Proc. of the International Conference on Computing and Advances in Information Technology (ICCAIT 2023)
21-23 November 2023, Ahmadu Bello University, Zaria, Nigeria

Web Application Security and Detecting Security Vulnerabilities

[1]A. A. Sambo, [1]M. A Sani, [1]M. Lawal
[1]*Department of Computer Science,*
Ahmadu Bello University,
Zaria – Nigeria.

***Abstract*—This paper presents a case study focusing on the exploitation of web application vulnerabilities and their solutions within the ABU domain. The project aimed to develop a system capable of encoding and decoding Base64 text through a user-friendly interface to facilitate efficient communication between users and the system. Furthermore, the study explores practical exploitation techniques using Kali Linux as the operating system and Burp Suite as a proxy tool, identifying seven vulnerabilities based on OWASP standards. These include two instances of brute force attacks on the student and staff domains, resulting in unauthorized access to user accounts; three cases of Insecure Direct Object References (IDOR) on the student and staff domains, leading to unauthorized data viewing and editing; one Cross-Site Scripting (XSS) vulnerability on the student Moodle domain, enabling arbitrary JavaScript code execution in users' browsers; and one HTML injection vulnerability on the student Moodle domain, allowing arbitrary HTML code execution that altered the website's appearance. The paper proposes solutions such as implementing reCAPTCHA to prevent brute force attacks and emphasizes the importance of robust authorization checks, input validation, and data sanitization.**

Keyword: ***Web Application Technology, Web Application Vulnerability, Web Application Security, Base64***

I. INTRODUCTION

Web application play a crucial role in our today's life as they enable individuals, institutions and organizations perform various tasks and activities online. It has become an essential tool for educational institutions as it provides a central platform for communication, course management, and personal information management for student and staff [8]. web applications can be defined as computer programs that run inside a web browser, where the code to be executed is fetched from a remote location. Web application security is the process of protecting websites and online services against different security threats that exploit vulnerabilities in an application's code [7]. The increasing dependence on technology and internet has resulted in a corresponding raise in number of web applications being developed and deployed. Web applications are vulnerable to a variety of security threats and attacks which can compromise the confidentiality, integrity and availability of sensitive information. There are various approaches to increase the web applications security such as validation of input and output data, avoiding storing data that you do not need on the website and its database.

The open web application security project (OWASP) provides the top ten projects which is a list of 10 most dangerous current web application security flows, along with effective methods of dealing with those flaws.

OWASP is an organization that provides unbiased and practical, cost-effective information about computer and internet applications. Paper members include a variety of security experts from around the world who share their knowledge of vulnerabilities, threats, attacks and countermeasures. It is an open community dedicated to enabling organizations to develop, purchase, and maintain applications that can be trusted. The widespread use of web applications has made them an attractive target for attackers seeking to compromise sensitive information or disrupt services. Despite advancements in web security, web application vulnerabilities continue to pose a significant risk to organizations and individuals. Many organizations and individuals are not fully aware of the types of vulnerabilities that exist or the measures they can take to prevent them. This paper will mainly focus on OWASP top ten risk, and its countermeasures released in 2010.

II. RELATED WORK

Mohammad et al. [2] OWASP introduced a risk assessment framework to calculate the risk. Based on risk assessment framework, 135 websites and web applications are scanned for security issues and vulnerabilities. The result shows that the amount of high risk and medium risk vulnerabilities are seriously more considerable in Afghanistan.

Daljit & Kaurb [3] includes the web attacks analysis from Website Hacking Incident Database (WHID) and other information security and news websites. Also, it is an effort to analyze various attacks on major categories of web sites which is a guide to developers to take respective appropriate preventive measures in future. The top web attacks have been identified and also the top vulnerable categories of web applications are analyzed.

Mohammad et al. [1] propose an efficient approach for assessing vulnerability that is based on the Python programming language. Which can used to conduct Vulnerability Assessment on web applications? This work will be useful for organizations and programmers to keep their information and applications more secure and viable for usage in sensitive environments.

Hasty et al. [5] present a survey on web application security aspects includes critical vulnerabilities, hacking tools and also approaches to improve web application and websites security level.

Sandeep et al. [4] described the different aspects of web security and its weakness. The main elements of web security techniques such as the passwords, encryption, authentication and integrity are also discussed in this paper.

The anatomy of a web application attack and the attack techniques are also covered in detail. This paper explores a number of methods for combatting this class of threats and assesses why they have not proven more successful. This paper proposes a better way for minimizing these types of web vulnerabilities. It also provides the best security mechanisms for the said attacks.

Murat et al. [6] provides an overview of web application security testing with different focused heading. These heading cover contribution types, web security testing tools and their sub features, specific questions/features to the security testing such as vulnerability types, system under testing (SUT) focused heading and more. We review and structure the body of knowledge related to web application security testing in the form of a systematic literature mapping (SLM.). As part of this study, we pose four sets of research question, define selection and exclusion criteria, and systematically develop and refine a classification schema.

III. Analysis And Design

This section talks about the system analysis, requirement analysis, methodology in addressing the ideas and the tools used to accomplish the paper ideas.

A. Requirements Analysis

Through the years, web applications launched by institutions provide automated activities base on category of the user. This web application developed using HTML, CSS, javaScript, PHP, and MySQL, it is hosted on an Apache/2.4.41 (Ubuntu) server. The system adopts a multi-tiered architecture to deliver dynamic and interactive web services. It's front-end, powered by HTML, CSS, and javaScript, offers an engaging user interface and responsive design. PHP manages server-side logic and serves as the bridge to the MySQL database, ensuring data management and processing. The inclusion of PHPMyAdmin simplifies database administration task.

B. Virtual Box

Virtual box commonly referred to VirtualBox, is open-source virtualization software that allows you to run multiple operating systems on a single physical computer. It enables the creation of virtual machines (VMs) or virtualized environments where you can install and run different operating systems, such as Windows, Linux, MacOs and more, simultaneously on a single host machine.

C. Kali Linux

Kali Linux is a specialized open-source operating system designed for cyber security professionals, ethical hackers, penetration testers. It is a powerful operating system that comes with pre-installed hacking tools.

D. Web Browser

A web browser is a software application that allows users to access and navigate the World Wide Web, which is a vast network of interconnected websites and web pages on the internet. Web browsers interpret and display the content of web pages, enabling users to interact with websites, view text, and images, videos and other multimedia elements, and access online services.

E. Burp Suite

Burp suite is a JAVA based tool used by security professionals, hackers, and penetration testers to access the security of web applications. It is developed by portswigger and is specifically designed for performing various security testing task on web applications and is serves as a proxy server between the user's browser and the targeted web application. This allows the intercept, inspect, and modify web traffic between the client and server. The following features are used during the study:

1. *Repeater*

Repeater allows users to manually manipulate and send HTTP request to a web application. This is valuable for testing specific parameters, payloads, or interactions to discover vulnerabilities that might not be found through automated scans.

2. *Intruder*

Intruder is used for automating customized attacks, such as brute force or dictionary attacks on login pages.

IV. Vulnerability Assessment

Vulnerability assessment is a process used to identify, evaluate, and prioritize potential weakness in various aspect of an organization's digital environment. The approach adopted in this project is assessing vulnerabilities using manual techniques.

A. Brute-force attack

A brute force attack is a cyber-attack method in which an attacker tries all possible combinations of passwords, encryption keys, or other access credentials until the correct one is found.

Fig. 1. Key steps of brute force attack

B. Brute-force attack on staff domain

Authentication provides a way for subject to be identified. An authentication is successful (authenticated) if the subject can proof it is what it is claimed to be otherwise authentication failed. Ahmadu Bello University uses the concept of single factor authentication whereby subject are identified through knowledge factor. Each user has his username and password that is only supposed to be known by him. In this case ABU provide username and password for authentication.

After filling the authentication requirement with ******* as username and ********** passwords, click on login. Then you go to burp suite and click on **proxy**, under proxy click on **HTTP history**, HTTP history where all the requests sent to the server are recorded, look for **post staff/login** request, right click and click on **Send to Intruder**.

In this attack we are only testing for password of the above username which we only need to set one payload position that will be testing different passwords until correct match is found. Click on **Intruder** and set the attack type to **Sinper** which support only one payload position, highlight the password and click on **Add $** to set the payload position.

Click on payload tab and change the payload type to **Brute Forcer** which will give you combination of lower and upper case alphabets characters A-Z,a-z and numbers from 0-9. Set min length and max length to 10 because the password is 10-digit length.

Click on **Options** scroll down to find **Grep-Match.** In this case, when a user is authenticated, the server redirects him to dashboard **https://portal.abu.edu.ng/staff/dashboard**. When grep-match finds dashboard link on HTTP response body, it will indicate with a blue tick meaning that correct match of username and password has been found.

After following the above procedures carefully, click on **Start attack** to start brute-force attack.

C. Brute-force attack on student domain

Referring to the above authentication concept of Ahmadu Bello University, here also we're brute-forcing username and password on student domain. In this case ABU provide registration number for every student as username and state of origin as password for authentication.

After filling the authentication requirement with ********* as username and ****** as password, click on login. Then you go to burp suite and click on **proxy**, under proxy click on **HTTP history**, that is where all the requests sent to the server are recorded, look for **post /login** request, right click and click on **Send to Intruder**.

In this attack we are creating a list of 200 registration number and a list of 36 states in Nigeria that will serve as usernames and passwords. Each username in the list will be tested with every password on the list until correct match are found. In this case, we need two payload positions. Click on **Intruder** set the Attack type to **Cluster bomb**. Highlight username, password and click on **Add $** to add the payload positions individually. Click on **payloads** tab and fill in payload 1 and 2 respectively, payload 1 will serve as list of usernames and payload 2 as list of passwords. Click on **Options** tab scroll down to find **Grep-Match.** In this case, the redirection link is **https://portal.abu.edu.ng/dashboard.** Click on **Start attack** to start brute-force attack.

1. Insecure Direct Object Reference (IDOR)

Insecure direct object reference (IDOR) is a vulnerability that occurs when a user is able to view, edit or delete unauthorized data.

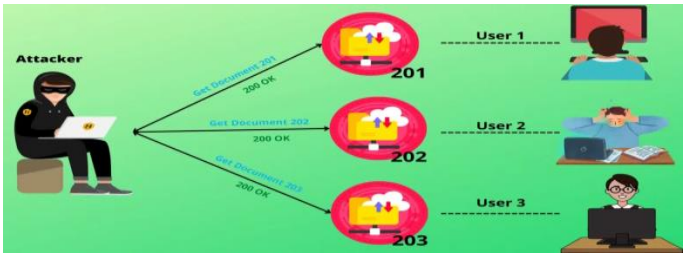

Fig. 2: Shows the concept of IDOR

i. IDOR on staff domain

The issue here is that the developer failed to implement proper access controls when calling resources so users can access and edit other users' data. The backend application uses the **"employee_id"** value when performing these actions without any other verification. So as an attacker we could easily modify and edit another user Curriculum vitae (CV) on the application. Sometimes it is as easy as changing your user id to another users, in this case the **employee_id** is "*****" which can easily be modified to iterate. Now, we change the employee_id to" ****" since the **employee_id** is guessable and increments by one for every user. This attack could also be scripted to exploit every user on the application.

Another IDOR again but on pay slip functionality which allow a user to view another user pay slip. The issue here is that the developer failed to implement proper access controls when calling resources so users can access and view other users' data. The backend application uses base64 encoding scheme to encode pay slip value. When performing these actions, the application uses laravel encryption that contains iv, value, and mac. iv is a randomly generated fixed-size data injected at each run. It is base64 encoded, value is the actual ciphered data encoded in base64, and the mac is the authentication code that is used to detect value tempering, generated hashing value and iv.

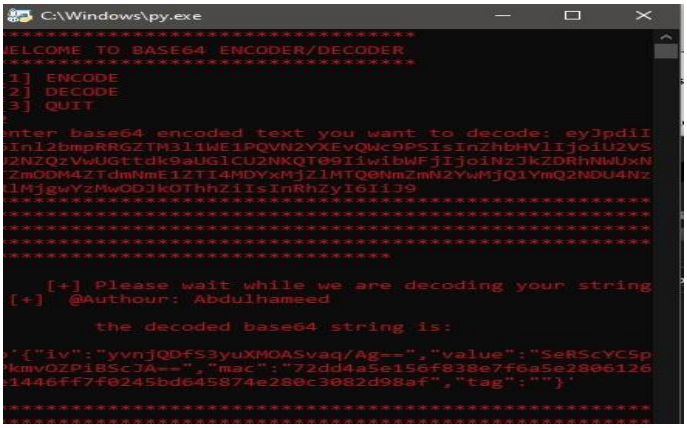

Fig.3: Shows base64 decode of laravel encryption

2. IDOR on student domain

The issue here is that the developer failed to implement proper access controls when calling resources so users can access and view other users' data. The backend application include **id** in Get request for each user while **id** is used for retrieving accommodation details of a user. When performing these actions, the application so that without any other verification or access control check. So, an attacker could easily modify and view another user accommodation details.

Now, we change **id** from "******" to" ******". Since the **id** is guessable and generated randomly for every user, this attack could also be scripted to exploit every user on the application.

3. Cross-site scripting (XSS)

XSS allows attackers to execute arbitrary java script code on the target browser. XSS is when attacker inject JavaScript code

through the vulnerable part of the website with the purpose to steal cookies.

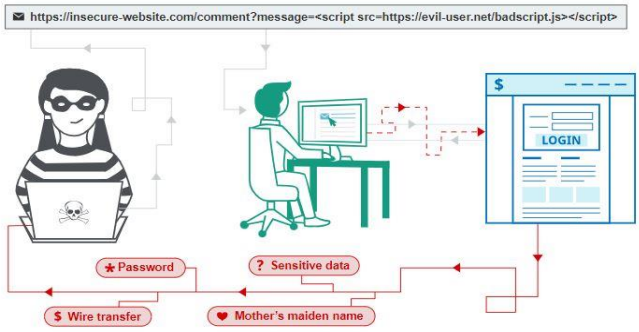

Fig. 4: Shows the concept of XSS

XSS on student Moodle

The issue here is that on student profile, there are two endpoints **Edit profile** and **Blog Entries,** under **Edit profile,** there's an endpoint **Description** where user will type his description without any character limitation, under **Blog Entries** there's and endpoint **Add a new entry** which allow user to add new blog post, when adding new blog post, there's an endpoint **Blog entry body** that allow user to type blog entry body without any character limitation.

The application is not sanitizing user input properly and it is using **outerHTML** to get or set HTML content of an element without proper validation or escaping when rendering user input on browser. So an attacker could input malicious payload to execute arbitrary java script code on user's browser.

```
> outerHTML
< `<img src="1" onerror="alert('HACKED!!!')">`
> onerror
< ƒ onerror(event) {
    alert('HACKED!!!')
  }
> document.domain
< 'moodle.abu.edu.ng'
```

Fig. 5: Shows the outerHTML content.

HTML injection

HTML injection is when attacker inject HTML code through the vulnerable part of the website with the purpose to change the website's design or to steal another person's identity or redirection.

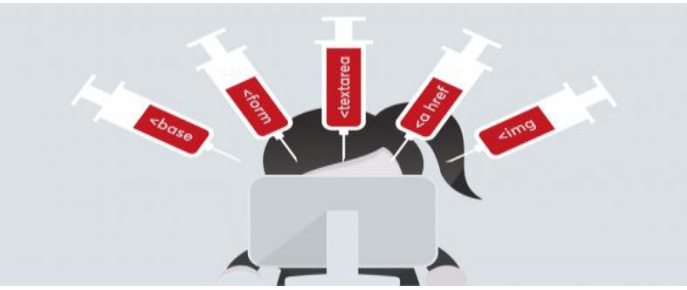

Fig. 6: Shows the concept of HTML injection

HTML injection on student Moodle

There is a functionality on the system that allow user to search and send private messages to users (all undergraduate students) on the system. The application is not sanitizing user input properly and setting HTML content without proper validation or escaping when rendering user input on browser. So, an attacker could input malicious payload to execute arbitrary HTML code on user's browser.

V. IMPLEMENTATION

This paper talks about the findings (results) of the paper and present in a clearly understandable form and requirement specifications for adopting the project. Its impact and possible solutions are pointed out.

1. Encoding

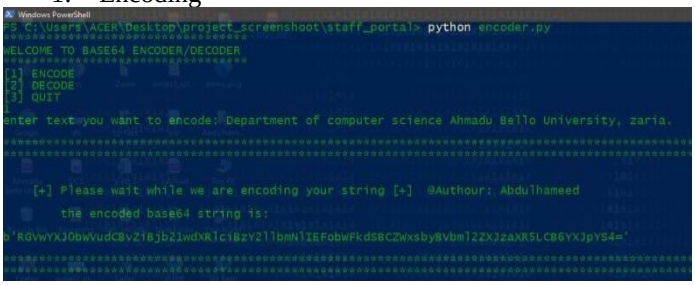

Fig. 7: Shows base64 encoding

shows the system version of base64 encoding text "Department of computer science Ahmadu Bello University, Zaria.

2. Decoding

Fig. 8: Shows base64 decoding

Shows the system version of base64 decoding text "RGVwYXJ0bWVudCBvZiBjb21wdXRlciBzY2llbmNlIEFobWFkdSBCZWxsbyBVbml2ZXJzaXR5LCB6YXJpYS4="

Student Domain

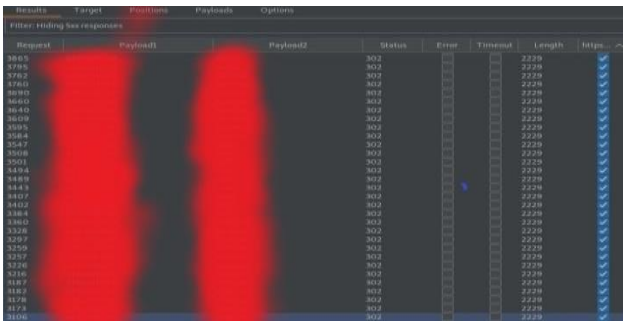

Fig. 9: Shows the result of brute force attack using cluster bomb attack

Impact

1. Unauthorized access to Account.
2. Identity theft.
3. Legal and academic implication.

Fig. 10: Shows IDOR led to view accommodation transaction receipt.

Impact

1. Unauthorized data access.
2. Data exfiltration.
3. Privacy violation.

i. ***Staff Domain***

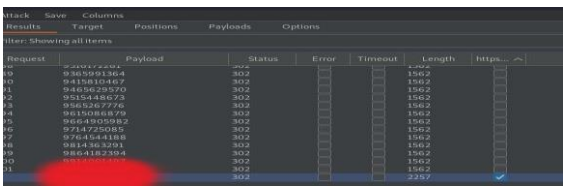

Fig.11: Shows the result of brute force attack using sniper attack.

Impact

1. Reputational damage.
2. Identity theft.
3. Social engineering.
4. Loss of digital assets.
5. Legal and financial implication.

Fig.12: Shows IDOR led to viewing and editing of Curriculum Vitae (CV)

Impact

1. Unauthorized data access.
2. Data exfiltration and modification.
3. Privacy violation.

Fig. 13: Shows IDOR led to viewing of salary pay slip

Impact

1. Unauthorized data access.
2. Data exfiltration.
3. Privacy violation.
4. Financial fraud.
5. Bank account compromise

4. Student Moodle Domain

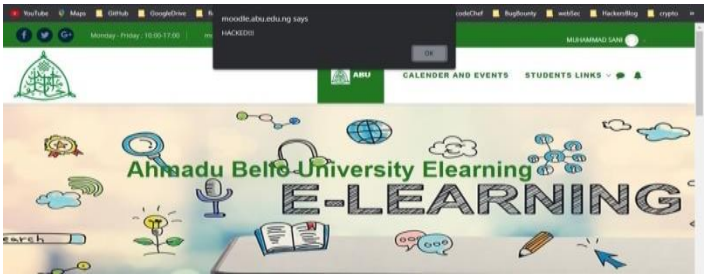

Fig. 14: Shows the result of cross-site scripting (xss)

Impact

1. Malware distribution
2. Account takeover
3. Cookie theft

Fig. 15: Shows the result of HTML injection

Impact

1. To change the displayed website's appearance.
2. To steal another person's identity.
3. To redirect user to malicious website.

VI. Conclusion

This paper identifies and analyzes seven vulnerabilities discovered within ABU's web applications, including two instances of brute force attacks, three cases of Insecure Direct Object References (IDOR), one Cross-Site Scripting (XSS) vulnerability, and one HTML injection vulnerability. By addressing the specific challenges faced within the Ahmadu Bello University domain, this case study provides valuable insights and practical recommendations to strengthen the security posture of ABU's digital infrastructure. It serves as a crucial resource for the university's IT and security teams in maintaining the integrity and resilience of their systems.

References

[1] M. Ali A. Hammoudeh, A. Alobaid, A. Alwabli, F. Alabdulmunim (2021). The Study on Assessment of Security Web Applications, November 2021, ijim, Vol. 15, No. 23. https://doi.org/10.3991/ijim.v15i23.27357

[2] M. M. Naier, A. Hamidi, R. Momand (2020) Analysis of Web Application Security Vulnerabilities, 2020, International Journal of Scientific Engineering and Science, Volume 4, Issue 8, pp. 45-51, http://ijses.com/

[3] D. Kaura, P. Kaurb (2016), Empirical Analysis of Web Attacks, December 2016 International Conference on Information Security & Privacy (ICISP2015),pp. 298 – 306 (http://creativecommons.org/licenses/by-nc-nd/4.0/).

[4] S. Kumar, R. Mahajan, N. Kumar, S. K. Khatri(2017),A Study on Web Application Security and Detecting Security Vulnerabilities, September 2017, 6th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO), 978-1-5090-3012-5/17/$31.00 ©2017 IEEE, https://www.researchgate.net/publication/324935252.

[5] H. Atashzar, A. Torkaman, M. B. Mohammad, H. Tadayon, (2011) A Survey on Web Application Vulnerabilities and Countermeasures, January 2011, Communication Usage Trend Survey, Ministry of Internal Affairs and Communications (MIC), http://www.soumu.go.jp.

[6] M. Aydos, C. Aldan, E. Coskun, A. Soydan, (2021), Security testing of web application: A systematic mapping of the literature september2021, Journal of King Saud University–Computer and Information Sciences 34 pp. n0 6775–6792, https://doi.org/10.1016/j.jksuci.2021.09.018, http://creativecommons.org/licenses/by-nc-nd/4.0/)

[7] Shuaibu, B. M., Norwawi, N. M., Selamat, M. H., & Al-Alwani, A. (2015). Systematic review of web application security development model. Artificial Intelligence Review, 43(2), 259–276.

[8] Dafydd, S., & Marcus, P. (2011). The Web Application Hackers Handbook: Finding and Exploiting Security Flaws. (2th ed.). John Wiley & Sons, Inc.