بررسی ۱۰ آسیب پذیری برتر حوزه وب

در سال ۲۰۱۷

فهرست مطالب

۱) تغییرات نسخه ۲۰۱۷ نسبت به ۲۰۱۳

مهمترین تغییراتی که در گزارش OWASP Top 10 سال ۲۰۱۷ نسبت به نسخه قبلی آن در سال ۲۰۱۳ وجود دارد شامل موارد زیر است:

- بند A4 با عنوان Insecure Direct Object References و بند A7 با عنوان Missing Function Level Access Control به صورت یک بند با عنوان A5:2017-Broken Access Control با یکدیگر ترکیب شده اند.
- بند A8 با عنوان Cross-Site Request Forgery (CSRF) از فهرست ۱۰ آسیب پذیری برتر سال ۲۰۱۷ خارج شده و در جایگاه ۱۳ قرار گرفته است.
- بند A10 با عنوان Unvalidated Redirects and Forwards از فهرست ۱۰ آسیب پذیری برتر سال ۲۰۱۷ خارج شده و در جایگاه ۲۵ قرار گرفته است.
- سه آسیب پذیری جدید با عنوان A4:2017-XML External Entities (XXE) و A8:2017-Insecure Deserialization و A10:2017-Insufficient Logging and Monitoring به ترتیب جایگاه های چهارم و هشتم و دهم را در سال ۲۰۱۷ به خود اختصاص داده اند.
- آسیب پذیری XSS از رتبه سوم در سال ۲۰۱۳ به رتبه هفتم در سال ۲۰۱۷ تنزل یافته است.

در ادامه مقایسه ای از تغییرات صورت گرفته در این دو نسخه مشاهده می نمایید:

OWASP Top 10 – 2013		OWASP Top 10 – 2017
A1 – Injection	→	A1:2017-Injection
A2 – Broken Authentication and Session Management	→	A2:2017-Broken Authentication
A3 – Cross-Site Scripting (XSS)	↘	A3:2017-Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	U	A4:2017-XML External Entities (XXE) [NEW]
A5 – Security Misconfiguration	↘	A5:2017-Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017-Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	U	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	☒	A8:2017-Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017-Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	☒	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

۲) مروری کلی بر ۱۰ آسیب پذیری برتر سال ۲۰۱۷

A1 : تزریق

آسیب پذیری نوع تزریق که به صورت تزریق SQL، تزریق NoSQL، تزریق دستورات OS و تزریق LDAP وجود دارد، زمانی رخ می دهد که داده هایی از منابع نامطمئن به صورت یک دستور یا پرس وجوی پایگاه داده به سمت یک مفسر ارسال شوند. درواقع درجاهایی که برنامه نویس انتظار ورود یک داده را دارد هکر به جای یک داده، کد یا دستور قابل اجرا را وارد می کند که با اجرای آن دسترسی غیرمجاز به اطلاعات خواهد داشت.

A2 : احرازهویت شکسته شده

امروزه رویه های کاری داخلی بسیاری از برنامه های کاربردی وابسته به احرازهویت و مدیریت نشست است که متاسفانه اغلب نیز به درستی پیاده سازی نمی شود و به هکرها این امکان را می دهد تا به اطلاعات پسوردها، کلیدها یا توکن های مربوط به نشست های کاربران به صورت غیرمجاز دسترسی پیدا کنند.

A3 : افشای اطلاعات مهم

بسیاری از برنامه های کاربردی وب و API ها مکانیسم مناسبی جهت محافظت از داده های حساس خود ندارند که هکر این فرصت را خواهد داشت تا داده های با محافظت ضعیف را دستکاری کرده یا به سرقت ببرد و یا از حساب های بانکی کاربران سو استفاده نماید. داده های حساس مستلزم محافظت های اضافه تری هستند مثلاً رمزنگاری اطلاعات در حین انتقال و یا رعایت ملاحظات ویژه در زمان انتقال از مرورگر به سمت سرور.

A4 : موجودیت های خارجی XML

بسیاری از پردازشگرهای XML قدیمی یا دارای پیکربندی ضعیف، مراجع مربوط به موجودیت بیرونی را درون سند XML ارزیابی می کنند. موجودیت های بیرونی می توانند توسط هکرها برای افشای URI فایل های داخلی ، اشتراک گذاری آن ها، اسکن پورت، اجرای کد از راه دور و انجام حملات منع سرویس(DoS) به کار روند.

A5 : کنترل دسترسی شکسته شده

اغلب مواقع برای کاربران احرازهویت شده محدودیت های لازم جهت کنترل دسترسی به درستی تعریف نمی شود. هکرها از این دسته آسیب پذیری ها برای مواردی چون دسترسی به داده ها یا قابلیت های غیرمجاز برنامه مثل دسترسی به اطلاعات سایر حساب های کاربری، مشاهده فایل های حساس برنامه، تغییر اطلاعات کاربران دیگر و تغییر حقوق دسترسی افراد استفاده می نمایند.

A6 : پیکربندی نامناسب امنیتی

این مورد در برنامه های کاربردی بسیار رایج بوده که نتیجه یک پیکربندی پیش فرض ناقص و ناامن، پیکربندی اشتباه پارامترهای هدر http و یا پیغام های خطایی که اطلاعات مهمی را به کاربر نمایش می دهند، است. برای مقابله با این نوع آسیب پذیری ها، نه تنها سیستم عامل ها، زیرساخت های طراحی و برنامه های کاربردی باید به شکل صحیح پیکربندی شوند بلکه بایستی به طور منظم به روز رسانی شده و وصله های امنیتی نصب شوند.

A7 : Cross-Site Scripting(XSS)

آسیب پذیری XSS زمانی رخ می دهد که یک برنامه بدون رعایت اعتبارسنجی ورودی و پاکسازی ورودی، امکان ورود داده های نامطمئن درون صفحات وب را فراهم کرده باشد. XSS به هکر امکان می دهد تا اسکریپت هایی را دررون مرورگر کاربران اجرا نموده و از این طریق نشست های کاربر را سرقت کرده ، کاربر را به سایت های جعلی هدایت نموده یا یک وب سایت را دیفیس نمایند.

A8 : Insecure Deserialization

این آسیب پذیری اغلب منجر به اجرای کد از راه دور می گردد. حتی در صورتی که این باگ منجر به اجرای کد از راه دور نشود می تواند حملاتی همچون حملات replay ، حملات تزریق و یا حملات ارتقای سطح دسترسی را به دنبال داشته باشد.

A9 : استفاده از کامپوننت های آسیب پذیر

کامپوننت ها مانند کتابخانه ها، فریم ورک ها و ماژول های نرم افزاری با همان دسترسی برنامه کاربردی اجرا می شوند. در صورت اکسپلویت یک کامپوننت آسیب پذیر، می تواند کل یک برنامه و داده های آن مورد نفوذ واقع شود. برنامه ها و APIهایی که از کامپوننت های با آسیب پذیری شناخته شده استفاده می کنند، می توانند امنیت برنامه را تضعیف کرده و منجر به انجام حملات متعدد علیه برنامه گردند.

نظارت و پایش ناکافی : A10

نظارت و پایش ناکافی در کنار نبود رویه پاسخگویی مناسب به حوادث امنیتی به مهاجمین امکان می دهد تا به سیستم های متعدد نفوذ کرده ، دسترسی خود را ماندگار نموده و از طریق یک سیستم به سایر سیستم های متصل شبکه نیز وارد شوند.

۳) آناتومی ده آسیب پذیری برتر وب

۳-۱) حمله تزریق

- **بردار حمله** : تقریباً هر منبع داده ای می تواند یک بردار برای حمله تزریق باشد. متغیرهای محیطی، پارامترهای ورودی، وب سرویس ها و کاربران برنامه همگی می توانند عامل بالقوه حمله باشند. این آسیب پذیری زمانی رخ می دهد که مهاجم کدمخرب را در قالب داده های مجاز به یک مفسر مانند مفسر زبان SQL ارسال کند.
- **نقاط ضعف امنیتی** : این نوع آسیب پذیری به خصوص در برنامه های قدیمی بسیار شایع است و معمولاً در پرس وجوهای SQL ، LDAP ، XParh و NoSQL ، دستورات OS، پارسرهای XML ، هدرهای SMTP و پرس و جوهای ORM یافت می شود. این نوع آسیب پذیری در زمان بررسی کدها به راحتی قابل شناسایی است. اسکنرها و فازرها به هکرها کمک می کنند تا این گونه آسیب پذیری را سریع تر و راحت تر کشف کنند.
- **تاثیرات حمله** : این نوع آسیب پذیری منجر به از بین رفتن یا خرابی داده ها یا مختل شدن دسترسی به برنامه می شوند. میزان تاثیر آن در کسب و کار بستگی به میزان محافظت از برنامه و داده های مرتبط در برابر این نوع حمله دارد.
- **مشخصات برنامه های آسیب پذیر** :

۱- داده های وارد شده توسط کاربر توسط برنامه اعتبارسنجی، فیلتر و پاکسازی نشود.

۲- داده های مخرب مستقیماً توسط پرس و جوهای پویا یا غیرپارامتریک برای مفسر ارسال شوند.

۳- داده های مخرب در پارامترهای جستجوی ORM به کار روند تا رکوردهای اضافه تری را استخراج کنند.

۴- داده های مخرب به صورت معمولی یا پشت سرهم به کار روند به نحوی که عبارت SQL حاوی هم داده ساختاری و هم مخرب در پرس و جوهای پویا یا رویه های ذخیره شده باشد.

- **نحوه مقابله** : برای مقابله با این نوع حمله باید داده ها از دستورات یا پرس و جوها تفکیک شود.

۱- یک روش خوب استفاده از یک API امن است که مانع استفاده از مفسر می شود و یا یک واسط پارامتریک ارائه می کند و یا با استفاده از ابزارهای ORM .

۲- استفاده از یک لیست سفید یا لیست کاراکترهای مجاز در سمت سرور. این روش همواره کارساز نیست زیرا برخی برنامه ها نیازمند استفاده از کاراکترهای خاصی هستند.

۳- برای بقیه پرس و جوهای پویا، کاراکترهای خاص را با روش مختص آن مفسر پاکسازی کنید. اسامی موجود در ساختار زبان SQL مانند نام جدول یا فیلدها را نمی توان به صورت دقیقی پاکسازی کرد.

۴- با دستور LIMIT یا سایر دستورات کنترلی SQL درون پرس و جو، جلوی دسترسی انبوه به رکوردها را در حمله تزریق بگیرید.

۳-۲) احراز هویت شکسته شده

- **بردار حمله** : هکرها با استفاده از لیست های اکانت های پیش فرض،
- **نقاط ضعف امنیتی** : انتشار این نوع حمله به دلیل نحوه طراحی و پیاده سازی اغلب سیستم های کنترل دسترسی بسیار فراگیر است. مدیریت نشست به عنوان بستر اصلی برای احرازهویت و کنترل دسترسی در اغلب برنامه های کاربردی به چشم می خورد. هکرها با استفاده از ابزارها و لیست پسوردها و دیکشنری های اختصاصی اقدام به شناسایی احرازهویت های شکسته شده می کنند.
- **تاثیرات حمله** : هکرها برای دراختیار گرفتن برنامه یا سیستم مقصد از تعداد محدودی اکانت یا در اغلب موارد اکانت مدیر سیستم استفاده می کنند. بسته به نوع برنامه دسترسی به اطلاعات احرازهویتی می توانند از جنبه های مالی، امنیتی و یا حقوقی برای هکرها سودمند باشد.
- **مشخصات برنامه های آسیب پذیر** :

تایید و تصدیق شناسه و هویت کاربر و اطلاعات نشست وی برای محافظت در برابر حملات مرتبط با احرازهویت ضروری است. موارد زیر به عنوان نقاط ضعف در برنامه های آسیب پذیر قابل ذکر است:

۱- برنامه امکان انجام حملات خودکار دیکشنری را با استفاده از لیست نام کاربری و پسورد توسط هکر فراهم کند.

۲- برنامه امکان انجام حملات بروت فورس را فراهم کند.
۳- برنامه امکان استفاده از نام کاربری و پسوردهای پیش فرض(مانند admin/admin) یا پسوردهای ضعیف را برای کاربران مهیا نماید.
۴- برنامه از رویه های ضعیف یا ناکارآمد برای بازیابی پسوردهای فراموش شده استفاده کند.
۵- برنامه فاقد مکانیسم احرازهویت چندعاملی باشد.
۶- برنامه مقدار Session ID را درون URL قرار داده باشد.
۷- برنامه مقدار Session ID را بعد از لاگین موفق تغییر نداده باشد.
۸- برنامه به طور صحیح مقادیر Session ID و توکن های احرازهویت به ویژه توکن های SSO را بعد از خروج کاربر از سیستم یا بیکار ماندن وی، غیرمعتبر ننموده باشد.

- **نحوه مقابله :**

۱- برنامه در صورت امکان باید از مکانیسم احرازهویت چندعاملی برای جلوگیری از حملات خودکار بروت فورس و استفاده مجدد از حساب های کاربری شنود شده استفاده کند.
۲- برنامه به هیچ عنوان امکان استفاده از حساب های پیش فرض را به ویژه برای حساب مدیر سیستم ندهد.
۳- از یک مکانیسم برای تست ضعیف بودن پسورد استفاده کند به عنوان مثال پسوردهای جدید کاربر را با لیستی از بدترین پسوردهای قابل انتخاب مقایسه کند و درصورت امکان، مانع انتخاب پسوردهای بد توسط کاربر شود.
۴- برنامه باید سیاست های امنیتی مربوط به پسورد را بر اساس خطوط راهنما مانند NIST 800 تعیین کند تا مواردی همچون طول پسورد، پیچیدگی و تاریخچه تعداد تغییرات را در نظر بگیرد.
۵- برنامه تلاش های ناموفق ورود به سیستم را محدود نموده یا برای آن وقفه ایجاد نماید تا جلوی اجرای حملات بروت فورس را بگیرد.
۶- از یک سیستم مدیریت نشست داخلی امن در سمت سرور برای ایجاد Session ID های جدید بعد از لاگین کاربر استفاده کنید. این شناسه نباید در URL مرورگر ذخیره شود و باید بعد از خروج کاربر از برنامه یا سپری شدن زمان بیکاری کاربر منقضی گردد.

۳-۳) افشای اطلاعات مهم

- **بردار حمله** : هکرها معمولاً به دنبال اطلاعات رمزشده نیستند و به جای آن به سرقت کلیدرمز، شنود داده های در حال انتقال با استفاده از حملات MitM یا کشف اطلاعات متنی ساده انتقال یافته بین مرورگر

کاربر و سرور هستند. در حین این عملیات پسوردهای پایگاه داده کشف شده یا با استفاده از زیرساخت های قدرتمند پردازشی کرک می شود.

- **نقاط ضعف امنیتی** : این حمله یکی از تاثیرگذارترین حملات در طی سالیان اخیر بوده است. ایراد اصلی فقط به خاطر رمزنگاری نشدن داده های مهم نیست بلکه در فرآیند رمزنگاری، مدیریت و ضعف در تولید کلید رمز، ضعف در الگوریتم رمزنگاری و پروتکل های مورد استفاده و تکنیک های هش موجود نقش مهمی در این خصوص ایفا می کنند. کشف نقاط ضعف موجود در داده های در حال انتقال آسان تر از داده های ذخیره شده بر روی سرور است.
- **مشخصات برنامه های آسیب پذیر** :

اولین گام تعیین نیازهای حفاظتی مورد نیاز در داده های در حال انتقال و ذخیره شده است. به طور مثال پسوردها، اطلاعات کارت های اعتباری، اطلاعات خصوصی افراد و اطلاعات محرمانه سازمانی نیازمند مراقبت ویژه هستند. در این خصوص بررسی موارد زیر در برنامه پیشنهاد می شود:

۱- آیا داده های ارسالی برنامه به صورت متنی ساده ارسال می شوند؟ این گونه اطلاعات از طریق پروتکل هایی همچون ftp، http و smtp ارسال می شوند.
۲- آیا داده هایی که به صورت متنی ساده مبادله می شوند دارای نسخه پشتیبان نیز هستند؟
۳- آیا از الگوریتم های ضعیف و قدیمی رمزنگاری در کدهای برنامه استفاده می شود؟
۴- آیا از کلیدهای پیش فرض در رمزنگاری استفاده می شود؟ آیا از کلیدهای رمزنگاری ضعیف استفاده می شود یا سیستم مدیریت کلید مناسب مورد استفاده قرار می گیرد؟
۵- آیا استفاده از رمزنگاری در ارتباط مرورگر با برنامه اجبار شده است؟
۶- آیا در صورت نامعتبر بودن گواهینامه ارسالی سرور، برنامه سمت کلاینت قادر به شناسایی است؟

- **نحوه مقابله** :

۱- داده های پردازش شده، ذخیره شده یا در حال انتقال برنامه را طبقه بندی کنید و بر اساس نوع کسب و کار سازمان و نیازمندی ها و الزامات کاری تعیین کنید چه داده هایی برای سازمان مهم و حساس هستند.
۲- برای هر نوع طبقه بندی داده، کنترل هایی را تعیین و اجرا کنید.
۳- داده های مهم که مورد نیاز نیستن را نگهداری و ذخیره سازی نکنید. در صورت امکان با روش های امن نسبت به امحای داده های غیرضروری حساس اقدام کنید.
۴- اطمینان حاصل کنید که تمامی داده های ذخیره شده مهم رمزنگاری شده اند.

۵- اطمینان حاصل کنید که از الگوریتم ها و پروتکل های جدید و قدرتمند با ساختار مدیریت کلید مناسب در برنامه ها استفاده کرده اید.
۶- داده های در حال انتقال برنامه را با پروتکل های امنی همچون TLS با پارامترهای مناسب رمزنگاری کنید. با استفاده از مکانیسم هایی مانند HSTS استفاده از رمزنگاری را اجبار کنید.
۷- قابلیت caching را برای پاسخ هایی از برنامه که شامل اطلاعات حساس است غیرفعال نمایید.
۸- پسوردها را با استفاده از توابع هش دارای salt و با استفاده از فاکتورهایی همچون Argon2 ,bcrypt , scrypt وPBKDF2 ذخیره سازی کنید.
۹- به صورت متناوب از صحت عملکرد تنظیمات امنیتی انجام شده اطمینان حاصل کنید.

۳-۴) موجودیت های خارجی XML

- **بردار حمله** : هکرها می توانند پردازشگرهای XML آسیب پذیر را با آپلود کردن فایلXML و قرار دادن محتوای مخرب درون سند آن، اکسپلویت کنند.
- **نقاط ضعف امنیتی** : به طور پیش فرض بسیاری از پردازشگرهای XML قدیمی، امکان تعیین یک موجودیت خارجی را می دهند که امکان ارجاع دهی و ارزیابی مجدد در حین پردازش XML را فراهم می کند. ابزارهای استاتیک یا SAST با بررسی وابستگی ها و پیکربندی این کار را انجام می دهند و ابزارهای DAST نیاز به مراحل اضافه تری برای شناسایی و اکسپلویت دارند.
- **تاثیرات حمله** : این نوع آسیب پذیری می تواند منجر به استخراج داده ها، اجرای یک درخواست راه دور از طریق سرور، اسکن سیستم های داخلی، اجرای یک حمله منع سرویس یا DoS و مواردی از این قبیل گردد.
- **مشخصات برنامه های آسیب پذیر** : برنامه ها یا به طور خاص وب سرویس های مبتنی بر XML، در صورتی در مقابل این نوع حمله آسیب پذیر هستند که :

۱- برنامه فایل XML را به طور مستقیم و از منابع نامطمئن دریافت کند یا امکان اضافه شدن داده های نامطمئن درون اسناد XML را بدهد. این فایل آلوده بعداً توسط یک پردازشگر XML، پردازش می گردد.
۲- همه پردازشگرهای XML برنامه یا وب سرویس های مبتنی بر SOAP دارای قابلیت DTD[1] فعال باشند. فرآیند غیرفعال کردن قابلیت DTD برای پردازش‌گرهای XML مختلف متفاوت است.

[1] Document Type Definitions

۳- اگر برنامه کاربردی برای تامین اهدافی همچون امنیت جامع یا SSO[1] از قابلیت SAML برای پردازش شناسه هویتی استفاده کند. قابلیت SAML از XML برای شناسایی هویت استفاده می کند که می تواند آسیب پذیر باشد.

۴- اگر برنامه از نسخه قبل از SOAP 1.2 استفاده کند، درصورت ارسال موجودیت XML به چارچوب SOAP، احتمال وقوع حمله XXE وجود خواهد داشت.

۵- آسیب پذیر بودن برنامه در برابر حمله XXE به معنای آسیب پذیر بودن برنامه در برابر حمله DoS است.

- **نحوه مقابله** : برای شناسایی و مقابله با حملات XXE، آموزش برنامه نویسان لازم است. در کنار آن برای مقابله و جلوگیری از این حملات، بایستی :

۱- درصورت امکان از فرمت های داده با پیچیدگی کمتر مانند JSON استفاده کنید.

۲- تمامی پردازش‌گرهای XML و کتابخانه های مورد استفاده در برنامه یا لایه های سیستم عامل را به روز رسانی و وصله کنید . از چک کننده های وابستگی استفاده کنید و SOAP را به SOAP 1.2 ارتقا دهید.

۳- در تمامی پردازش‌گرهای XML برنامه، XXE و DTD را غیرفعال کنید.

۴- اعتبار سنجی ورودی و پاکسازی ورودی بر اساس لیست سفید را در سمت سرور برای جلوگیری از تزریق داده های آلوده درون اسناد XML پیاده سازی کنید.

۵- قابلیت آپلود فایل XML یا XSL، بایستی ورودی را بر اساس اعتبارسنجی XSD مورد بررسی قرار دهد.

۶- ابزارهای SAST برای شناسایی XXE در سورس کد برنامه به کار می روند. اگرچه در برنامه های بزرگ و پیچیده بازبینی دستی کد جایگزین مناسب تری است.

۷- در صورت عدم امکان پیاده سازی این کنترل ها، میتوان با استفاده از WAF نسبت به شناسایی یا بلاک کردن حملات XXE اقدام کرد.

۳-۵) کنترل دسترسی شکسته شده

- **بردار حمله** : اکسپلویت کنترل دسترسی یک مهارت پایه ای برای هکرها محسوب می شود. ابزارهای SAST و DAST قادر به شناسایی نبود کنترل دسترسی مناسب در برنامه ها هستند اما نمی توانند تشخیص دهند که در صورت وجود به چه میزان کارآمد است.

[1] Single Sign On

- **نقاط ضعف امنیتی** : ضعف در کنترل دسترسی به دلیل نبود قابلیت شناسایی خودکار یا عدم بررسی مناسب توسط برنامه نویس، بسیار رایج است.
- **مشخصات برنامه های آسیب پذیر** :

آسیب پذیری های رایج مرتبط با کنترل دسترسی عبارتند از :

۱- دور زدن کنترل دسترسی با تغییر URL یا تغییر حالت برنامه یا صفحه HTML رخ می دهد.
۲- امکان تغییر کلید اصلی به رکورد کاربر دیگری در برنامه برای مشاهده یا ویرایش حساب کاربری فرد دیگر.
۳- ارتقای سطح دسترسی یک کاربر عادی به راهبر سامانه یا ارتقای دسترسی به یک کاربر عادی بدون لاگین شدن به برنامه.
۴- امکان تغییر در متادیتای برنامه مثل دستکاری کوکی یا یک فیلد مخفی یا توکن کنترل دسترسی برای ارتقای سطح دسترسی موجود.
۵- امکان مرور صفحاتی از برنامه که نیازمند احرازهویت هستند یا امکان مشاهده صفحات ادمین توسط کاربر عادی یا دسترسی به API ها بدون کنترل دسترسی برای عملیات Post ، PUT و Delete.

- **نحوه مقابله** :

کنترل دسترسی تنها در صورتی کارآمد است که در کد سمت سرور اعمال شود زیرا هکر امکان ایجاد تغییر در کنترل دسترسی را ندارد.

۱- مکانیسم های کنترل دسترسی را یکبار پیاده سازی نموده و درون برنامه از آن ها استفاده کنید.
۲- مدل کنترل دسترسی باید بر اساس مالکیت رکورد باشد و امکان ایجاد، مشاهده، تغییر یا حذف تمامی رکوردها را به کاربر ندهد.
۳- محدودیت های کسب و کاری باید در مدل کنترل دسترسی در نظر گرفته شود.
۴- اماکن فهرست گیری از دایرکتوری وب سرور را غیرفعال کنید و اطمینان یابید که فایل های متادیتا و پشتیبان درون شاخه ریشه برنامه قرار نداشته باشد.
۵- لاگ مربوط به شکست های کنترل دسترسی را ثبت نموده و امکان اطلاع رسانی به راهبر برنامه را فعال کنید.
۶- توکن های JWT باید پس از خروج کاربر از برنامه نامعتبر شوند.

۳-۶) پیکربندی نادرست امنیتی

- **بردار حمله** : هکرها اغلب به دنبال پسوردهای پیش فرض، صفحات بلا استفاده برنامه، آسیب پذیری های وصله نشده، فایل ها و دایرکتوری های محافظت نشده برای دسترسی غیرمجاز به برنامه هستند.
- **نقاط ضعف امنیتی** : پیکربندی نادرست امنیتی در هر سطحی از یک برنامه کاربردی شامل سرویس های شبکه، وب سرور، سرور برنامه کاربردی، پایگاه داده، کدهای اختصاصی، چارچوب برنامه، ماشین مجازی پیش فرض و فضای ذخیره سازی می تواند رخ دهد. اسکنرهای خودکار برای شناسایی آسیب پذیری های پیکربندی های نادرست امنیتی، اکانت های پیش فرض و سرویس های غیرضروری مفید هستند.
- **تاثیرات حمله**: اغلب آسیب پذیری ها برای مهاجمین امکان دسترسی غیرمجاز به داده ها و قابلیت های موجود در برنامه ها را فراهم می کنند. تقریباً بیشتر آنها موجب دسترسی کامل هکر به سیستم آسیب پذیر می شوند.
- **مشخصات برنامه های آسیب پذیر** :

۱- عدم وجود هر یک از قابلیت های مقاوم سازی در هر یک از لایه های برنامه

۲- نصب و فعال سازی قابلیت های غیرضروری مثل پورت ها، سرویس ها، حساب های کاربری و حقوق دسترسی

۳- حساب های کاربری و پسوردهای موجود در برنامه فعال بوده و بدون تغییر باقی مانده است.

۴- برای سیستم های به روز رسانی شده، آخرین قابلیت های امنیتی غیرفعال است یا به درستی پیکربندی نشده است.

۵- تنظیمات امنیتی در سرورهای برنامه کاربردی، زیر ساخت ها و کتابخانه ها و لایه پایگاه داده بدون درنظرگرفتن ملاحظات امنیتی انجام شده است.

۶- نرم افزارهای نصب شده روی سرور برنامه قدیمی و آسیب پذیر هستند.

۷- بدون انجام رویه های امنیتی مراقبتی دوره ای، سیستم در سطح ریسک بالاتری قرار خواهد گرفت.

- **نحوه مقابله** :

برای پوشش مشکلات مرتبط با پیکربندی نادرست امنیتی در برنامه باید فرآیندهای نصب امن اجرا شود که شامل موارد زیر است:

۱- محیط برنامه باید فاقد هر گونه قابلیت ها و مولفه های غیرضروری باشد که بایستی در صورت وجود حذف شوند.

۲- بازبینی و پیکربندی تمامی تنظیمات امنیتی و به روز رسانی و وصله نقاط ضعف موجود به عنوان بخشی از فرآیند مدیریت وصله های امنیتی.
۳- در یک برنامه با معماری بخش بندی شده باید یک جداسازی امن بین مولفه های مختلف برنامه وجود داشته باشد.
۴- راهنمایی های امنیتی لازم همواره باید در اختیار کاربران برنامه قرار داده شود.
۵- باید یک فرآیند خودکار جهت تعیین کارآیی تنظیمات و پیکربندی امنیتی صورت گرفته تعریف شود.

۳-۷) حمله XSS

- **بردار حمله** : ابزارهای خودکار قابلیت شناسایی و اکسپلویت هر سه نوع آسیب پذیری XSS را دارند که در زیرساخت هایی نظیر php ، J2EE / JSP و ASP.Net به خوبی عمل می کنند.
- **نقاط ضعف امنیتی** :این حمله از لحاظ وسعت در مقام دوم حملات وب قرار داشته و حدود دو سوم برنامه های وب در مقابل این حمله آسیب پذیر هستند.
- **تاثیرات حمله**: تاثیر حمله XSS برای دو نوع حمله Reflected و DOM متوسط و برای حمله نوع Stored شدید است که موجب اجرای کد از راه دور در مرورگر قربانی می شود و منجر به سرقت حساب کاربری، شناسه نشست کاربر یا تحویل بدافزار به قربانی می شود.
- **مشخصات برنامه های آسیب پذیر** : سه نوع حمله XSS، مرورگر کاربران را هدف قرار می دهند که عبارتند از:

۱- Stored XSS**:** در این حالت برنامه کاربردی یا API ورودی غیرپاکسازی شده ای از کاربر را در برنامه ذخیره می کند که بعداً توسط یک کاربر دیگر یا راهبر برنامه مشاهده می شود. این آسیب پذیری اغلب بحرانی بوده و درجه ریسک بالایی برای آن در نظر گرفته می شود.
۲- Reflected XSS**:** در این حالت برنامه کاربردی یا API دارای ورودی اعتبارسنجی نشده یا فیلتر نشده ای است که به عنوان بخشی از خروجی HTML ظاهر می شود. یک حمله موفق به مهاجم امکان می دهد تا کد HTML یا جاوااسکریپت موردنظرش را در مرورگر قربانی اجرا نماید. اغلب به کاربر یک لینک آلوده ارائه می شود که موجب هدایت وی به سمت یک صفحه مخرب مربوط به هکر می گردد.
۳- DOM XSS**:** زیرساخت های جاوا اسکریپتی، برنامه های یک صفحه ای و API های حاوی داده های قابل کنترل توسط هکر بر روی یک صفحه هستند، نسبت به این نوع حمله آسیب پذیرند. در حالت ایده آل برنامه نبایستی اجازه ارسال داده های قابل کنترل هکر را به API های ناامن جاوا اسکریپت بدهد.

معمولاً حملات XSS منجر به مواردی همچون سرقت نشست یا حساب کاربری، دور زدن MFA، خرابکاری در نودهای DOM و حمله برعلیه مرورگر کاربران(مانند دانلود برنامه های آلوده و ثبت کلیدهای فشرده شده کاربر) می گردد.

- **نحوه مقابله :**

جلوگیری از این حملات نیازمند جداکردن داده های نامطمئن از کد مربوط به محتویات مرورگر است که از طریق انجام موارد زیر حاصل می گردد:

۱- استفاده از چارچوب هایی که به طورخودکار جلوی این حملات را می گیرند مثل Ruby on Rails ، React JS.

۲- فیلتر کردن داده های مربوط به درخواست های نامعتبر HTTP بر اساس محتوای خروجی HTML (body, attribute, JavaScript, CSS, URL) جلوی حملات Stored XSS و Reflected XSS را می گیرد.

۳- هنگام ویرایش سندی از مرورگر در سمت کاربر، فعال سازی قابلیت کدگذاری حساس به متن می تواند جلوی حملات DOM XSS را بگیرد.

۴- فعال سازی یک سیاست CSP[1] یک کنترل دفاع در عمق مناسب برای مقابله با حمله XSS است. این قابلیت در صورتی که آسیب پذیری دیگری که امکان آپلود کد یا فایل آلوده بر روی سرور را فراهم کند، وجود نداشته باشد.

۳-۸) Deserialization نا امن در برنامه

- **بردار حمله** : اکسپلویت کردن Deserialization در برنامه ها تا حدودی مشکل است زیرا این قبیل اکسپلویت ها بدون انجام تغییر در کد اکسپلویت لایه های پایین تر به درستی کار نمی کنند.
- **نقاط ضعف امنیتی** : برخی از ابزارهای اسکن آسیب پذیری قابلیت شناسایی این نوع باگ را دارند اما برای اعتبار بخشیدن به آن وجود عامل انسانی مورد نیاز است. حجم داده مورد نیاز برای شناسایی و یافتن این نوع آسیب پذیری ها با بزرگ شدن برنامه ها اغلب افزایش می یابد.
- **تاثیرات حمله**: این آسیب پذیری موجب بروز حملات اجرای کد از راه دور می شود که یکی از خطرناک ترین حملات موجود محسوب می گردد. با توجه به نیاز کسب و کار و نوع داده های برنامه کاربردی، سطح محافظت در برابر این نوع حملات متغیر خواهد بود.

[1] Content Security Policy

- **مشخصات برنامه های آسیب پذیر** : برنامه کاربردی یا API ویندوزی در صورتی که اشیا یا مولفه های آلوده ارائه شده از سوی هکر را Deserialize کنند، نسبت به این حمله آسیب پذیر هستند.

Deserialization عکس عمل سریال سازی در برنامه کاربردی است. سریال سازی در برنامه زمانی برای موارد زیر به کار می رود:

- در فرآیندهای RPC/IPC[1]
- پروتکل های شبکه های سیمی، وب سرویس ها و واسطه های تبادل پیام[2]
- کش کردن / ماندگار سازی در برنامه
- در پایگاه های داده، سرورهای کش و سیستم فایل ها
- کوکی های HTTP ، پارمترهای فرم HTML و توکن های احرازهویت API

این حملات در دو دسته اصلی خلاصه می شود که عبارتند از :

۱- اشیا و ساختارهای داده ای مرتبط با حمله در زمانی که هکر ساختار منطقی برنامه را تغییر داده یا قابلیت اجرای کد از راه دور را به دست می آورد. این در صورتی است که کلاس هایی در برنامه وجود داشته باشد که رفتار موجود را در زمان Deserialization یا بعد از آن تغییر دهد.

۲- حملات از نوع دستکاری داده ها مانند حملات مرتبط با کنترل دسترسی که در آن محتویات ساختار داده ای دچار تغییر می شود.

- **نحوه مقابله** : تنها روش ساختاری ایمن برای جلوگیری از این نوع حملات، عدم دریافت اشیا و مولفه های سریال سازی شده از منابع نامعتبر و هم چنین استفاده از ابزارهای سریال سازی است که تنها اجازه استفاده از انواع داده اصلی را می دهد.

درصورتی که موارد فوق میسر نباشد باید :

۱- کنترل های مربوط به جامعیت داده مانند امضای دیجیتال را برای تمامی اشیای سریال سازی شده اعمال کنید تا جلوی ایجاد اشیاء آلوده یا دستکاری داده ها را بگیرید.

۲- محدودیت های مربوط به نوع داده صریح[3] در حین انجام Deserialization و قبل از ایجاد اشیا را اعمال کنید.

[1] Remote Process Communication / Inter Process Communication

[2] Message Brokers

[3] Strict data type

۳- درصورت امکان، کد مربوط به Deserialization را در محیطی با دسترسی محدود ایزوله و اجرا نمایید.
۴- لاگ مربوط به خطاها و استثنائات Deserialization را ثبت کنید مثلاً جایی که نوع داده دریافتی مغایر نوع داده مورد نظر نیست یا یک استثناء در حین کار رخ داده است.
۵- اتصالات شبکه ای مربوط به سرورهایی که عمل Deserialization را انجام می دهند را پایش کرده و محدود نمایید.

۳-۹) استفاده از کامپوننت های آسیب پذیر در برنامه

- **بردار حمله** : معمولاً یافتن اکسپلویت های آماده برای آسیب پذیری های شناخته شده، کار چندان دشواری نیست. اما نوشتن یک اکسپلویت جدید و اختصاصی برای یک آسیب پذیری نیاز به تلاش زیادی دارد.
- **نقاط ضعف امنیتی** : در توسعه و استفاده از کامپوننت های در برنامه، معمولاً تیم برنامه نویسی اطلاعات دقیقی از نحوه به روز رسانی کامپوننت های به کار رفته در برنامه و حتی فهرست آن ندارد. برخی اسکنرها نظیر retire.js قادر به شناسایی این موارد هستند اما تعیین قابلیت اکسپلویت کامپوننت نیاز به تلاش و صرف زمان بیشتری دارد.
- **تاثیرات حمله**: در حالی که برخی از آسیب پذیری های شناخته شده تاثیرات جزئی دارند اما بسیاری از نقض های امنیتی و افشای اطلاعات عمده، ناشی از اکسپلویت آسیب پذیری های شناخته شده در کامپوننت ها هستند.
- **مشخصات برنامه های آسیب پذیر** :

۱- در صورتی که اطلاعات دقیقی از کامپوننت های مورد استفاده در برنامه (کامپوننت های سمت سرور و سمت کلاینت) و نسخه آن موجود نباشد. این مورد شامل کامپوننت های استفاده شده در برنامه به طور مستقیم یا وابسته به سایر کامپوننت ها می باشد.
۲- برنامه هایی که هر یک از قسمت های درگیر در برنامه به روز نباشند. این مورد شامل سیستم عامل، وب سرور، پایگاه داده، APIها و کامپوننت های برنامه و سایر کتابخانه های زمان اجرا است.
۳- در صورتی که اسکن آسیب پذیری ها به صورت منظم انجام نشود یا از طریق بولتن های امنیتی، آسیب پذیری های مربوط به کامپوننت های برنامه و وصله احتمالی آن ها نظارت نگردد.
۴- در صورتی که زیرساخت ها و چارچوب های برنامه به صورت منظم به روز رسانی نشود. این موضوع شامل سازمان هایی است که مدیریت وصله های امنیتی به صورت نظام مند و دقیق صورت نمی گیرد و آسیب پذیری شناخته شده، مدت ها به صورت فعال باقی می ماند.

۵- در صورتی که پیکربندی کامپوننت ها به صورت امن انجام نشده باشد.

- **نحوه مقابله :**

۱- برنامه باید رویه های مناسبی برای حذف وابستگی های بلا استفاده و قابلیت ها ،کامپوننت ها، فایل ها و مستندات غیر ضروری داشته باشد.

۲- برنامه باید رویه های مناسبی برای کنترل مستمر نسخه کامپوننت ها و وابستگی آنها به یکدیگر در سمت کلاینت و سرور داشته باشد. اینکار با ابزارهایی همچون versions، DependencyCheck و retire.js قابل انجام است.

۳- به طور مداوم منابعی مانند CVE و NVD را کنترل نمایید تا آسیب پذیری های عمومی شده در کامپوننت ها را شناسایی کنید و از ابزارهایی برای تحلیل خودکار این موارد استفاده نمایید و با عضویت در منابع خبری مرتبط از آخرین اخبار آسیب پذیری ها مطلع شوید.

۴- کامپوننت ها را فقط از منابع معتبر و رسمی دریافت کنید و در صورت امکان از بسته های تایید شده برای کاهش خطر وجود کامپوننت های آلوده و تغییر یافته استفاده نمایید.

۵- کامپوننت هایی که نسخه های قدیمی بوده و فاقد وصله های امنیتی هستند را کنترل نمایید و در صورت عدم وجود و انتشار وصله برای آن ها، از یک Virtual Patch برای پایش، شناسایی و محافظت در برابر تهدیدات استفاده نمایید.

۳-۱۰) نظارت و پایش ناکافی

- **بردار حمله** : نظارت و پایش ناکافی زمینه ساز هر نوع حادثه مهم امنیتی محسوب می شود. هکرها با تکیه بر عدم وجود نظارت و واکنش مناسب و سریع از سوی هدف، بدون ترس از شناسایی و به دام افتادن به دنبال رسیدن به اهداف مورد نظرشان خواهند بود.
- **نقاط ضعف امنیتی** : یکی از راهکارهای تشخیص مناسب بودن نظارت و پایش، بررسی لاگ های مرتبط با تست نفوذ است. تمامی عملیات انجام شده در حین تست نفوذ بایستی ثبت گردد تا خسارات و آسیب های احتمالی قابل پیش بینی و سنجش باشد.
- **تاثیرات حمله**: بسیاری از حملات موفق با کنکاش و بررسی دقیق آسیب پذیری ها شروع می شود. اغلب آسیب پذیری های واقعی در نهایت می تواند منجر به اکسپلویت سیستم گردد.
- **مشخصات برنامه های آسیب پذیر** : نظارت، پایش و واکنش نامناسب به رخدادهای امنیتی زمانی اتفاق می افتد که :

۱- رخدادهای قابل ممیزی شامل لاگین های موفق و ناموفق، تراکنش های با ارزش بالا به درستی ثبت نشود.
۲- لاگ های ثبت شده برنامه کاربردی جهت عملیات بدخواهانه به درستی پایش نشود.
۳- کیفیت پاسخ و واکنش در برابر رویدادهای امنیتی، متناسب با سطح ریسک داده های برنامه نباشد.
۴- تست نفوذ و اسکن با ابزارهای خودکار موجب فعال شدن سیستم های امنیتی برنامه نشود.

در سازمان های بزرگ عدم وجود پاسخ مناسب به رویدادهای مهم مانند بلاک کردن خودکار حملات علیه برنامه های وب، سازمان را با تهدیدات اساسی مواجه خواهد کرد. این پاسخ حتماً نباید برای هکر قابل مشاهده باشد و تنها برنامه کاربردی و زیرساخت های مرتبط با آن را میتوان برای هشداردهی به عامل انسانی یا ابزارهای جلوگیری، برای واکنش سریع پیکربندی کرد.

- **نحوه مقابله :**

۱- اطمینان یابید که تمامی خطاهای لاگین، کنترل دسترسی و خطاهای اعتبارسنجی ورودی سمت سرور با اطلاعات کافی از کاربر، برای شناسایی حساب های کاربری مشکوک یا غیرمجاز ثبت شده است. ضمن اینکه این اطلاعات بایستی برای مدت زمان معینی نگهداری شوند تا امکان انجام عملیات فارنزیک و تحلیل های احتمالی آتی وجود داشته باشد.
۲- اطمینان یابید که تمامی لاگ ها با فرمتی تولید می شوند که امکان استفاده از آن ها توسط سیستم مدیریت مرکزی لاگ فراهم باشد.
۳- اطمینان یابید که تراکنش های ارزشمند برنامه دارای امکان ممیزی/ردگیری به همراه کنترل های جامعیت داده هستند تا امکان تغییر یا حذف آن ها میسر نباشد.
۴- یک سیستم نظارت و هشدار کارآمد راه اندازی کنید تا فعالیت های مشکوک شناسایی و در زمان مشخص به آن پاسخ مناسبی داده شود.
۵- یک روش اجرایی و برنامه جهت پاسخگویی به حوادث امنیتی و بازگشت از شرایط بحرانی داشته باشید(مطابق سند NIST 800-61)

بدین منظور برنامه ها و ابزارهای امنیتی متن باز یا تجاری متعددی نظیر OWASP AppSensor و فایروال های برنامه کاربردی نظیر modSecurity و همچنین نرم افزارهای تجمیع لاگ با امکان تولید داشبورد و هشدار موجود و قابل بهره برداری است.