چگونگی حمله Injection SQL

حمله SQL Injection چیست؟

حمله SQL Injection یا همان تزریق کد SQL نوعی حمله است که در آن فرد یا گروه مهاجم سعی دارند با ارسال کدهای مختلف برای پایگاه داده راهی را برای نفوذ و یا بدست آوردن اطلاعات و یا حتی تخریب اطلاعاتی که نمی دانند چیست و در کجا قرار دارد، دارند. این نوع از حملات یک نوع خطرناک به شمار می آید و از خانواده حملات Code Injection هستند که همان تزریق کد می باشد. عموما URL هایی که کوئری های مختلفی را درون خود دارند، صفحه های وارد کردن اطلاعات به فرم ها و همینطور صفحه های لاگین از مواردی هستند که می توانند اهداف مناسبی برای اجرای حمله SQL Injection باشد.

اجرای حمله SQL Injection به این صورت است که یک فهرست از دستورات و کوئری های مختلف برای پایگاه داده تهیه شده و از روزنه ای که تشخیص داده شده است و هکر ها فکر می کنند که ممکن است کدهایی که از آنجا وارد می شود روی پایگاه داده پردازش شوند به سایت تزریق می شود. این تزریق ها میتواند به وسیله ابزار های خاصی مانند پک های نرم افزاری موجود روی jSQL Injection ،Kali Linux برای پلتفرم های جاوا و NoSQLMap برای پایگاه داده های NOSQL انجام بگیرد.

اجرای حمله SQL Injection روی پایگاه داده های غیر رابطه ای نیز ممکن می باشد و واژه حمله SQL Injection دقیقا به معنی تزریق کد SQL نیست. بلکه هدف تزریق کوئری هایی است که بتواند از پایگاه داده اطلاعاتی را بدست بیاورد. تشخیص و جلوگیری از این حملات کار ساده ای نیست و معمولا پایگاه داده کوئری ای را که به آن برسد بررسی کرده و آن را اجرا میکند. خیلی از این کوئری ها حتی ممکن است که خروجی خاصی نداشته باشند و فقط به دنبال حذف پایگاه داده و مواردی از این دست باشند.

تاثیر حمله تزریق کد SQL چیست؟

حمله موفقیت آمیز SQL Injection می تواند منجر به دسترسی غیر مجاز به داده های حساس مانند گذرواژه ها، اطلاعات کارت اعتباری یا اطلاعات شخصی کاربر شود.

بسیاری از موارد نقض داده های مشهور در سال های اخیر نتیجه حملات SQL Injection بوده و منجر به صدمه به اعتبار و جریمه های نظارتی شده است.

در بعضی موارد، یک مهاجم می تواند یک پشتیبان مداوم را در سیستم های سازمان بدست آورد و منجر به سازش طولانی مدت شود که می تواند برای مدت طولانی بدون اینکه شناسایی شود کار خود را ادامه دهد.

انواع حمله SQL Injection کدامند؟

طیف گسترده ای از آسیب پذیری ها، حملات و تکنیک های SQL Injection وجود دارد که در شرایط مختلف ایجاد می شوند. برخی از نمونه های معمول تزریق SQL عبارتند از:

- **بازیابی داده های پنهان:** جایی که می توانید یک درخواست SQL را برای بازگشت نتایج اضافی اصلاح کنید.
- **برهم زدن منطق برنامه:** جایی که می توانید یک سوال را برای تغییر در منطق برنامه تغییر دهید.
- **حمله به UNION:** که در آن می توانید داده ها را از جداول پایگاه داده های مختلف بازیابی کنید.
- **بررسی بانک اطلاعاتی:** جایی که می توانید اطلاعات مربوط به نسخه و ساختار بانک اطلاعات را استخراج کنید.
- **تزریق کور SQL:** که در آن نتایج جستجوی شما کنترل شده در پاسخ برنامه نیست.
- **تزریق SQL در قسمت های مختلف پرس و جو**
- **تزریق مرتبه دوم SQL**

1. بازیابی داده های پنهان

یک برنامه خرید را در نظر بگیرید که محصولات را در دسته های مختلف نمایش می دهد. وقتی کاربر بر روی دسته هدایا کلیک کند، مرورگر، URL را درخواست می کند:

“https://insecure-website.com/products?category=Gifts”

این امر باعث می شود تا یک نرم افزار SQL برای بازگرداندن جزئیات مربوط به محصولات از پایگاه داده، از یک SQL Query استفاده کند:

```
SELECT * FROM products WHERE category = 'Gifts' AND released = 1
```

این پرس و جو (SQL query) از database اطلاعات زیر را درخواست می کند:

- (*) all details
- from the products table
- where the category is Gifts
- .and released is 1

“released = 1” برای مخفی کردن محصولاتی که عرضه نمی شوند، استفاده می شود. برای محصولات منتشر نشده “released = 0” استفاده می شود.

این برنامه هیچ گونه دفاعی را در برابر حملات تزریق SQL پیاده سازی نمی کند، بنابراین یک مهاجم می تواند حمله ای مانند:

“–‘https://insecure-website.com/products?category=Gifts”

نتیجه جستجوی SQL به صورت زیر می باشد:

```
SELECT * FROM products WHERE category = 'Gifts'--' AND released = 1
```

پرس و جوی اصلاح شده، تمام مواردی را که در آن دسته از هدایا 1 برابر با 1 است را نشان می دهد، از آنجا که 1 = 1 همیشه درست است، پرس و جو تمام موارد را برمی گرداند.

2.برهم زدن منطق برنامه

برنامه ای را در نظر بگیرید که به کاربران اجازه می دهد با یک نام کاربری و رمز عبور وارد شوند. اگر کاربر نام کاربری و رمز عبور را ارسال کند، برنامه با انجام SQL زیر، اعتبار نامه را بررسی می کند:

```
SELECT * FROM users WHERE username = 'wiener' AND password =
'blue_cheese'
```

اگر پرس و جو، جزئیات یک کاربر را بازگرداند، ورود به سیستم موفقیت آمیز خواهد بود. در غیر این صورت درخواست پذیرفته نمی شود.

در این بخش، یک مهاجم می تواند مانند هر کاربر و بدون رمز ورود به سادگی با استفاده از”SQL comment--” وارد سیستم شود.

بخش چک کردن رمز عبور را از قسمت “WHERE” انجام میدهد و آن را حذف می کند. به عنوان مثال، ارسال نام کاربری ” –‘administrator“و یک رمز عبور خالی به صورت خط زیر:

```
SELECT * FROM users WHERE username = 'administrator'--' AND
password = ''
```

می تواند کاربر را که نام کاربری آن administrator است از دیتابیس دریافت کند و هکر خود را به جای کاربر ثبت کند.

3.حمله به UNION

در مواردی که نتایج یک پرس و جو SQL در پاسخ برنامه ها برگردانده شود، مهاجم می تواند از آسیب پذیری SQL Injection برای بازیابی داده ها از جدول های دیگر در پایگاه داده استفاده کند.

این کار با استفاده از کلمه کلیدی UNION انجام می شود و به شما امکان می دهد "Select query" اضافی را نیز انجام دهید و نتایج را به پرس و جو اصلی اضافه کنید.

به عنوان مثال، اگر برنامه ای درخواست زیر را که حاوی ورودی کاربر "gifts" است، اجرا کند:

```
SELECT name, description FROM products WHERE category = 'Gifts'
```

سپس هکر می تواند ورودی زیر را ارسال کند:

```
UNION SELECT username, password FROM users--
```

این امر باعث می شود برنامه تمام نام های کاربری و کلمات عبور را به همراه نام و توضیحات محصولات بازگرداند.

4.بررسی بانک اطلاعاتی

پس از شناسایی اولیه آسیب پذیری SQL Injection، به طور کلی دستیابی به برخی اطلاعات در مورد خود پایگاه داده مفید است. این اطلاعات اغلب می تواند راه را برای بهره برداری بیشتر هموار کند.

می توانید جزئیات نسخه مربوط به پایگاه داده را پرس و جو کنید. نحوه انجام این کار بستگی به نوع بانک اطلاعاتی دارد، بنابراین می توانید نوع پایگاه داده را از هر تکنیک دیگری بدست آورید. به عنوان مثال، در Oracle می توانید این کار را انجام دهید:

```
SELECT * FROM v$version
```

همچنین می توانید تعیین کنید که جداول بانک اطلاعاتی وجود دارد و کدام ستون ها را شامل می شود. به عنوان مثال، در اکثر بانک های اطلاعاتی می توانید عبارت زیر را برای لیست کردن جدول ها اجرا کنید:

```
SELECT * FROM information_schema.tables
```

5.تزریق کور SQL

بسیاری از موارد تزریق SQL آسیب پذیری کور است. این بدان معناست که برنامه نتایج جستجوی SQL یا جزئیات هرگونه خطای پایگاه داده را در پاسخ های خود باز نمی گرداند.

از آسیب پذیری های کور هنوز هم می توان برای دسترسی به داده های غیر مجاز استفاده کرد، اما تکنیک های موجود عموماً پیچیده تر و دشوار هستند.

بسته به ماهیت آسیب پذیری و پایگاه داده، از تکنیک های زیر می توان برای سوءاستفاده از آسیب پذیری های تزریق کور SQL استفاده کرد:

- شما می توانید منطق پرس و جو را تغییر دهید تا بسته به حقیقت یک شرط واحد، اختلاف قابل تشخیص در پاسخ برنامه ایجاد شود. این ممکن است شامل تزریق یک شرایط جدید به برخی از منطق بولی و یا ایجاد مشروط مانند خطای تقسیم بر صفر باشد.
- شما می توانید به طور مشروط یک تأخیر زمانی در پردازش پرس و جو ایجاد کنید که به شما این امکان را می دهد تا واقعیت شرط را بر اساس زمانی که برنامه پاسخ می دهد استنباط کنید.
- با استفاده از تکنیک های OAST می توانید تعامل خارج از باند شبکه را آغاز کنید. این روش بسیار قدرتمند است و در شرایطی کار می کند که تکنیک های دیگر این کار را انجام ندهند. شما می توانید مستقیماً داده ها را از طریق کانال خارج از باند، مجزا کنید، به عنوان مثال با قرار دادن داده ها در جستجوی DNS برای دامنه ای که شما کنترل می کنید.

تزریق SQL در قسمت های مختلف پرس و جو

بیشتر آسیب پذیری های تزریق SQL در بند "WHERE" از "SELECT Query" بوجود می آیند. این نوع تزریق SQL معمولاً توسط آزمایش کنندگان باتجربه قابل درک است.

اما آسیب پذیری های تزریق SQL در اصل می توانند در هر مکانی از پرس و جو و در انواع مختلف نمایش داده شود. رایج ترین مکان ها که در آن تزریق SQL بوجود می آید یه شرح زیر می باشد:

- در بیانیه های UPDATE، در مقادیر به روز شده یا بند WHERE.
- در عبارت INSERT، درون مقادیر درج شده.
- در عبارت SELECT، در جدول یا نام ستون آمده است.
- در بیانیه های SELECT، در بند ORDER BY.

تزریق مرتبه دوم SQL

تزریق مرتبه اول SQL در جایی بوجود می آید که برنامه از کاربر درخواست HTTP دریافت کرده و در طی پردازش آن، ورودی را در یک پرس و جو SQL به روشی ناامن وارد می کند.

در تزریق SQL مرتبه دوم (که به عنوان تزریق SQL نیز شناخته می شود)، برنامه از درخواست HTTP ورودی کاربر را می گیرد و آن را برای استفاده بعدی ذخیره می کند.

این کار معمولاً با قرار دادن ورودی در یک پایگاه داده انجام می شود، اما در نقطه ای که داده ها ذخیره می شوند، آسیب پذیری ایجاد نمی شود.

بعداً هنگام رسیدگی به درخواست HTTP، برنامه داده های ذخیره شده را بازیابی می کند و آن را به روش نا امن در یک پرس و جو SQL میگنجاند.

تزریق مرتبه دوم SQL اغلب در شرایطی بوجود می آید که توسعه دهندگان از آسیب پذیری های تزریق SQL آگاه هستند و بنابراین با اطمینان می توانید محل اولیه ورود به پایگاه داده را کنترل کنید.

وقتی داده ها بعدا پردازش می شوند، بی خطر تلقی می شوند، زیرا در گذشته با خیال راحت در پایگاه داده قرار گرفته اند. در این مرحله، داده ها به شیوه ای نا امن اداره می شوند، زیرا توسعه دهنده فکر می کند که این موضوع قابل اعتماد است.

نحوه تشخیص آسیب پذیری های تزریق SQL

اکثر آسیب پذیری های تزریق SQL را می توان به سرعت و با اطمینان با استفاده از اسکنر آسیب پذیری وب "Burp Suite" یافت.

SQL Injection می تواند به صورت دستی و با استفاده از یک مجموعه سیستماتیک از تست در برابر هر نقطه ورود به برنامه تشخیص داده شود. به طور معمول شامل موارد زیر است:

- ارسال شخصیت نقل قول واحد و به دنبال خطا یا ناهنجاری های دیگر.
- ارسال برخی از نحوه های خاص SQL که به مقدار پایه (اصلی) از نقطه ورود و به یک مقدار دیگر ارزشیابی می کنند و به دنبال تفاوت های سیستماتیک در پاسخ های برنامه کاربردی هستند.
- ارسال شرایط بولی مانند 1 = 1 و 1 = 2 و جستجوی تفاوت در پاسخ برنامه.
- ارسال بار های با هدف ایجاد تاخیر در هنگام اجرای درخواست در SQL و به دنبال تفاوت در زمان لازم برای پاسخگویی هستید.
- ارسال بار های OAST با هدف ایجاد تعامل شبکه خارج از باند هنگام اجرای یک پرس و جو SQL و نظارت بر هرگونه تعامل حاصل از آن، طراحی شده است.

روش جلوگیری از حمله SQL Injection چیست

در مورد اینکه حمله SQL Injection را چگونه باید خنثی کنیم باید بگوییم که میتوان روش های این کار را به دو صورت تعیین نمود. در اولین اقدام شما باید مطمئن باشید که سایت شما انقدر امن می باشد که صرفا کوئری های صحیح را به پایگاه داده ارسال کند و بتواند در مقابل موارد ناخواسته مقاومت کند. به طور کلی به اینصورت باشد که درز و سوراخی برای تزریق روی آن وجود نداشته باشد. در مورد دوم نیز پایگاه داده شما

باید کمی هوشمند باشد و بتواند تشخیص دهد که چه دستوری را نباید اجرا کند و این هم باز بستگی به تنظیمات و طراحی ای است که شما روی آن انجام می دهید.

به طور کلی می توان روش های جلوگیری از حمله SQL Injection را به این صورت معرفی کرد:

نصب پچ ها و آپدیت های امنیتی

وقتی که برای شما پیامی مبنی بر اینکه افزونه های شما و یا سیستم مدیریت محتوای شما نیاز به آپدیت دارد به دست شما میرسد بهتر است تعلل نکنید. زیرا خیلی از باگ ها که به صورت ناگهانی پیدا می شوند روی خیلی از سایت ها وجود داشته و این حملات جدید در ابعاد وسیعی اتفاق می افتد که دردسر ساز خواهد بود. در صورتی که پچ ها و آپدیت ها را به موقع نصب کنید می توانید امنیت سایت خود را هم در مقابل حمله SQL Injection و هم هر حمله دیگری بالا ببرید.

کنترل ورودی ها

مورد کاربردی دیگری که باید به سراغ آن برروید این است که ورودیهای خود را کنترل کنید و مطمئن باشید امکان ارسال کوئری های از پیش تعیین نشده و دور از انتظار در آنها وجود ندارد. در صورتی که از سایت های شخصی سازی شده و یک CMS اختصاصی استفاده می کنید باید توجه داشته باشید که این کار را حتما در اولویت قرار دهید. اگر این سوال برای شما وجود دارد که چگونه باید این مورد رفع شود باید بگوییم که در اینجا خلاقیت برنامه نویس در میان است و برنامه نویس شما باید از روش هایی که بلد است استفاده کند و یا روش های جدیدی را یاد بگیرد.

عموما با کمی بازی با کد ها می توان از این مشکلات جلوگیری کرد و کار زیاد سختی در پیش نخواهد بود. اما در نظر داشته باشید همین کار ساده بسیار حیاتی و مهم است.

ایجاد سطوح دسترسی روی پایگاه داده

یکی از اصلی ترین کارهایی که باید انجام شود این است که با استفاده از پرمیشن ها و سطوح دسترسی پایگاه داده خود را محدود کنید. در این حالت تنها کسانی که دسترسی های بالا داشته باشند میتوانند عملیات های مهم مانند مشاهده جداول خاص و یا حذف یک جدول از روی پایگاه داده را انجام دهند. و تقریبا می توان گفت با اگر حمله SQL Injection انجام شود و کوئری های مخرب به پایگاه داده ها هم برسد مجوز انجام اعمالی که میخواهد صادر نمی شود.

باید توجه داشته باشید که تایید اکانت ها و احراز هویت در این پایگاه داده ها باید به شدت قوی باشد وگرنه این کار می تواند بی فایده باشد.

تنظیم امکانات پایگاه داده برای برخورد

یکی دیگر از مواردی که باید در نظر داشته باشید برخورد های پایگاه داده با کوئری های نامشخص می باشد که می تواند بسیار کمک کننده باشد. گاهی اوقات پایگاه داده های شما می توانند در صورت مشاهده کوئری های

عجیب و غریب که از مکان های نامشخص رسیده اند واکنش هایی را نشان دهند. به اینصورت که علاوه بر اجرا نشدن آن کاربر مورد نظر می تواند به عنوان یک کاربر مشکوک و مخرب بن شده و دسترسی آن به سایت قطع شود. به کارگیری این مکانیزم ها میتواند بسیار کمک کننده باشد.

دقت کردن به ارور ها

یکی دیگر از موارد که می تواند به شما بگوید که یک حمله SQL Injection روی سایت شما در حال اجراست و باید فکری به حال آن بکنید، ارور هایی است که پایگاه داده شما به شما باز می گرداند. در موراد زیادی پایگاه داده متوجه دستورات بی معنی ای که به آن می رسد شده و خطاهایی را مبنی بر اطلاعات و دستورات ناقص باز می گرداند. اگر به این خطاها بی توجه باشید ممکن است حمله SQL Injection بالاخره به ثمر برسد و پایگاه داده شما را منهدم کند!

اما اگر به پیغام ها و ارور هایی که پایگاه داده به شما میدهد توجه کنید می توانید قبل از اینکه این اتفاق بیفتد آن را شناسایی کرده و دسترسی حمله کننده یا حمله کنندگان را مسدود کنید.

به طور کلی در مورد روش های جلوگیری از حمله SQL Injection می توانیم بگوییم که این نوع از حمله حمله ای است که بر پایه خلاقیت و پیدا کردن نقاط ضعف انجام می شود و برای اینکه بتوانید آنها را برطرف کنید باید شما هم خلاق باشید و از سیستم هایی استفاده کنید که به هیچ عنوان راه ورود و تزریقی باقی نمی گذارد.

چرا فایروال های عام منظوره (مانند فایروال ویندوز) نمی توانند جلوی چنین حمله ای را بگیرند. با جستجو در اینترنت، گونه هایی از فایروال را توضیح دهید که می توانند مانع چنین حمالاتی شوند.

هدف اصلی هر فایروال نظارت و مسدود کردن درخواست های نامعتبر است. WAF یک فایروال تخصصی برای وب سایت ها و برنامه های وب است که از آنها در برابر درخواست های مخرب خارجی به سرور وب محافظت می کند. در همین حال ، فایروال های شبکه داده های منتقل شده بین دو یا چند سرور وب را ایمن می کنند.

چند نمونه از Web App Firewall ها:

1. Cloudflare WAF
2. Sucuri Website Firewall
3. AppTrana
4. AWS WAF
5. Akamai WAF

تهیه کننده: علی حبیبیان صدیق

منابع:

https://www.maralhost.com/kb/what-is-sql-injection-and-how-we-ca-scape/?__cf_chl_managed_tk__=pmd_gdsN1UAFxzRvYFtkoAmk16wrp4oCcPeYtQslwQzTbd8-1632222403-0-gqNtZGzNAuWjcnBszRNl#hmlh_SQL_Injection_chyst

https://momtazserver.com/%D8%AD%D9%85%D9%84%D9%87-%D8%AA%D8%B2%D8%B1%DB%8C%D9%82-%DA%A9%D8%AF-sql-injection/

https://www.softwaretestinghelp.com/web-application-firewall-waf/