	✓	✓	×	Integer Overflow	✓	×	×	×
JSON Hijacking	✓	×	×	×	Source Code Disclosure	✓		✓	✓
Path Traversal & Local File Inclusion	✓	✓	✓	✓	Old, backup and Unreferenced Files	✓	✓	✓	✓
Remote File Inclusion	✓	✓	✓	✓	Padding Oracle	×	✓	×	×
Command Injection	✓	✓	✓	✓	Forceful Browsing/ Authentication Bypass	✓	✓	✓	×
Unrestricted File Upload	✓	✓	✓	×	Privilege Escalation	✓	×	×	×
Open Redirect	✓	✓	✓	✓	Xml External Entity	✓	✓	×	×
CLRF injection	✓	✓	✓	✓	Weak Session Identifier	✓	×	✓	×
LDAP Injection	✓	✓	✓	×	Session Fixation	✓	✓	×	×
XPath injection	✓	✓	✓	×	Cross Site Request Forgery	✓	✓	✓	×
SMTP/IMAP/EMAIL Injection	✓	✓	✓	×	Application Denial of service	✓	✓	✓	×
Server-Side Includes Injection	✓	×	✓	×	**درصد شناسایی آسیب‌پذیری‌های مختلف در هر پویش‌گر**	**91.9**	**76.7**	**69.6**	**30.3**

در منبع [۳۱] از رایج‌ترین مدل آسیب‌پذیری اعتبارسنجی ورودی به نام Tainted Mode Model جهت شناسایی آسیب‌پذیری‌های داخلی استفاده شده است. این مقاله tainted mode model کلاسیک بهبود داده تا جریان‌های دادهٔ درونی قابل بررسی باشند. همچنین روش جدیدی با استفاده از اطلاعات به‌دست‌آمده از تجزیه‌وتحلیل پویا برای انجام آزمون نفوذ خودکار معرفی کرده است که با توجه به دید وسیع‌تری که از برنامهٔ کاربردی می‌دهد، صحت و دقت بیشتری خواهد داشت و صحت روال‌های اعتبارسنجی ورودی، قابل آزمایش خواهد بود. با استفاده از tainted mode

model بهبودیافتهٔ برنامه‌های کاربردی به‌صورت زیر در این مقاله مدل شده‌اند:

$$W: (Scheme, Req \times State \rightarrow DDG \times Resp \times [query] \times State)$$

Scheme مجموعه‌ای از رویه‌های دادهٔ ارتباطی[35] است که پایگاه دادهٔ برنامه را نشان می‌دهد. Req درخواست http ارسال‌شده به برنامهٔ کاربردی است. State در سمت چپ به حالت برنامهٔ کاربردی اشاره دارد که شامل محتویات محیط برنامه (مانند پایگاه داده، فایل‌های سیستمی،LDAP و...) می‌شود. DDG = (V,E) گراف وابستگی داده‌هاست که نشان‌دهندهٔ مسیر اجرا و جریان داده‌ای است که برای پردازش درخواست گرفته‌شده، توسط برنامه به‌دست‌آمده است. Resp پاسخ برگردانده‌شده از طرف برنامهٔ کاربردی است. [query] مجموعه‌ای از پرس‌وجوهایی از پایگاه داده است که در هنگام پردازش درخواست، توسط برنامهٔ کاربردی تولید می‌شود. State در سمت راست حالتی است که برنامه به آن منتقل می‌شود.

رویکرد پیاده‌سازی‌شده دارای سه قسمت اصلی است: پیمانه تجزیه‌وتحلیل پویا که اثرات اجرای برنامه را جمع‌آوری می‌کند؛ تحلیل‌گر، که DDGها را برای آثار جمع‌آوری‌شده می‌سازد و پیمانهٔ آزمون نفوذ که ورودی‌های عادی یا بدخواه را به برنامهٔ کاربردی وارد می‌کند. برای ارزیابی سه برنامهٔ کاربردی به زبان پایتون به نام‌های Test application، Spyce و Trac مورد استفاده قرارگرفته است. در نتایج ارائه‌شده مقدار مثبت اشتباه صفر است.

در مقالهٔ [۳۲] پویش‌گری برای شناسایی آسیب‌پذیری‌های تزریق ارائه شده است. این سامانه وب‌سایت‌ها را با هدف یافتن آسیب‌پذیری‌های تزریق SQL و XSS به‌صورت خودکار تحلیل می‌کند. سامانه مطرح‌شده در این منبع شامل دو جزء است: spider و scanner. از spider برای پیمایش سایت و پیداکردن نقاط ورودی استفاده می‌شود. پویش‌گر آزمون، تزریق و تحلیل پاسخ را شروع می‌کند و شامل دو قسمت تحلیل‌گر پاسخ و نویسندهٔ قوانین است. این سامانه در VMware work station ACE با دو میزبان یکی برای میزبان دفاع و دیگری برای میزبان وب‌سایت اجرا شده است. سامانه با PHP5 و MySQL طراحی شده و از پیمانهٔ cURL برای اجرای حمله‌ها استفاده می‌کند. برای انجام ارزیابی هفت برنامهٔ کاربردی از NVD[36] انتخاب‌شده است. درنهایت مقایسه‌ای بین پویش‌گر طراحی‌شده و برخی دیگر از پویش‌گرها انجام‌گرفته و نتیجه به این صورت است که این سامانه مؤثر است و تشخیص آسیب‌پذیری بر اساس نقاط ورودی به‌یقین می‌تواند باعث پیداکردن آسیب‌پذیری‌ها شود.

BLOCK یک روش جعبه‌سیاه است که در [۳۳] برای شناسایی حمله‌های نقض حالت[37] بر مبنای ابزار WebScrab طراحی شده است. در این مقاله برنامهٔ کاربردی به‌عنوان یک سامانه حالتمند در نظر گرفته می‌شود و از تعامل بین مشتری و برنامه، یعنی روابط بین درخواست‌های وب، پاسخ‌ها و متغیرهای نشست، مدل رفتاری برنامه به‌دست می‌آید. BLOCK برای شناسایی حمله‌های نقض حالت دو مرحلهٔ کلیدی دارد: در مرحلهٔ آموزش مدل رفتار مورد نظر با مشاهدهٔ توالی درخواست/پاسخ وب و مقادیر متغیر متناظر با نشست در طول اجرای بدون حملهٔ آن به‌دست می‌آید. در مرحلهٔ شناسایی مدل به‌دست‌آمده برای ارزیابی هر درخواست ورودی و پاسخ خروجی وب استفاده می‌شود و هرگونه نقضی شناسایی می‌شود. برای ارزیابی این ابزار از برنامه‌های Scarf، Simplecms، BloggIt، WackoPicko و OsCommerce استفاده شده است. نتایج نشان می‌دهد که این روش در شناسایی حمله‌های نقض حالت مؤثر است و سربار کمی را متحمل می‌شود. ازآنجاکه این روش مستقل از کد برنامه بوده و به‌راحتی می‌توان از آن برای تعداد زیادی از برنامه‌های تحت وب با چارچوب‌های مختلف استفاده کرد، اهمیت ویژه‌ای دارد.

در مقالهٔ [۳۴] از ماشین حالت میلی برای مدل‌کردن برنامه‌های تحت وب برای مدیریت درخواست‌هایی که موجب تغییر حالت برنامه می‌شوند، استفاده شده است. برای تشخیص تغییر حالت در این مدل متفاوت‌بودن پاسخ برگردانده‌شده در صورت یکسان‌بودن درخواست‌های ارسال‌شده، مورد توجه قرارگرفته است. برای پیاده‌سازی این روش از htmlUnit و مرحلهٔ مربوط به ابزار W3AF استفاده شده است. در گراف حالتی که این ابزار تولید می‌کند، گره‌ها نشان‌دهندهٔ حالت‌ها و یال‌ها نشان‌دهندهٔ درخواست‌های ارسالی به سمت برنامه‌ها هستند. در این مقاله با استفاده از مسئلهٔ رنگ‌آمیزی گراف حالت‌های مشابه با هم ترکیب شده‌اند. برای ارزیابی، این محصول به همراه پویش‌گرهای wget، W3AF و skipfish روی برنامه‌های Gallery، دو نسخه از PhpBB، scraf، vanilla forums، WackoPicko و سه نسخه wordPress مورد بررسی قرارگرفته‌اند. در نتایج ارزیابی دیده می‌شود که پویش‌گر طراحی‌شده نه‌تنها قابلیت

[35] Relational data schemes
[36] National Vulnerability Database
[37] State violation

اجـرای کـد بیشـتری از برنامـهٔ وب را دارد، بلکـه قـادر بـه شناسایی آسیب‌پذیری‌های شناخته‌نشده توسط پویش‌گرهای دیگـر نیـز هسـت. همچنـین ایـن ابـزار بـه‌دلیـل اسـتفاده از HtmlUnit بـا وجـود بـه‌کـارگیری فـازر ابـزار W3AF، دارای مثبـت اشـتباه کمتـری نسـبت بـه ایـن ابـزار اسـت. از محدودیت‌های این ابزار می‌توان به عدم پشتیبانی از AJAX و قابل‌استفاده‌نبودن برای برنامه‌هایی که مـی‌تواننـد بـه‌صـورت عمومی قابل‌استفاده باشند بـه‌دلیـل احتمـال تـأثیرگذاشـتن کاربران روی الگوریتم تشخیص تغییر حالت، اشاره کرد.

در [۱۸] یک ابزار جعبـه‌سـیاه خودکـار بـرای شناسـایی آسـیب‌پـذیری‌هـای XSS reflected و XSS stored در برنامه‌های تحت وب ارائه شـده اسـت. ایـن ابـزار از تعـاملات کاربر برای انجام آزمون مؤثرتر استفاده می‌کند. ابتدا تعـاملات کاربر ضبط و سـپس روی ایـن تعـاملات تغییراتـی در جهـت حمله صورت می‌گیرد و درنهایـت ایـن تـراکنش دوبـاره روی سامانه اجرا می‌شود. برای ارزیابی عملکرد این ابزار با Spider، Spider Brup، W3AF و Acunetix روی سـه برنامـه از چارچوب Django از زوایای مختلف مقایسه شده است. نتـایج نشان می‌دهد که روش ارائه‌شده قادر بـه شناسـایی اشـکالات بیشتری نسبت به ابزارهای تجاری و منبع باز نـام‌بـرده شـده است.

در [۳۵] ابزاری به نام MiMoSA بر مبنای تحلیل ایسـتا برای برنامه‌های PHP ارائه شده است. این مقالـه حملـه‌هـای چنـدپیمانـه را بـه دو دسـتهٔ جریـان داده و گـردش کـار و حالـت‌هـای ورودی را بـه دو دسـتهٔ سـمت میزبـان و سـمت مشـتری تقسـیم کـرده اسـت. تجزیـه‌وتحلیلـی کـه بـرای MiMoSA در نظر گرفته شده دارای دو مرحلـه اسـت: intra-module که هر پیمانه از برنامه را به‌تنهایی بررسی می‌کنـد و مرحله inter-module که کل برنامه را در بر می‌گیـرد، هـدف از تحلیل intra-module خلاصـه‌کـردن هـر پیمانـه از برنامـهٔ کاربردی با تعریف پیش‌شرط‌ها، پس‌شـرط‌هـا و sinkهاسـت. همچنین تمام پیوندهایی کـه در هـر پیمانـه وجـود دارد نیـز استخراج می‌شود. این مرحله وابسته بـه زبـان برنامـه‌نویسـی اسـت؛ سـپس از ایـن اطلاعـات در تحلیـل inter-module استفاده می‌شود تا حالت آیندهٔ جریان کـار برنامـه بـه دسـت آید. در مرحله تحلیل inter-module نتـایج بـه‌دسـت‌آمـده از تحلیل intra-module در یک گراف یکتا جمع‌آوری می‌شـوند که این گراف جریان کار آیندهٔ کل برنامه را مدل می‌کند. بعد از آن از یـک تکنیـک بررسـی مـدل بـرای شناسـایی آسیب‌پذیری‌های جریان داده و نقص‌های جریان کاری آینـده استفاده می‌شود. این مرحله مستقل از زبان برنامه‌نویسی است که در ایجاد برنامهٔ کاربردی استفاده شده است. برای ارزیـابی این ابزار از سه برنامـهٔ Aphpkb، BloggIt، MyEasyMarket، Scarf و SimpleCms استفاده شده است کـه بـه زبـان PHP نوشته شده‌اند. نتایج ارزیابی نشـان مـی‌دهـد کـه MiMoSA قادر به شناسایی تمام آسیب‌پذیری‌های شناخته‌شـده بـوده و تعدادی آسیب‌پذیری جدید نیز کشف کرده است. در ارزیـابی ایـن ابـزار تنهـا یـک مثبـت اشـتباه دیـده مـی‌شـود. تعـداد حالت‌هایی که این ابزار برای یـک برنامـه در نظـر مـی‌گیـرد، بیشتر از تعداد حالات واقعی موجود در کد برنامه اسـت. ایـن مشکل به دو دلیل اصـلی اتفـاق مـی‌افتـد: نخسـت اینکـه در MiMoSA ممکن است، حالاتی متناظر با مسـیرهـایی تولیـد شود که در برنامه قابل‌اجرا نباشند و دوم احتمال حالـت‌هـای تکراری در این ابزار با شرط‌های هم‌تراز ولی متفاوت.

در منبـع [۳۶] ابتـدا از تجزیـه‌وتحلیـل پویـا و مشـاهدهٔ عملکرد برنامهٔ کاربردی برای پی‌بردن به خصوصـیات رفتـاری آن استفاده شده است؛ سپس خصوصیات پیدا شدهٔ فیلترشده تا مقدار مثبت اشتباه کاهش یابد و از مـدل بررسـی نمـادین روی ورودی‌ها استفاده‌شده تا مسیرهـای برنامـه کـه موجـب نقص این شرایط می‌شوند، شناسایی شوند. تمرکز ایـن مقالـه بر روی آسیب‌پذیری‌های منطقی اسـت. ابـزار ارائـه‌شـده کـه Waler نام دارد، برای برنامه‌های بر مبنای servlet که در جاوا نوشته شده‌اند، کاربرد دارد. برای ارزیـابی ایـن ابـزار از دوازده برنامه استفاده شده است. به گفتهٔ این مقاله، Waler نخستین ابزاری است که می‌تواند بـه‌صـورت خودکـار و بـدون دخالـت انسان جریان‌های منطقی برنامهٔ کاربردی را تشخیص دهد.

ابزارهای متنوع بسیاری برای اجرای هر یـک از گـام‌هـای آزمون نفوذ وجود دارند. در مقالهٔ [۳۷] برای صـرفه‌جـویی در زمـان تعـدادی از ایـن ابـزارهـا از قبیـل theHarvester، Metagoofile، ZAP، NMAP، Nessus و Metasploit بـا یکدیگر تلفیق شده‌اند. زبان مورد استفاده در این مقاله پایتون است. عملکرد این روش پیشنهادی را می‌توان در سـه مرحلـه توضـیح داد: جمـع‌آوری اطلاعـات، تجزیـه‌وتحلیـل اطلاعـات به‌دست‌آمـده و اسـتفاده از ایـن اطلاعـات بـرای پیـداکـردن آسیب‌پذیری‌های ممکـن. در ابتـدا ابـزارهـای theHarvester، Metagoofile و NMAP بـه‌عنـوان ابـزارهـای جمـع‌آوری اطلاعات و Nessus و ZAP به‌عنوان دو پـویش‌گـر بـه‌ترتیـب اجرا می‌شوند و اطلاعات به‌دست‌آمده از هر ابزار در یک پوشهٔ مجزا ذخیره می‌شود؛ سپس نتـایج بـه‌دسـت‌آمـده از ZAP و

Nessus تجزیه و از پروندهٔ خروجی به‌دست‌آمده برای انجام مرحلهٔ حمله توسط ابزار Metasploit استفاده می‌شود.

در این بخش ۱۰ پژوهشی که روش یا ابزار جدیدی را برای پیدا کردن آسیب‌پذیری‌های تحت وب ارائه کرده بودند، مورد توجه قرار دادیم. هرکدام از پژوهش‌های مطرح‌شده بر پایه تحلیل ایستا یا پویا و یا ترکیبی از آنها هستند که خلاصه‌ای از هرکدام از مقالات بررسی‌شده در جدول ۱۰ آورده شده است.

(جدول ۱۰): آنالیز استفاده‌شده در تحقیقات

منبع	سال	ایستا	پویا	توضیحات
[۲۹]	۲۰۰۶	✓		شناسایی آسیب‌پذیری cross-site scripting
[۳۰]	۲۰۰۸	✓	✓	شناسایی اشتباهات موجود در برنامه‌های کاربردی با شناسایی ورودی‌های هر مسیر
[۳۱]	۲۰۰۸		✓	شناسایی آسیب‌پذیری‌های داخلی با استفاده از TMD
[۳۲]	۲۰۱۰		✓	برای شناسایی آسیب‌پذیری‌های تزریق و XSS
[۳۳]	۲۰۱۱	✓	✓	شناسایی حمله‌های نقض حالت بر مبنای ابزار WebScrab
[۳۴]	۲۰۱۲	✓	✓	مدل کردن برنامه با استفاده از ماشین حالت باهدف شناسایی آسیب‌پذیری‌های بیشتر
[۱۸]	۲۰۰۸		✓	شناسایی آسیب‌پذیری‌های reflected XSS و stored XSS
[۳۵]	۲۰۰۷	✓		تحلیل کد برنامه برای رسیدن به گراف جریان کار
[۳۶]	۲۰۱۰		✓	شناسایی جریان‌های منطقی برنامه‌های کاربردی
[۳۷]	۲۰۱۳	✓		ترکیب ابزارهای مختلف برای صرفه‌جویی در زمان.

۴-۳-۳-طراحی محیط‌های آزمون

برنامه‌های کاربردی آسیب‌پذیر، برای ارزیابی پویش‌گرهای آسیب‌پذیری وب طراحی می‌شوند. در این قسمت تعدادی از این برنامه‌های آسیب‌پذیر که مورد توجه دوره‌های آموزش آزمون نفوذ و همچنین مقالات متعدد هستند، معرفی می‌کنیم. ازجمله برنامه‌های مطرح در این حوزه می‌توان به موارد زیر اشاره کرد:

- Damn Vulnerable Web App(DVWA)[38]
- OWASP WebGoat[39]
- WackoPicko[23]
- BodgeIt[40]
- WAVSEP[41]
- REFApp[42]

DVWA یک برنامهٔ کاربردی وب آسیب‌پذیر است که با استفاده از PHP و MySQL طراحی شده است. هدف اصلی این برنامه کمک به افراد متخصص در زمینهٔ امنیت برای سنجش مهارت و ابزارهای آنها، کمک به برنامه‌نویسان برای داشتن درک بهتری از امنیت برنامهٔ کاربردی تحت وب و کمک به مدرسان و دانش‌آموزان برای آموختن و یادگیری امنیت برنامه‌های کاربردی است. آسیب‌پذیری‌هایی که DVWA پوشش می‌دهد، شامل Brute force login، command execution، CSRF، file inclusion، SQL injection، upload vulnerability و XSS است[۳۸].

برنامهٔ WebGoat توسط شرکت OWASP برای معرفی نقص‌های امنیتی مرسوم در برنامه‌های کاربردی طراحی شده و بر مبنای J2EE است. آسیب‌پذیری‌هایی که این برنامه پوشش می‌دهد، شامل آسیب‌پذیری‌های access control flaws، AJAX security، authentication flaws، سرریز بافر، کیفیت کد، هم‌زمانی، XSS، IMPROPER ERROR HANDLING، جریان‌های تزریق، منع سرویس، ارتباط ناامن، تنظیمات ناامن، ذخیرهٔ ناامن، اجرای بدخواه، parameter tampering ، سرویس‌های وب و عملکردهای ادمین است[۳۹].

WackoPicko یک وب‌سایت آسیب‌پذیر است که توسط Adam Doupé طراحی شده است. برای نخستین بار در [۲۳] مورد استفاده قرار گرفت. آسیب‌پذیری‌های این برنامهٔ کاربردی شامل reflected XSS، stored XSS، Session ID vulnerability، stored SQL injection، reflected SQL injection، Directory Traversal، multi-step stored XSS، forceful browsing، Command-line Injection، File Inclusion، Parameter Manipulation، Reflected XSS

Reflected XSS، Logic Flaw، Behind JavaScript Weak username/password و Behind a Flash Form است.

BodgeIt یک برنامهٔ آسیب‌پذیر تحت وب است که توسط Simon Bennett با هدف کمک به آزمون نفوذکنندگان طراحی شد. آسیب‌پذیری‌های موجود در این برنامه شامل cross site scripting، تزریق SQL، hidden(but unprotected) content، cross site request forgery، debug code، insecure object references و application logic vulnerabilities است[40].

WAVSEP[38] یک برنامهٔ کاربردی آسیب‌پذیر است که برای کمک به ارزیابی ویژگی‌ها، کیفیت و دقت پویش‌گرهای وب طراحی شده است. این برنامهٔ کاربردی شامل ترکیبی از صفحات آسیب‌پذیر است که می‌توان از آن‌ها برای آزمایش خصوصیات مختلف پویش‌گرها استفاده کرد. با استفاده از این برنامهٔ آسیب‌پذیر می‌توان پویش‌گرها را در مواردی از قبیل path traversal، remote file inclusion، reflected XSS، error based SQL injection، blind SQL injection و time based SQL injection ارزیابی نمود[41].

REFApp ترکیبی از ۲۷ مورد آزمون است. مورد ۱ تا ۱۲ دارای آسیب‌پذیری XSS نوع نخست هستند که بلافاصله بعد از واردکردن ورودی حمله، نتیجهٔ آن به کاربر نشان داده خواهد شد. موارد ۱۳ تا ۲۲ دارای آسیب‌پذیری XSS نوع دوم هستند. در این موارد پردازش داده به دو شکل، صورت می‌گیرد: نمایش نتیجهٔ واردشدن داده‌های ورودی بعد از چند گام متوالی و یا نمایش نتیجه، در موردی به‌جز موردی که در آن ورودی وارد شده است. موارد ۲۳ تا ۲۷ برای آزمایش قابلیت پویش‌گرها در زمینه‌هایی خاص مانند واردکردن تنظیمات مربوط به احراز هویت و برخورد با فناوری جاواسکریپت، در نظر گرفته شده‌اند[۴۲].

در جدول ۱۱ خلاصه‌ای از آسیب‌پذیری‌های موجود در برنامه‌های آسیب‌پذیر معرفی‌شده در این بخش، آورده شده است.

۶- نتیجه‌گیری

در این مقاله مروری بر پژوهش‌های انجام‌شده در زمینهٔ آزمون نفوذ و به‌خصوص آزمون نفوذ وب داشتیم. انجام آزمون نفوذ به‌صورت دستی از لحاظ هزینه و زمان مقرون‌به‌صرفه نیست؛ بنابراین انجام این کار به‌صورت خودکار مورد توجه قرار می‌گیرد. برای انجام آزمون نفوذ وب به‌صورت خودکار از پویش‌گرهای وب استفاده می‌شود که در ابتدا هدف را مورد خزش قرار داده و سپس نتایج به‌دست آمده از مرحلهٔ قبل را مورد حمله قرار می‌دهد و درنهایت گزارشی از آسیب‌پذیری‌های موجود در هدف را اعلام می‌کند.

در این مقاله، پژوهش‌های موجود در زمینهٔ آزمون نفوذ وب را در سه دسته بررسی کردیم: مقالاتی که پویش‌گرهای موجود را مورد مقایسه و تحلیل قرار داده و مقالاتی که روش یا ابزاری جدید را برای انجام آزمون نفوذ پیشنهاد داده و مقالاتی که محیط آزمونی را برای آزمایش ابزارهای مختلف ارائه کرده‌اند. با توجه به بررسی مقالاتی که پویش‌گرهای مختلف را مورد تحلیل قرار داده‌اند، پویش‌گرهای Acunetix Web Vulnerability Scanner و IBM Rational AppScan و آسیب‌پذیری‌های SQL Injection و XSS بیشتر از بقیه مورد توجه قرارگرفته‌اند. همچنین مروری بر ده پژوهش که روش یا ابزار جدیدی را برای انجام آزمون نفوذ ارائه داده بودند، داشتیم که برخی از آنها بر پایهٔ تحلیل پویا، برخی بر پایهٔ تحلیل ایستا و برخی بر پایهٔ ترکیبی از تحلیل‌ها بودند. برای اینکه هر روش یا ابزاری که در زمینهٔ آزمون نفوذ وجود دارد، مورد ارزیابی قرار گیرد نیاز به استفاده از محیط‌های آزمون است که در بخش پایانی چهار مورد از این محیط‌ها را معرفی کرده‌ایم.

ازجمله مشکلات موجود در پویش‌گرهای موجود عدم پشتیبانی از حمله‌هایی مانند Stored XSS و Stored SQL است که نیاز به انجام چند مرحله برای تکمیل حمله دارند، عدم پشتیبانی پویش‌گرها از فناوری‌های جدید و آسیب‌پذیری‌های مربوط به جریان‌های منطقی برنامه است که امید است در کارهای آینده مورد توجه پژوهش‌گران قرار گیرد.

[38] Web Application Vulnerability Scanner Evaluation Project

(جدول ۱۱): آسیب‌پذیری‌های موجود در برنامه‌های آسیب‌پذیر

REFApp	WAVSEP	BodgeIt	WackoPicko	WebGoat	DVWA	
			✓		✓	**Brute Force**
✓		✓			✓	**CSRF**
	✓		✓		✓	**File Inclusion**
✓	✓	✓	✓	✓	✓	**SQL Injection**
				✓		**Access Control Flaws**
✓	✓	✓	✓	✓	✓	**XSS**
				✓		**Buffer Overflows**
		✓	✓			**logic vulnerabilities**
				✓		**AJAX Security**

۷- مراجع

[1] .http://searchnetworking.techtarget.com/tutorial/Network-penetration-testing-guide.

[2] Samant, N., Automated penetration testing. 2011, San Jose State University.

[3] http://www.coresecurity.com/comparing-security-testing-options.

[4] http://www.c2networksecurity.com/.

[5] Engebretson, P., The Basics of Hacking and Penetration Testing: Ethical Hacking and Penetration Testing Made Easy. 2013: Elsevier.

[6] Scarfone, K., et al., Technical guide to information security testing and assessment. NIST Special Publication, 2008. **800**: p. 115.

[7] Alisherov, F. and F. Sattarova. Methodology for penetration testing. in International Journal of Grid and Distributed Computing. 2009. Citeseer.

[8] Herzog, P., Open-source security testing methodology manual. Institute for Security and Open Methodologies (ISECOM), 2003.

۹. عارف‌زاده, ع., ارائه یک طرح آزمون نفوذ برای برنامه‌های کاربردی تحت وب، مبتنی بر طرح آزمون RUP. 1390, دانشگاه صنعتی مالک اشتر.

۱۰. مرواری, س., ارائه یک متدولوژی آزمون نفوذ از نظر ساختار بهبود یافته. ۱۳۹۱, دانشگاه صنعتی مالک اشتر.

[11] Doupé, A.L., Advanced Automated Web Application Vulnerability Analysis. 2014, UNIVERSITY OF CALIFORNIA Santa Barbara.

[12] Wichers, D., OWASP Top-10 2013. 2010.

[13] Manju Khari , N.S., An Overview of Black Box Web Vulnerability Scanners. 2014 International Journal of Advanced Research, 2014.

[14] Shelly, D.A., Using a Web Server Test Bed to Analyze the Limitations of Web Application Vulnerability Scanners. 2010, Virginia Polytechnic Institute and State University.

[15] http://www.sectoolmarket.com/general-features-comparison-unified-list.html#Glossary.

[16] Vieira, M., N. Antunes, and H. Madeira. Using web security scanners to detect vulnerabilities in web services. in Dependable Systems & Networks, 2009. DSN'09. IEEE/IFIP International Conference on. 2009. IEEE.

[17] Fonseca, J., M. Vieira, and H. Madeira. Testing and comparing Web vulnerability scanning tools for SQL injection and XSS attacks. in Dependable Computing, 2007. PRDC 2007. 13th Pacific Rim International Symposium on. 2 .۰۰۷IEEE.

[18] McAllister, S., E. Kirda, and C. Kruegel. Leveraging user interactions for in-depth testing of web applications. in Recent Advances in Intrusion Detection. 2008. Springer.

[19] Fong, E., et al. Building a test suite for web application scanners. in Hawaii International Conference on System Sciences, Proceedings of the 41st Annual. 2008. IEEE.

[20] Bau, J., et al. State of the art: Automated black-box web application vulnerability testing. in Security and Privacy (SP), 2010 IEEE Symposium on .۲۰۱۰ .IEEE.

[21] Antunes, N., et al. Effective detection of SQL/XPath injection vulnerabilities in web services. in Services Computing, 2009. SCC'09. IEEE International Conference on. 2009. IEEE.

[22] Antunes, N. and M. Vieira. Detecting SQL injection vulnerabilities in web services. in Dependable Computing, 2009. LADC'09. Fourth Latin-American Symposium on. 2009. IEEE.

[23] Doupé, A., M. Cova, and G. Vigna, Why Johnny can't pentest: An analysis of black-box web vulnerability scanners, in Detection of Intrusions and Malware, and Vulnerability Assessment. 2010, Springer. p. 111-131.

[24] Khoury, N., et al. An analysis of black-box web application security scanners against stored SQL

injection. in Privacy, security, risk and trust (passat), 2011 ieee third international conference on and 2011 ieee third international conference on social computing (socialcom). 2011. IEEE.

[25] Alassmi, S., et al., An Analysis of the Effectiveness of Black-Box Web Application Scanners in Detection of Stored XSSI Vulnerabilities.

[26] Suteva, N., D. Zlatkovski, and A. Mileva, Evaluation and Testing of Several Free/Open Source Web Vulnerability Scanners. 2013.

[27] SAEED, F.A. and E.A. ELGABAR, ASSESSMENT OF OPEN SOURCE WEB APPLICATION SECURITY SCANNERS. Journal of Theoretical and Applied Information Technology, 2014. **61**.(۲)

[28] http://www.sectoolmarket.com/.

[29] Jovanovic, N., C. Kruegel, and E. Kirda. Pixy: A static analysis tool for detecting web application vulnerabilities. in Security and Privacy, 2006 IEEE Symposium on. 2006. IEEE.

[30] Balzarotti, D., et al. Saner: Composing static and dynamic analysis to validate sanitization in web applications. in Security and Privacy, 2008. SP 2008. IEEE Symposium on. 2008. IEEE.

[31] Petukhov, A. and D. Kozlov, Detecting security vulnerabilities in web applications using dynamic analysis with penetration testing. Computing Systems Lab, Department of Computer Science, Moscow State University, 2008.

[32] Chen, J.-M. and C.-L. Wu. An automated vulnerability scanner for injection attack based on injection point. in Computer Symposium (ICS), 2010 International. 2010. IEEE.

[33] Li, X. and Y. Xue. BLOCK: a black-box approach for detection of state violation attacks towards web applications. in Proceedings of the 27th Annual Computer Security Applications Conference. 2011. ACM.

[34] Doupé, A., et al. Enemy of the State: A State-Aware Black-Box Web Vulnerability Scanner. in USENIX Security Symposium. 2012.

[35] Balzarotti, D., et al. Multi-module vulnerability analysis of web-based applications. in Proceedings of the 14th ACM conference on Computer and communications security. 2007. ACM.

[36] Felmetsger, V., et al. Toward automated detection of logic vulnerabilities in web applications. in USENIX Security Symposium. 2010.

[37] Haubris, K.P. and J.J. Pauli. Improving the Efficiency and Effectiveness of Penetration Test Automation. in Information Technology: New Generations (ITNG), 2013 Tenth International Conference on. 2013. IEEE.

[38] http://www.dvwa.co.uk/.

[39] https://www.owasp.org/index.php/Category:OWASP_WebGoat_Project.

[40] https://code.google.com/p/bodgeit./

[41] https://code.google.com/p/wavsep./

[42] Korscheck, C., Automatic Detection of Second-Order Cross-Site Scripting Vulnerabilities. Wilhelm Schickard Institute, University of Tubingen. Diploma Report, 2010.

مهین السادات میرجلیلی، فارغ‌التحصیل کارشناسی ارشد رشتهٔ فناوری اطلاعات و گرایش امنیت اطلاعات از دانشگاه صنعتی مالک اشتر در سال ۹۳، و دارای مدرک کارشناسی مهندسی فناوری اطلاعات از دانشگاه صنعتی اصفهان است. زمینه‌های پژوهشی ایشان آزمون نفوذ، امنیت وب و آسیب‌پذیری‌های موجود در سطح وب است.

علیرضا نوروزی مدرک کارشناسی را در رشتهٔ مهندسی کامپیوتر (نرم‌افزار) از دانشگاه فردوسی مشهد اخذ کرده است، سپس دوره کارشناسی ارشد خود را در رشتهٔ علوم کامپیوتر در دانشگاه صنعتی شریف به اتمام رساند. وی مدرک دکترای خود را در رشتهٔ علوم کامپیوتر از دانشگاه صنعتی امیرکبیر اخذ کرده است. ایشان در حال حاضر، استادیار دانشگاه صنعتی مالک اشتر بوده و در پژوهشکده امنیت عضو هیأت علمی گروه علمی امنیت اطلاعات و ارتباطات است. زمینه‌های پژوهشی مورد علاقهٔ ایشان موضوعات امنیت شبکه، حلات سایبری، ارزیابی امنیتی، فرماندهی و کنترل و مدیریت بحران است.

میترا علی‌دوستی مدرک کارشناسی و کارشناسی ارشد خود را از دانشگاه علم و صنعت ایران در رشتهٔ مهندسی کامپیوتر به‌ترتیب در سال‌های ۸۸ و ۹۱ اخذ کرده است. ایشان هم‌اکنون دانشجو دکترای رشتهٔ مهندسی کامپیوتر در دانشگاه صنعتی مالک اشتر هستند. زمینه‌های پژوهشی ایشان، امنیت شبکه‌های کامپیوتری، امنیت وب و آزمون نفوذ است.