عنوان مقاله: حمله SQL Injection چیست؟ صفرتاصد حمله تزریق SQL و راه های جلوگیری از آن

نویسنده مقاله: تیم فنی نیک‌آموز

تاریخ انتشار: ۹ اردیبهشت ۱۴۰۳

منبع: https://nikamooz.com/sql-injection-attack/

حمله SQL Injection نوعی سواستفاده امنیتی است که در آن، هکر یا اصطلاحاً Attacker، کد SQL را به فیلدهای ورودی یک وب‌اپلیکیشن یا کوئری پایگاه داده تزریق می‌کند. هدف از این کار، دستکاری کوئری SQL اپلیکیشن به‌منظور دسترسی به داده‌های حساس یا انجام اقدامات غیرمجاز است. در این مقاله، پس از بررسی چیستی حمله تزریق SQL، نحوه اجرا، چگونگی تشخیص و روش‌های جلوگیری از آن شرح داده می‌شوند.

آشنایی با ساختار کوئری ها

در ابتدا باید توجه داشت که برای شناسایی آسیب‌پذیری های حمله SQL Injection ، لازم است شما ساختار زبان SQL را درک کنید. SQL یک زبان کوئری‌نویسی ساختاریافته است که برای برقراری ارتباط و دستکاری پایگاه‌های داده، استانداردسازی شده است. به‌طور مثال، شما باید با مباحثی همچون **دستور Select** و **دستور آپدیت** و **دستور Insert** و **عبارت Where** و همچنین **نوع های داده در SQL Server** آشنایی داشته باشید. از میان **انواع سیستم مدیریت پایگاه داده رابطه ای (RDBMS)** ، می‌توان SQL Server را یکی از پرکاربردترین آن‌ها درنظر داشت. برای آشنایی بیشتر با این‌RDBMS ، پیشنهاد می‌شود به **مقاله آموزش SQL Server** مراجعه کنید.

حمله SQL Injection چیست؟

SQL Injection نوعی حمله (Attack) محسوب می‌شود که در آن، دستورات مخرب SQL درون فیلدهای ورودی یک وب‌اپلیکیشن تزریق می‌شوند تا بدین شیوه، **پایگاه داده** (Database) اپلیکیشن دستکاری شود. در این حمله، از اعتبارسنجی ضعیف ورودی و روش‌های کدگذاری ناامن سوءاستفاده می‌شود و Attacker ها این اجازه را خواهند داشت که دستورات دلخواه SQL را اجرا کنند. به‌طورکلی، انواع حملات SQL Injection ، از نوع کلاسیک آن گرفته تا نوع Blind آن، هر یک چالش‌های خاصی را ایجاد می‌کنند و لازم است از معیارهایی برای کاهش ریسک‌های مربوط به آن‌ها استفاده کرد.

حملات SQL Injection چطور اجرا می شوند؟

حمله SQL Injection از رویکردی گام‌به‌گام پیروی می‌کند؛ بدین شیوه، Attacker ها می‌توانند از آسیب‌پذیری‌های موجود در لایه پایگاه داده وب‌اپلیکیشن سوءاستفاده کنند. شما با درک صحیح فرآیند اجرای حمله تزریق SQL ، می‌توانید به شناسایی و کاهش اثرات آسیب‌پذیری‌های تزریق اسکریپت SQL بپردازید. نحوه اجرای حملات SQL Injection به شرح زیر است:

۱- شناسایی فیلدهای ورودی آسیب پذیر

تشخیص فیلدهای ورودی که مستقیماً داده‌های کاربر را بدون اعتبارسنجی مناسب در دستورات SQL قرار می‌دهند، یکی از گام‌های کلیدی در راه‌اندازی حمله SQL Injection محسوب می‌شود. این فیلدهای ورودی، نقاط ورود (Entry Point) بالقوه‌ای برای تزریق کد مخرب SQL و سوءاستفاده از آسیب‌پذیری‌های موجود در اپلیکیشن هستند. فرم‌های Login، کادر جستجو (Search Box) و فرم‌های ورودی کاربران، نمونه‌هایی از نقاط ورود متداول هستند.

۲- تست آسیب پذیری

پس از اینکه نقاط بالقوه تزریق شناسایی شدند، Attacker ها با به‌کارگیری تکنیک‌های مختلف مانند قراردادن کاراکترهای خاص و سینتکس SQL، نقاط تزریق را مورد تست و آزمایش قرار می‌دهند. در این بررسی‌ها، تعیین می‌شود که آیا اپلیکیشن به حملات تزریق آسیب‌پذیر است یا خیر. در این گام، جستجو برای پیام‌های خطا، رفتار غیرمنتظره یا نشت داده (Data Leakage)، که نشان‌دهنده آسیب‌پذیری‌های بالقوه هستند، صورت می‌گیرد.

۳- ساختن کد مخرب (Payload)

حمله‌کنندگان، [کدهای مخرب] (Payload) حمله SQL Injection را متناسب با آسیب‌پذیری‌های مشخصی که در طول تست پیدا کرده‌اند، طراحی می‌کنند. این کدهای مخرب به‌منظور دستکاری دستورات SQL و با هدف دستیابی به نتایج موردنظر ایجاد می‌شوند. استخراج اطلاعات حساس یا دور زدن احراز هویت، از مواردی هستند که به‌عنوان نتیجه از حمله SQL Injection انتظار می‌روند. توجه شود که براساس رفتار مشاهده‌شده از اپلیکیشن، Attacker می‌تواند از تکنیک‌های تزریق SQL مانند روش Union-Based ، Error-Based یا Time-Based استفاده کند.

۴- اجرای حمله

با ایجاد کد مخرب مناسب، Attacker ها ورودی مخرب را ازطریق فیلدهای ورودی آسیب‌پذیر ارسال خواهند کرد. وب‌اپلیکیشن بدون استفاده از اعتبارسنجی مناسب، ورودی را پردازش می‌کند و به‌دنبال آن، دستورات تزریق‌شده SQL اجرا می‌شوند.

۵- تجزیه و تحلیل نتایج

Attacker ها پاسخ اپلیکیشن را برای تعیین اینکه آیا حمله موفقیت‌آمیز بوده است یا خیر، تجزیه‌وتحلیل می‌کنند. اگر فرآیند موفقیت‌آمیز باشد، احتمالاً دستیابی به اطلاعات ارزشمندی میسر می‌شود. این دیتای ارزشمند می‌تواند محتوای پایگاه داده، پیام‌های خطایی که ساختار دیتابیس را آشکار می‌کنند یا سایر موارد باشد.

۶- تکرار و بهبود

ممکن است Attacker ها طبق نتایج مشاهده‌شده، شیوه‌های حمله خود را تکرار کرده، آن‌ها را بهبود بخشند و به‌طور مداوم، به‌دنبال آسیب‌پذیری‌های بیشتری باشند.

پیاده سازی فنی حمله SQL Injection

در این بخش قصد داریم با بررسی یک سناریو، حمله SQL Injection را شرح دهیم.

یک فرم ورود (Login) متشکل از دو فیلد برای نام کاربری و رمز عبور درنظر بگیرید که ازطریق کوئری زیر بررسی می‌شوند:

```
SELECT * FROM users WHERE username = 'input_username' AND password = 'input_password';
```

برای عبور از مرحله احراز هویت، فرد حمله‌کننده یا همان Attacker می‌تواند کد مخرب زیر را در فیلد Username تزریق کند.

```
' OR 1=1 -
```

کوئری حاصل، مشابه زیر خواهد بود:

```
SELECT * FROM users WHERE username = '' OR 1=1 --' AND password = 'input_password';
```

در کوئری فوق، دو خط تیره (–) باعث می‌شود تا بخش باقی‌مانده از کوئری، کامنت شده و غیرفعال بماند. درنتیجه، فرآیند بررسی رمز عبور دور زده خواهد شد و به فرد حمله‌کننده اجازه می‌دهد تا بدون دسترسی به رمز عبور معتبر وارد شود.

در کنار موارد فوق، به ملاحظات جانبی زیر نیز توجه کنید:

- حمله SQL Injection تنها برروی سیستم‌هایی مجاز است که اجازه صریح برای تست آن‌ها داشته باشید.
- برای اجتناب از آسیب‌رساندن به سیستم‌های عملیاتی یا نقض‌کردن اصول اخلاقی، احتیاط شرط است.
- هنگام انجام تست‌های امنیتی و تست نفوذ (Penetration Test)، همیشه قوانین و اصول اخلاقی را رعایت کنید.

جلوگیری از SQL Injection

جلوگیری از حملات تزریق SQL ، مستلزم استفاده ترکیبی از روش‌های امن کدگذاری، اعتبارسنجی ورودی و استفاده از تکنولوژی‌های دفاعی مناسب است. در این بخش، مهم‌ترین استراتژی‌های لازم برای جلوگیری از آسیب‌پذیری های حمله SQL Injection بررسی خواهند شد.

۱- استفاده کوئری های پارامتربندی شده (Parameterized) یا عبارات آماده

به جای الحاق مستقیم ورودی کاربر به کوئری SQL ، از کوئری های پارامتربندی شده یا Prepared Statement ها استفاده کنید که توسط زبان برنامه نویسی یا فریم‌ورک ارائه شده‌اند. این نوع کوئری‌ها، کد SQL را از ورودی کاربر جدا می‌کنند و از تزریق دستورات مخرب SQL توسط Attacker جلوگیری خواهند کرد.

۲- اعتبارسنجی و پاک سازی ورودی

اعتبارسنجی و پاک‌سازی ورودی باید به‌طور دقیق انجام شود تا این اطمینان به وجود آید که داده‌های کاربر با فرمت موردانتظار تطبیق دارند و حاوی کاراکترهای مخرب نیستند. شما باید ورودی را برمبنای یک لیست سفید (Whitelist) متشکل از کاراکترهای مجاز، اعتبارسنجی کنید یا از عبارات منظم (Regular Expression) برای اعمال محدودیت بر ورودی بهره ببرید.

۳- پیروی از اصل حداقل امتیاز (Least Privilege Principle)

از اصل حداقل امتیاز پیروی کنید و به کاربران پایگاه داده تنها تا سطح لازم برای عملکرد اپلیکیشن دسترسی دهید. استفاده از حساب‌های کاربری با سطح دسترسی بالا، مانند اکانت‌های دارای امتیازات Administrative، در کد برنامه اجتناب کنید.

۴- استقرار فایروال های وب اپلیکیشن (WAF)

با استقرار فایروال‌های وب‌اپلیکیشن (WAF) می‌توانید بر ترافیک ورودی HTTP نظارت کرده و آن‌ها را فیلتر کنید. در چنین شرایطی، امکان بلوک کردن درخواست‌هایی به‌وجود می‌آید که نشان‌دهنده حمله تزریق SQL هستند. WAF ها به‌واسطه قابلیت تحلیل درخواست‌ها و پاسخ‌های HTTP، به‌صورت بلادرنگ در تشخیص و جلوگیری از حمله SQL Injection به شما کمک می‌کنند.

۵- حسابرسی امنیتی منظم

حسابرسی امنیتی (Security Audits) و ارزیابی آسیب‌پذیری وب‌اپلیکیشن‌ها باید به‌صورت منظم انجام شوند تا آسیب پذیری SQL Injection پیش از سوءاستفاده، شناسایی و رفع شود. می‌توان از ابزارهای اسکن خودکار و همچنین، ابزارهای بررسی Manual کد به منظور تشخیص آسیب‌پذیری‌های بالقوه و ضعف‌های امنیتی استفاده کرد.

۶- پارامترسازی ورودی

از تکنیک‌های پارامترسازی ورودی مانند استور پروسیجر (Stored Procedure)، کوئری‌های پارامتربندی‌شده و فریمورک‌های ORM به هدف مدیریت ایمن ورودی کاربر استفاده کنید. کوئری‌های پارامتربندی‌شده تضمین می‌کنند که ورودی کاربر تنها به‌عنوان دیتا درنظر گرفته شود و نه کد اجرایی. این امر، خطر آسیب پذیری حمله SQL Injection را کاهش خواهد داد.

۷- مدیریت خطا

مکانیزم‌های مدیریت خطای قدرتمندی را پیاده‌سازی کنید تا از نمایش پیغام‌های خطا با جزئیات زیاد به حمله‌کنندگان جلوگیری شود. ازسوی دیگر، مطمئن شوید که پیام‌های خطایی که به کاربر نمایش داده می‌شوند، اطلاعات حساسی را درمورد ساختار پایگاه داده یا کوئری آشکار نکنند.

۸- آموزش توسعه دهندگان

توسعه‌دهندگان باید درخصوص شیوه‌های کدنویسی ایمن و آسیب پذیری های رایج مانند SQL Injection ، آموزش ببینند و مهارت کافی کسب کنند. منابع آموزشی را در اختیار توسعه‌دهندگان قرار دهید تا خطرات مرتبط با مدیریت نادرست ورودی و اهمیت شیوه‌های کدنویسی ایمن را به‌خوبی درک کنند.

با اجرای این اقدامات پیشگیرانه، سازمان‌ها می‌توان به‌صورت چشم‌گیری از خطر آسیب پذیری های حمله SQL Injection بکاهند و وب‌سایت‌ها و پایگاه داده‌های خود را در مقابل سوءاستفاده‌های مخرب محافظت کنند.

تشخیص آسیب پذیری در برابر حمله SQL Injection

برای تشخیص آسیب پذیری های SQL Injection در هنگام روبه‌رویی با این نوع حملات، ترکیبی از تست، اسکن خودکار و تکنیک‌های مانیتورینگ نیاز است. در این بخش، به بررسی برخی از شیوه های تشخیص آسیب‌پذیری های تزریق اس کیو ال می‌پردازیم.

۱- بررسی کد به صورت دستی

سورس‌کد اپلیکیشن را به‌صورت Manual و کامل بررسی کنید تا آسیب‌پذیری‌های بالقوه حمله SQL Injection شناسایی شوند. توجه کنید که در کدام قسمت، داده‌های کاربر بدون اعتبارسنجی یا پاک‌سازی مناسب، مستقیماً در کوئری‌های SQL قرار گرفته‌اند. به جای استفاده از الحاق رشته (String Concatenation)، کوئری‌های SQL پویایی را جستجو کنید که با استفاده از کوئری‌های پارامتربندی‌شده یا عبارات آماده (Prepared Statements) ساخته شده‌اند.

۲- تجزیه و تحلیل استاتیک خودکار

ابزارهای آنالیز استاتیک اتوماتیک را به‌منظور اسکن کدبیس (Codebase) اپلیکیشن برای الگوها و نشانه‌های شناخته‌شده آسیب‌پذیری های حمله SQL Injection استفاده کنید. ابزارهای آنالیز ایستا این قابلیت را دارند که بدون اجرای اپلیکیشن، آسیب‌پذیری‌های بالقوه، همچون رسیدگی نامناسب به ورودی و ساخت کوئری‌های SQL ناامن را تشخیص دهند.

۳- تست امنیت برنامه پویا (DAST)

با ارسال دیتای ورودی مخرب، به اجرای تست پویایی اپلیکیشن برای شناسایی آسیب‌پذیری‌ها به‌صورت بلادرنگ بپردازید. فریمورک‌های تست نفوذ و اسکنرهای آسیب‌پذیری را به هدف شبیه‌سازی حملات SQL Injection و تشخیص آسیب‌پذیری‌ها در روتین‌های اعتبارسنجی ورودی برنامه به کار ببرید.

۴- تست اعتبارسنجی ورودی (Input Validation Testing)

با ارسال انواع مختلفی از ورودی‌ها، ازجمله کاراکترهای خاص و دستورات SQL، مکانیزم‌های اعتبارسنجی ورودی را بررسی کرده تا بدین طریق، ضعف‌های احتمالی در روتین‌های اعتبارسنجی ورودی شناسایی شوند. علاوه‌براین، رفتارهای غیرمنتظره یا پیام‌های خطایی که ممکن است نشان‌دهنده رسیدگی نامناسب به ورودی کاربر باشد را جستجو کنید.

۵- تست مبتنی بر خطا (Error-Based Testing)

از پیام‌های خطای تولیدشده توسط اپلیکیشن برای تعیین وجود آسیب‌پذیری‌های حمله SQL Injection استفاده کنید.

پاسخ‌های خطا را برای نشانه‌هایی از خطاهای دستور SQL یا Exception های پایگاه داده، تجزیه‌وتحلیل کنید.

۶- تست مبتنی بر زمان (Time-Based Testing)

با وارد کردن تأخیرهای زمانی در داده‌های ورودی، بررسی کنید که آیا زمان پاسخ (Response Time) اپلیکیشن براساس ورودی تزریق‌شده تغییر می‌کند یا خیر. درصورت وجود مغایرت در زمان پاسخ، احتمالاً آسیب‌پذیری بالقوه حمله SQL Injection وجود داشته باشد. در چنین شرایطی، ممکن است فرد حمله‌کننده اطلاعات را به‌صورت مرحله‌ای و افزایشی استخراج کنند.

۷- مانیتورینگ مداوم

شما می‌توانید با پیاده‌سازی روش‌های لاگ‌گیری (Logging) و مانیتورینگ قدرتمند، به شناسایی رفتارهای غیرعادی و فعالیت‌های مشکوکی بپردازید. بدین شیوه می‌توانید مواردی که ممکن است بیان‌گر حملات تزریق SQL باشد را تشخیص دهید. توجه کنید که به‌واسطه مانیتور کردن لاگ‌های پایگاه داده، امکان تشخیص کوئری‌ها یا الگوهای دسترسی غیرعادی فراهم می‌شود.

۸- یکپارچه سازی Threat Intelligence

ازطریق سورس‌های Threat Intelligence، از تهدیدات نوظهور و تکنیک‌های حمله سایبری آگاه باشید. درحقیقت، شما می‌توانید از پایگاه‌های داده Threat Intelligence، به منظور تشخیص شاخص‌های شناخته‌شده حمله SQL Injection و کاهش آسیب‌پذیری‌ها به‌صورت پیشگیرانه بهره‌مند شوید.

جمع بندی: مقابله با حملات SQL Injection

در این مقاله، حمله SQL Injection به‌همراه شیوه‌های جلوگیری از آن بررسی شد. به‌طورکلی، حملات تزریق SQL تهدیداتی جدی برای Web Application ها و پایگاه‌های داده ایجاد می‌کنند و ممکن است به Data Breach، اختلال در سرویس و آسیب‌های مختلفی منجر شود. سازمان‌ها برای به حداقل رساندن این خطرات، ملزم هستند از اقدامات امنیتی پیشگیرانه‌ای مانند اعتبارسنجی ورودی، کوئری‌های پارامتربندی‌شده، اصل حداقل امتیاز، فایروال‌های وب‌اپلیکیشن، حسابرسی امنیتی منظم، مدیریت Patch و سایر موارد استفاده کنند. درحقیقت، سازمان‌ها با اولویت‌دهی به شیوه‌های کدگذاری امن و حساسیت به تهدیدات امنیتی سایبری، می‌توانند از یکپارچگی و محرمانگی داده‌ها محافظت کنند.