Digital Blind Spots: A Field Study of Common Website Insecurities in Small Businesses

Lucas Zhang, Zhi Qu, Andrew Ma, Russell Xue, Nicholas Peng, Alan Kang

lucas.zhang@youthcyberdefender.org

Abstract

Small businesses are increasingly targeted by cyberattacks, yet they often lack the resources to invest in professional security assessments, leaving them vulnerable to common yet critical threats. This paper presents findings from student-led security evaluations guided by the Educational Cybersecurity Assessment Framework for Small Businesses (ECAF), a streamlined process for conducting authorized, community-driven assessments. This paper reveals four prevalent and high-impact vulnerabilities uncovered in real-world assessments, including uncensored file uploads, improper authorization, insecure data at rest, and SQL injections, and provides actionable remediation strategies. These case studies illustrate the critical yet often overlooked security challenges small businesses face. This paper will help developers to quickly identify common vulnerabilities on small business websites, improve application quality by fixing these issues before being exploited by cyber attackers.

Biography

The authors are high school students from the Portland Metro Area and members of Youth Cyber Defender, a 501(c)(3) non-profit organization dedicated to youth cybersecurity education and service. They all have cybersecurity training for 3-5 years and have participated in cybersecurity competitions such as CyberPatriot and picoCTF. In the past two years, they focused their efforts on helping secure local businesses from cyberattacks.

1 Introduction

The student authors of this paper have developed their interests in cybersecurity since 2021. During one of the early training sessions, a small business owner concerned about his online presence approached us and wondered if we could help him secure his environment. This sparked a mission for us—to help small businesses vulnerable to cyberattacks with accessible testing while educating the next generation of cybersecurity professionals. Since then, Youth Cyber Defender and its members have conducted multiple cybersecurity assessment tests on small businesses, through which we have found common, easily exploitable, and critical vulnerabilities across the websites of small businesses.

This paper is organized as follows: firstly, this paper will overview of the background and motivations for our work, the framework with which we followed, and then discuss examples of four categories of common vulnerabilities: abusable file upload, improper authorization, insecurities of data at rest, and risk of SQL injection. Finally, the paper ends with a conclusion and future work.

2 Background and Motivation

Nowadays, cybersecurity attacks are commonplace in news headlines. Most attacks in the news are associated with large companies or government agencies. It is daunting to read about millions of users' data compromised from a data breach or millions of dollars lost from ransomware. Every year, large companies and the government increase their budget and hire cybersecurity professionals to improve their cybersecurity defenses.

However, the cybersecurity risks associated with small businesses are dangerously overlooked or ignored. Small businesses employ nearly half of the American workforce and represent 43.5% of America's GDP, according to the U.S. Chamber of Commerce. They are the backbone of local economic ecosystems. Due to the constraints of resources, small business owners often hire inexperienced developers or support staff to build their websites with open-source software, default out-of-the-box configurations, insufficient validation testing, and a lack of security integrated into the design.

A 2023 Business Impact Report: Small Businesses and Cyberattacks from Tripwire found 73% of small business owners and leaders reported experiencing data breaches or cyberattacks. Studies conducted by other organizations report other alarming facts:

- Nearly 43% of cyberattacks are targeted at small businesses (SMB).
- Only 14% of these SMBs are prepared to face such an attack.
- On average, SMBs spend between $826 and $653,587 on cybersecurity incidents.

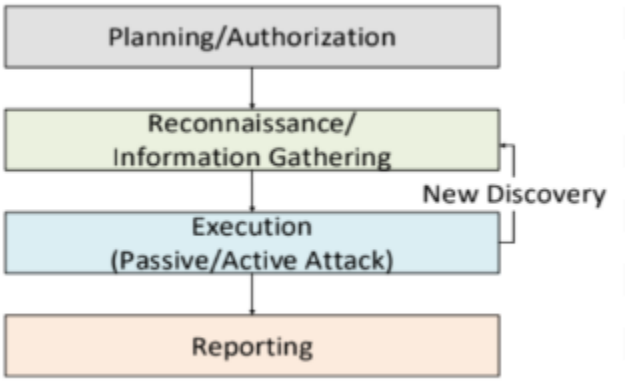
To enhance systems' security, the best practices in the cybersecurity industry call for a security assessment or penetration testing with the purpose of discovering potential security vulnerabilities that could be exploited by hackers. Both tests simulate a cyberattack designed to evaluate the security of a system, network, or application. It involves authorized attempts to identify vulnerabilities and exploit weaknesses, mimicking the tactics of malicious attackers, but without causing harm. However, the cost for this type of professional service, ranging from $5,000 to hundreds of thousands of dollars, is beyond the budget for most small businesses.

Students from Youth Cyber Defender witness this urgent need from local small businesses. We also see this as a unique opportunity to make a real impact on our community by helping business owners avoid potential financial disasters from cyberattacks. Equipped with the skills from years of cybersecurity training and supervised by their mentor-coaches, students mobilized themselves, reached out to local businesses, performed security assessments under authorization, discovered critical security vulnerabilities that could lead to total compromise of the business websites, and helped owners to remediate the problems.

Before our assessments, we leveraged local connections with small businesses and our previous experience in cybersecurity to obtain authorization to conduct testing. In addition, the small businesses' awareness of the dangers of cyberattacks, combined with the lack of affordable cybersecurity evaluations, acted as a large factor in our receiving authorization. Following our initial assessments, we presented the methodology we used at cybersecurity conferences, leading us to connect with cybersecurity professionals and other small businesses outside of the wider Portland area and the Pacific Northwest.

3 Methodology and Framework

All our assessments follow the Educational Cybersecurity Assessment Framework for Small Businesses (ECAF) – a framework developed by Ethan Zhang. This framework is designed to enable students to provide cybersecurity assessment services to small businesses.



ECAF consists of four stages:

- Authorization: The assessment team receives authorization from business owners for the assessment scope, timeline, roles, and responsibilities. As many small businesses recognize the need for protecting their cyber-presence but lack the means, our affordable services are a large selling point.
- Reconnaissance: In the reconnaissance phase, the assessment team gathers site or user information from the Internet before exploiting or validating any vulnerabilities. This phase is critical for understanding the target's environment, identifying potential vulnerabilities, and planning the attack strategy.
- Execution: After analyzing the results from reconnaissance, the assessment team lays out a plan to exploit or validate the vulnerabilities and execute the plan. Some results from execution may lead to further reconnaissance.
- Reporting: All the findings from the reconnaissance and execution phases will be documented in a final report that will be delivered to the business owners. The report will contain details of steps that can be used to reproduce the problems. It also includes recommendations for remediation solutions.

In the past two years, students from YCD have performed multiple assessments for local small businesses. We found common security issues on SMB websites that led to the compromise of data confidentiality and integrity. By getting familiar with common methodologies, developers can quickly discover site vulnerabilities and greatly improve site security.

4 Case Studies

In this section, we will overview the most common and critical security issues discovered from our assessment exercises. All the cases studied here are from real websites with the identifiable information redacted for privacy purposes.

4.1 Uncensored File Upload

Many websites offer a file upload feature for users. Generally intended for form, text, or photo file uploads, this feature, without limitations on file types and contents, can be exploited to run a shell and execute system commands. During the assessments, we were able to access confidential files uploaded by other users, including sensitive private information in signed forms with signatures. This vulnerability creates an entry point that allows a malicious user to inject code and other strategies to control the website.

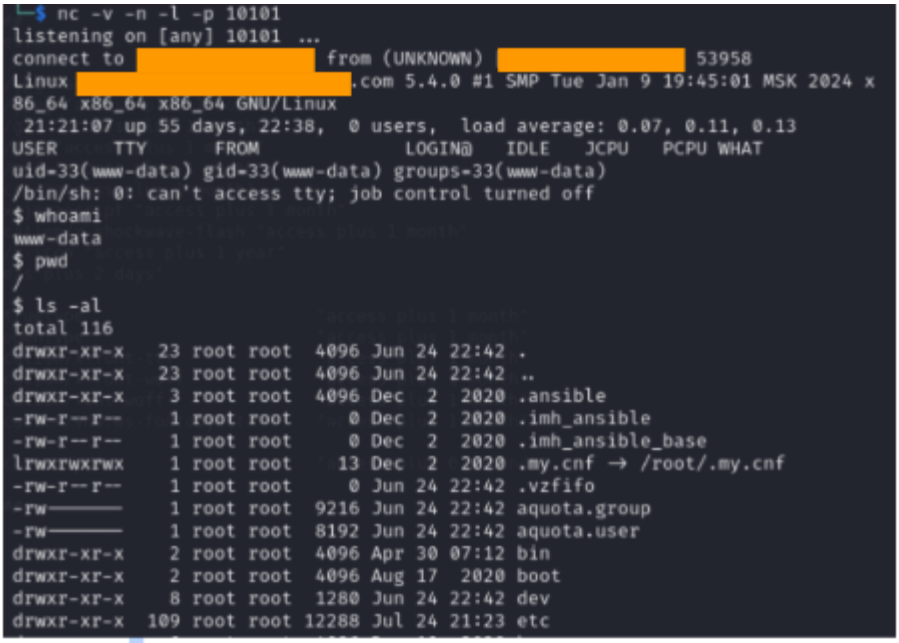
4.1.1 Validation of Uncensored File Upload

PHP is a very common programming language used in website development. One website we assessed has a file upload feature for its users to upload the health and immunization forms.

The website failed to check which files were allowed for upload, so we uploaded a PHP file with a simple one-line PHP web shell code:

<?=`$_GET[0]`?>

The website executed the code after this file was uploaded, allowing us to run Linux commands in the system through web browser, such as listing all other uploaded files. We exploited this vulnerability further by uploading a reverse shell PHP exploit code, which can be found on the Internet. With interactive system shell, we were able to compromise their hosting servers. To ensure malicious actors could not access data retrieved by us, we did not keep copies of the sensitive data itself.



4.1.2 Recommendation

While removing the feature completely is the safest option, the business may need to have the feature for its business function. One way to reduce the harm would be to restrict the type of file uploaded to specific formats, such as PDF, JPG, PNG, etc. Implementing this guardrail will reduce the risk, thus preventing an easy entry point.

4.2 Improper Authorization

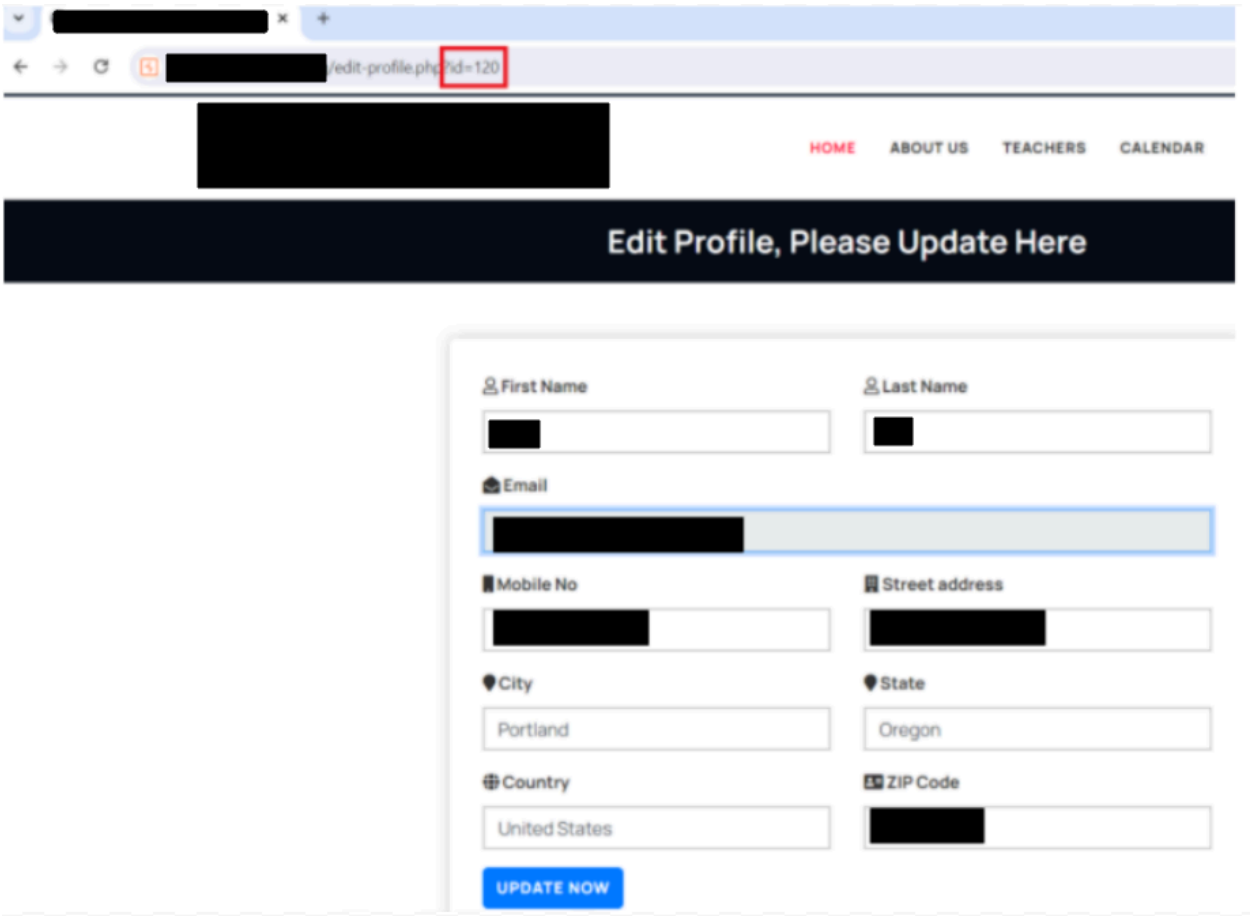
Improper authorization is a vulnerability that a system fails to check the user's permission on the requested data, so it grants the user access to unauthorized information or functions. Improper authorization may lead to privilege escalation that allows unauthorized modifications to the system. We have witnessed multiple cases of improper authorization through our assessments.

On one website that provides education services for students, we explored the improper authorization and were able to view and modify any user's information and profile. This was achieved by simply modifying the ID field as part of the HTTP GET request.

4.2.1 Validation of Improper Authorization

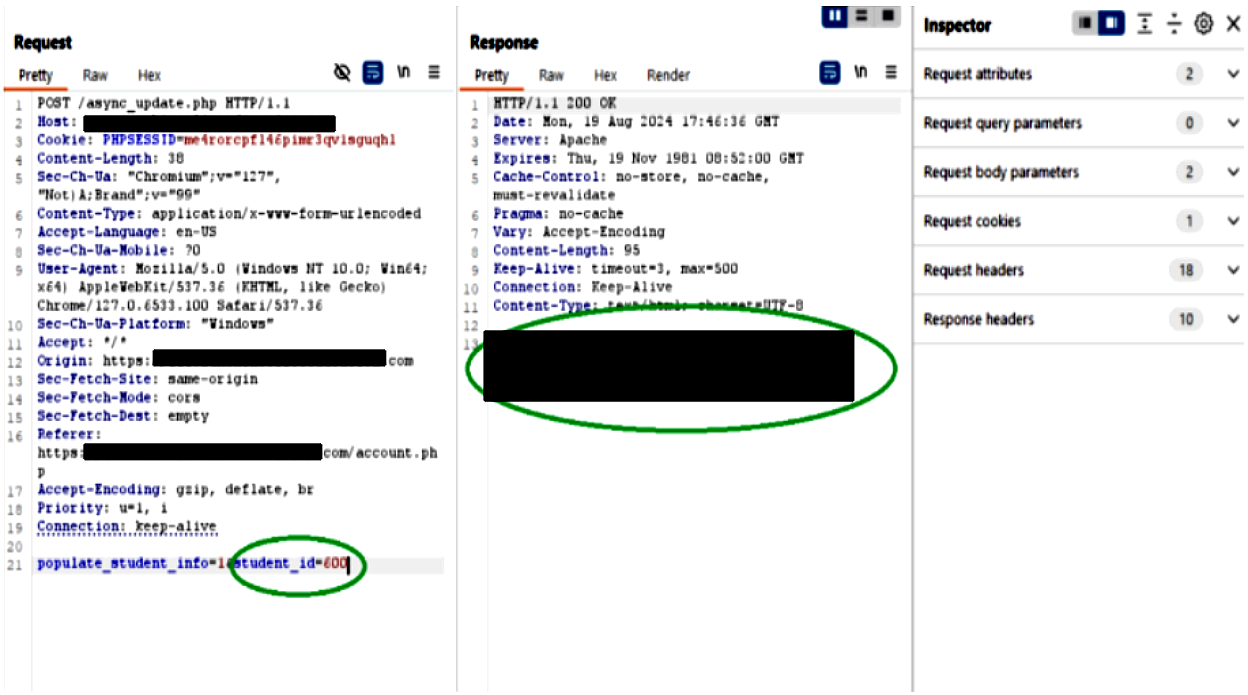
One of the easiest ways to discover improper authorization is by changing URL parameters. Many times, the data used to query the backend database is embedded as an HTTP GET parameter and displayed on the user's browser. One common function of most websites with authenticated users is to show the user's own profile information. By changing this request to another user's identity, the application may display unexpected information.

The picture shown below is a user profile edit function that is designed for users to modify their own profiles. However, by simply changing the id field in the HTTP GET request, we were able to modify another user's profile—a clear sign of improper authorization.

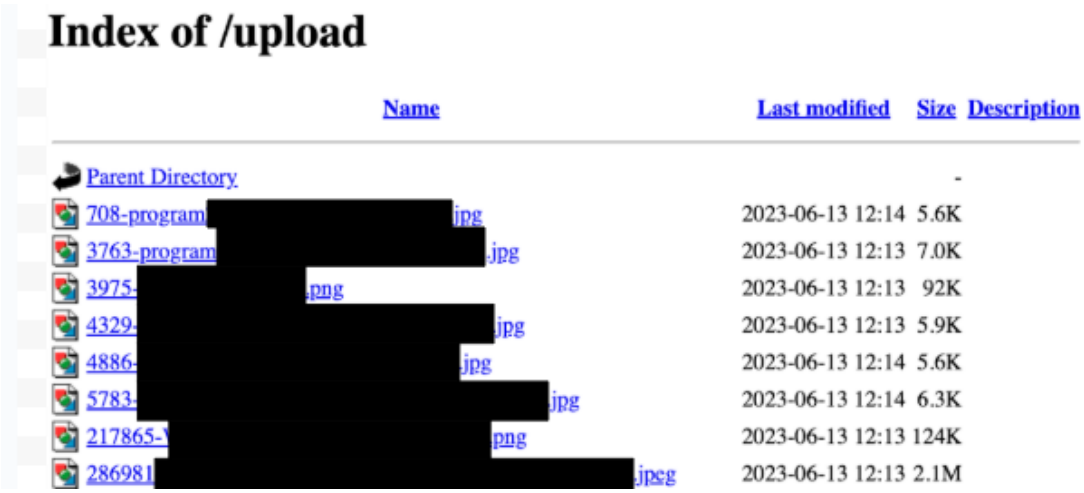


An assessment with another website discovered the same improper authorization issue with even more severe consequences. We were able to delete all the user accounts from the system as a regular user. This involved using a tool called Burp Suite, which can capture all the network traffic between local browsers and remote web servers. We were able to modify the HTTP request to delete any user we wanted.

The picture below shows that the delete function relies on an id parameter sent through HTTP POST request which means users will not be able to modify its value directly on their browser. By using Burp Suite, we were able to modify the id parameter and delete any user in the system.



Another example of improper authorization is closely associated with the file upload function. Files uploaded by other users are available to anyone on the Internet. Ideally, files should only be accessible to the user who uploaded them.



4.2.2 Recommendation

There are multiple steps we can take to protect systems from this attack. Applications should not trust any data from the user's browser, and all data from the browser should be inspected for its legitimacy before being processed by the backend application. Furthermore, it is recommended to enforce correct authorization controls and checks, such as role-based (RBAC), attribute-based (ABAC), or policy-based access control.

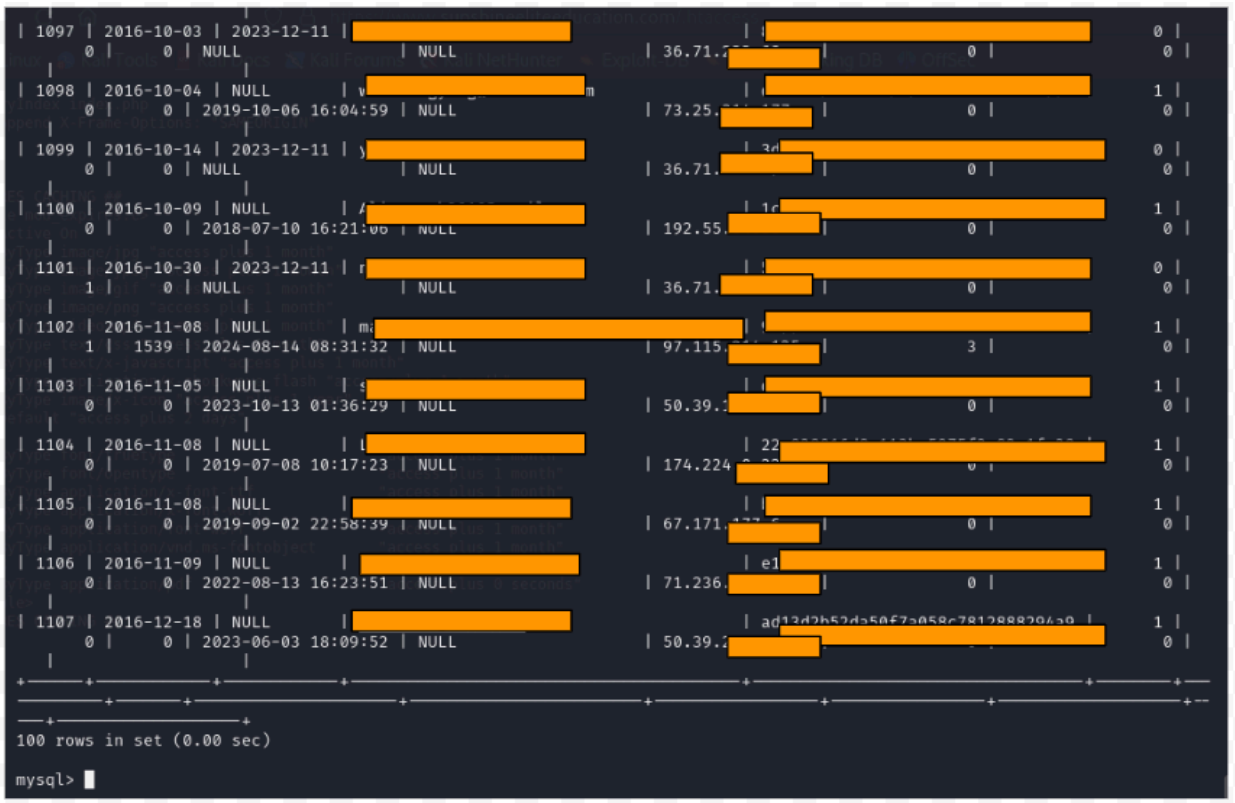
4.3 Data at Rest

Throughout our assessments, we identified several critical and easily exploitable vulnerabilities in the websites and digital infrastructure of small websites, which can be classified as ‘data at rest’ issues, with many related to the use of MySQL databases. ‘Data at rest’—as implied by the name—refers to data at a state of rest, rather than when it is being transmitted or used in any manner. MySQL is a commonly used open-source database that utilizes SQL (Structured Query Language) to manage databases. After accessing MySQL databases, we discovered that many passwords and other sensitive pieces of information were not encrypted properly. Two scenarios we encountered were the lack of any encryption and the encryption of passwords in weak hash formats, such as the use of MD5.

4.3.1 Passwords at Rest

Though we find it is very common that plaintext data is stored in a MySQL database, exploiting this issue does require direct access to file systems. It is more common for an attacker to exploit SQL Injection vulnerability, or access the database through the MySQL command line. Once a hacker gains access to the database, the first target normally is the user table with all the passwords. Here we see two common missteps: the first is that passwords are stored in plaintext, and the second is that passwords are stored in the vulnerable MD5 format. We dumped more than 7000 passwords in MD5 format and were able to crack many of them.

The picture below shows the content of a user table with MD5-hashed passwords.



4.3.2 Recommendation

MySQL version 8.0 or later has native support for database encryption. Developers should leverage this capability to encrypt the database at rest. More importantly, the password data field should be stored in a more robust hash format such as SHA256 or SHA512. Validation can be easily achieved by checking the password field through the MySQL command line.

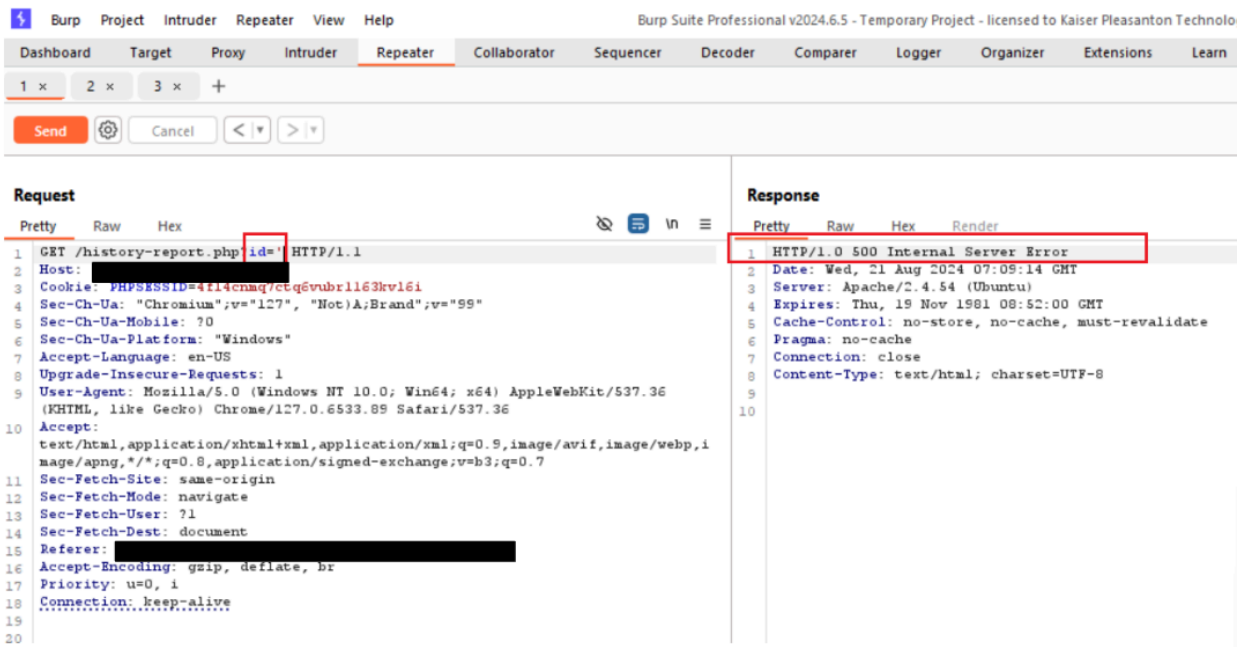
4.4 SQL Injection

SQL Injection (SQLi), or Structured Query Language Injection, is a technique used by attackers to gain unauthorized access to data by inserting malicious SQL code into input fields—such as login forms—on a website. Modern websites often contain numerous input fields, which can make them vulnerable targets. By exploiting poorly secured inputs, an attacker can manipulate SQL queries to bypass authentication mechanisms, potentially gaining access without a valid password. This can allow them to interact directly with the backend database, extract sensitive information, or even alter or delete data.

4.4.1 Discovery of SQL Injection

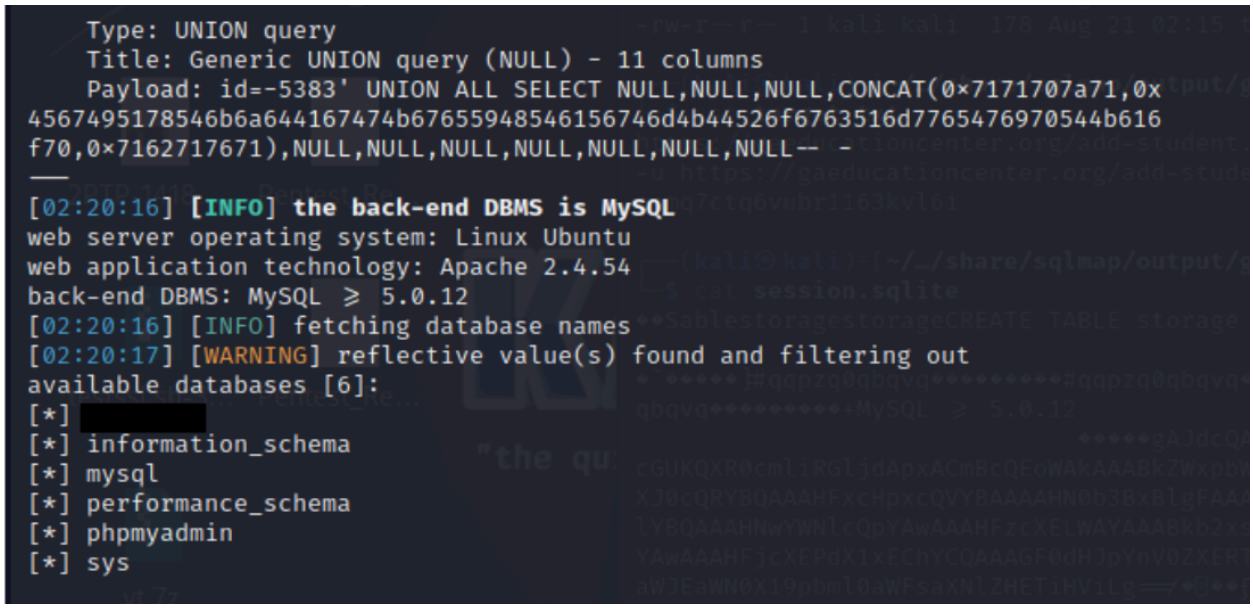
The first step to exploiting SQL Injection is to identify its existence. By using open-source tools like BurpSuite and SQLmap, which can discover data input points that are vulnerable to SQL Injection, we were able to retrieve names, Date of Birth(DOB), emails, and passwords.

In the screenshot below, we can see Burp Suite being used to identify vulnerable input fields by entering special characters, such as the single quotation mark, to cause system errors.



Once the SQL Injection vulnerability is confirmed, we can use SQLmap to run SQL queries to interact with the backend database.

In the screenshots below, you can see that we used SQLmap to gain direct access to the website's databases.



4.4.2 Recommendation

The most effective defense against SQL Injection is to use prepared statements or parameterized queries. This can prevent SQL code from being interpreted as part of the query structure. There are many examples on the Internet for different programming languages.

Another effective way is to validate and sanitize user inputs, reject unexpected formats or characters from users.

5 Conclusion

Due to resource constraints, small business applications focus more on quick delivery of business functions, with security features often being overlooked. Though there are many potential security vulnerabilities, the security vulnerabilities discussed in this paper are commonly found in our assessment exercises. These four issues are typically associated with high business risks that may lead to total compromise of the site and can be easily avoided if developers are familiar with the technology of discovering and remediating these issues.

In the future, we aim to involve more young students in cybersecurity training and provide more cybersecurity services to local businesses. Additionally, we plan to expand our cybersecurity service beyond web applications to include mobile applications that are becoming more and more popular.

Acknowledgements

The authors would like to thank team captain Ethan Zhang for his leadership during cybersecurity training and assessments. The authors would also like to thank the team coach, Yongtian Zhang, for his excellent mentorship throughout our cybersecurity journey. Finally, the authors would like to thank the local small business owners assessed for trusting us to help them with cybersecurity matters.

References

U.S. Chamber of Commerce, “Small Business Data Center”, https://www.uschamber.com/small-business/small-business-data-center (accessed June 7, 2025)

Tripwire, December 27, 2023, “2023 Business Impact Report: Small Businesses and Cyberattacks”, https://www.tripwire.com/state-of-security/business-impact-report-small-businesses-and-cyberattacks (accessed June 7, 2025)

OWASP Top 10, https://owasp.org/www-project-top-ten/ (accessed June 2024)

Ethan Zhang, April 2025, “ECAF: Educational Cybersecurity Assessment Framework for Small Businesses”, In book: Foundations of Computer Science and Frontiers in Education: Computer Science and Computer Engineering (pp.123-136), Springer

Simple PHP Web Shell, https://github.com/bayufedra/Tiny-PHP-Webshell/blob/master/README.md (accessed June 2024)

Fast101, Github, “PHP Reverse Shell”, https://github.com/flast101/reverse-shell-cheatsheet/blob/master/php-reverse-shell.php (accessed July 2024)

Burp Suite, https://portswigger.net/burp (accessed July 2024)

SQL Injection, https://owasp.org/www-community/attacks/SQL_Injection (accessed June 2024)

SQLmap, https://github.com/sqlmapproject/sqlmap/wiki/usage (accessed July 2024)