Comparative Analysis of Automated Scanning and Manual Penetration Testing for Enhanced Cybersecurity

Nikhil Rane and Amna Qureshi
School of Computer Science, Artificial Intelligence and Electronics
University of Bradford, Bradford, United Kingdom
{nsrane,a.qureshi19}@bradford.ac.uk

2024 12th International Symposium on Digital Forensics and Security (ISDFS) | 979-8-3503-3036-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/ISDFS60797.2024.10527240

Abstract— **Web platform security has become a significant concern in the current cyber world. Adversaries constantly advance their skills and technologies to bypass modern cyber defence techniques to lure website vulnerabilities. In the cyber world, finding and mitigating vulnerabilities on the website is essential to avoid any damage to the organization. Two key techniques - vulnerability assessment and penetration testing - play a crucial role in identifying and mitigating these weaknesses. While vulnerability assessment scans the platform, revealing potential flaws, penetration testing goes a step further, simulating real-world attack scenarios to assess their true exploitability and possible damage. This paper compares automated scanning and manual penetration testing to evaluate the effectiveness of these techniques in uncovering vulnerabilities. The experimental results confirm that manual penetration testing is more effective than automated testing in terms of accuracy. Additionally, practical studies highlight the importance of a penetration tester's skills and experience in identifying and exploiting security weaknesses. Automated tools may also generate false positive results.**

Keywords— Vulnerability analysis, Penetration testing, Website security, Ethical hacking.

I. INTRODUCTION

In today's digital age, businesses are increasingly moving towards cloud-based resources to scale their operations and enhance their functionality. However, with the growing dependence on the internet, the risk of cyber threats has also increased. Unfortunately, most businesses prioritize functionality over security, leaving their websites vulnerable to attack. This can have grave consequences, including jeopardizing the safety of critical data and damaging the reputation of the organization. Therefore, it is vital to identify these vulnerabilities at an early stage, before they become accessible to hackers. Vulnerability assessment and Penetration testing [1] are two widely recognized methods that can help businesses achieve this objective. By adopting these measures, businesses can protect themselves from potential cyber threats and comply with data regulations.

Vulnerability Assessment is a security process in which an expert evaluates a target using vulnerability analysis tools [2]. This approach helps an organization review its website for security threats by performing vulnerability scanning to determine whether the organization is at risk. The process involves security specialists scanning the target website with vulnerability assessment tools [3]. These tools examine the vulnerability, determine its severity based on the analysis, and propose mitigating strategies.

In contrast to vulnerability assessment, penetration testing [4] involves a tester manually testing the website using tools. The pentester accesses the target, carries out vulnerability analysis using an automated vulnerability analysis tool, and exploits the vulnerability based on the analysis. The tester rates the severity of the vulnerability based on the threat modelling (like DREAD and STRIDE) assessment and documents the vulnerabilities exploited [5]. This research paper aims to compare the two processes of finding website vulnerabilities using the DREAD threat model and provides a practical comparison of both methods.

This paper is structured as follows. Section II presents the background of vulnerability analysis and penetration testing. Section III discusses types of vulnerabilities. Section IV describes the results of the vulnerability assessment on the targeted domain. Section V discusses the practical exploitation of the vulnerabilities in the target domain. Section VI presents the comparative analysis of vulnerability assessment and penetration testing. Finally, Section VI concludes this paper.

II. BACKGROUND

This section presents the background of vulnerability analysis and penetration testing.

A. Vulnerability Assessment

A website vulnerability is a weakness or threat that attackers can exploit to gain unauthorized access or cause harm to the website or its users [6]. Security experts identify these vulnerabilities by using tools to analyze the potential threats an attacker can exploit. Automated tools scan the website to identify potential security vulnerabilities that can be exploited. The scan produces a list of identified vulnerabilities, their impact, severity levels, and necessary remediations [7]. The severity of the identified security vulnerabilities is classified by the tool. The identified vulnerabilities don't always need to be present on the website.

Vulnerability analysis is also essential for compliance. Many industries and regulatory frameworks require regular vulnerability assessments to ensure compliance with security standards [8]. For example, the Payment Card Industry Data Security Standard (PCI DSS) requires regular vulnerability assessments to ensure compliance with its security standards. Organizations can verify that they satisfy these requirements and avoid potential penalties or legal ramifications by conducting frequent vulnerability assessments [9]. Another significant reason vulnerability analysis is crucial is that organizations can detect vulnerabilities before they are exploited, rather than waiting for an attack to occur. This proactive approach can help organizations stay ahead of cyber threats and prevent possible breaches from occurring. The

979-8-3503-3036-6/24/$31.00 ©2024 IEEE

commonly used tools for scanning websites are Acunetix, Nikto, Netsparker, Nmap, and Burp Suite.

B. Penetration Testing

Penetration testing is a strategic process in which cybersecurity experts simulate attacks in a test environment to exploit identified threats on a website. This process is more of a manual approach which is performed by ethical hackers who use tools and their skills to identify vulnerabilities and weaknesses in the website before malicious attackers exploit them [10]. They can use any tools or techniques to achieve their goal by simulating different types of attacks, such as brute-force attacks, business logic vulnerabilities, or access control vulnerabilities. Once the testing is completed, a report is generated that outlines the vulnerabilities and weaknesses that were discovered. The severity of the exploited vulnerability is rated using the DREAD threat modelling, which takes into account the Damage, Reproducibility, Exploitability, Affected users, and Discoverability of the vulnerability. Based on the results, security experts recommend mitigation measures for the vulnerabilities, such as installing security patches and applying effective countermeasures.

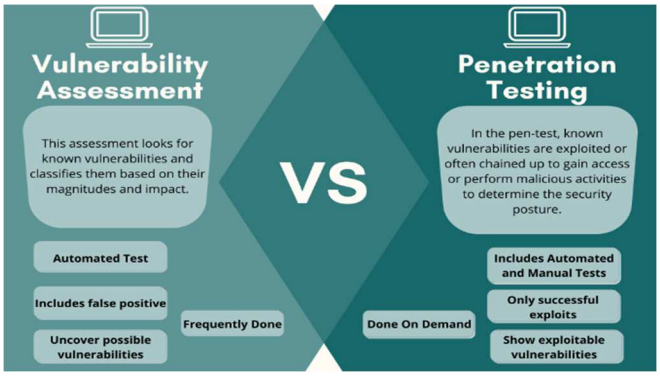

Fig. 1. Vulnerability Assessment vs Penetration Testing

By proactively identifying vulnerabilities, organizations can take steps to address them before malicious attackers can exploit them. Some commonly used tools for scanning websites and exploiting vulnerabilities include SQLmap, Burp Suite, Nmap, Metasploit, and Netsparker. Fig. 1 highlights the difference between vulnerability assessment and penetration testing.

C. Types of Penetration Methodologies

The three primary methodologies for website penetration testing are black box, grey box, and white box testing to identify vulnerabilities [11].

1. Black Box testing

Black box testing is a type of security testing that simulates a real-world attack on a website [12]. The tester works as an outsider, just like a real attacker, with no prior knowledge of the website, its internal documentation, source code, or configuration information that would give them an advantage. The testing process begins with the security expert gathering information on the website using various methods, such as port scanning, network mapping, and web application crawling. This information helps them identify potential attack vectors and vulnerabilities in the web application. The tester then attempts to exploit the identified vulnerabilities to gain unauthorized access or extract sensitive information using a combination of automated tools and manual techniques. The goal of this testing is to identify vulnerabilities that may not be detected in other types of testing. Black box testing provides a more realistic view of the website's security posture, as it approaches the website as an external attacker [13]. This helps to identify potential gaps in security controls and processes. However, it is a time-consuming and resource-intensive process because the tester has to perform reconnaissance to gather information about the target website before launching attacks and has no access to the code or documentation, which requires more scanning and skills.

2. White Box testing

White box testing is a comprehensive security testing approach that involves a tester having complete understanding and access to the target website [14]. This access can include source code, network diagrams, website architecture, and other internal documentation. The tester leverages this information to identify potential vulnerabilities and weaknesses in the website's design and implementation. They can then perform various tests to determine if these vulnerabilities can be exploited by attackers or not. These tests range from basic vulnerability scans to advanced exploitation techniques. One of the main benefits of white box testing is that it can identify vulnerabilities before the website is deployed, saving time and resources in the long run. Additionally, it can be more comprehensive than black box testing since the tester has full knowledge of the website. However, it is important to keep in mind that white-box testing has certain limitations [15]. Since the tester has access to internal information that external attackers would not have, the testing may not accurately reflect the real-world security posture of the website. The tester needs to have full access to the website, which can be expensive and time-consuming. They may also require a deep understanding of the website architecture and implementation.

3. Grey Box testing

Grey box testing is a type of security testing where the tester has limited insider knowledge about the website [16]. In this type of testing, the tester has some access to the website but not full access like in white box testing. The tester may be given some information about the website, such as user credentials, network diagrams, or website architecture, and they use this information to identify weaknesses and vulnerabilities in the website's design and implementation. It can also involve a degree of reconnaissance, where the tester uses the limited access they have to the website to gather additional information that can be used in tests, e.g., security experts may attempt to perform password guessing or brute force attacks to gain additional access to the website. One of the primary advantages of grey box testing is its ability to provide a more realistic assessment of the website's security posture than black box testing alone [17]. By having some level of insider knowledge, the tester can simulate the types of attacks that an insider with some level of access might carry out. However, it is important to note that grey box testing also has its limitations. The testing may not fully reflect the real-world security posture of the website since the tester has only partial knowledge and access to the website. Grey box testing can be challenging to execute, as it requires

trust and collaboration between the tester and the website administrators [18].

III. Types of Vulnerabilities

Vulnerabilities refer to weaknesses or flaws within a system or website that can be exploited by attackers to compromise its security. These vulnerabilities might arise from configuration weaknesses, design flaws, or even human errors, ultimately exposing the system to potential risks and unauthorized access. This section discusses some of these vulnerabilities.

A. Reflected Cross-Site Scripting

Cross-site scripting (XSS) is a security vulnerability that happens when a website fails to properly validate and sanitize user input in search fields. This can give attackers an opportunity to inject their own malicious code into the website. When other users view that web page, they may unknowingly execute the intruder's code, which could lead to stealing their personal information session cookie or more harmful attacks [19].

B. SQL Injection

SQL injection is a type of security vulnerability that occurs when a hacker tries to insert malicious SQL statements into a vulnerable website or application. Structured Query Language (SQL) is a programming language used for managing databases, and SQL injection attacks involve inserting SQL code into input fields or other critical sections of a web application, which can lead to the exposure of sensitive user data or the compromise of data integrity [19].

C. URL Redirection

URL redirection on a login page is a security issue that allows attackers to send users to victim websites where they might be tricked into giving away their login credentials [20].

D. Cross-Site Request Forgery

Cross-Site Request Forgery (CSRF) is a type of security vulnerability that enables attackers to execute malicious actions on behalf of the user without their knowledge or consent. This can happen when a user unknowingly clicks on a link or submits a form on a website that has been maliciously modified by an attacker [19].

E. Privilege Escalation

Privilege Escalation is a security issue where an attacker gains access to higher levels of permissions on a website than they are supposed to have. This can enable them to execute actions that are usually restricted to more privileged users, such as deleting users or accessing sensitive data [19].

F. Brute Force

Brute force on a login page is a type of security vulnerability where a hacker tries to guess a victim's login credentials by repeatedly attempting different combinations of passwords until the correct one is found. Automated tools such as Burp Suite can be used to speed up and make this process more effective [21].

G. Insecure Direct Object Reference

It is a security vulnerability that allows attackers to access or manipulate sensitive information on a website by referencing its unique identifier in the URL or a hidden form field.

H. Clickjacking

Clickjacking is a form of attack in which an attacker tricks the victim into clicking on something the victim did not intend to click on. The attacker can accomplish this by adding a hidden element on a website, such as a fake login button. When the victim clicks on what appears to be the real button, they are actually clicking on the concealed one. This can lead to dangerous consequences, such as the attacker stealing the victim's login credentials [18].

IV. Vulnerability Assessment

This section presents the results of the vulnerability assessment conducted on a target website (https://hack-yourself-first.com/) using an automated tool, Netsparker, newly known as Invicti.

A. SQL Injection

During the target website's vulnerability analysis, Netsparker discovered an SQL injection vulnerability, which the tool confirmed, as shown in Fig. 2.

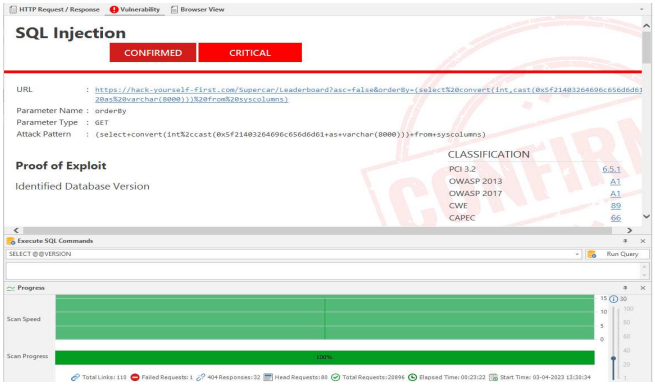

Fig. 2. SQL Injection Vulnerability.

B. Reflected Cross-Site Scripting

Upon conducting a vulnerability analysis on the target website, Netsparker has detected the reflected cross-site scripting vulnerability (as shown in Fig. 3), and it is confirmed as per the tool.

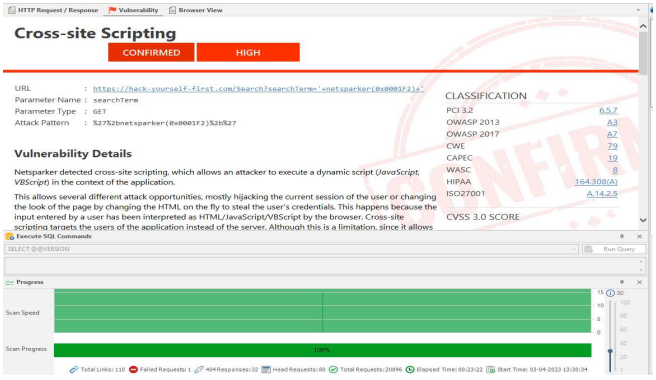

Fig. 3. XSS Vulnerability.

C. Cross-Site Request Forgery

During vulnerability analysis of the target, Netsparker detected a CSRF vulnerability, as shown in Fig. 4.

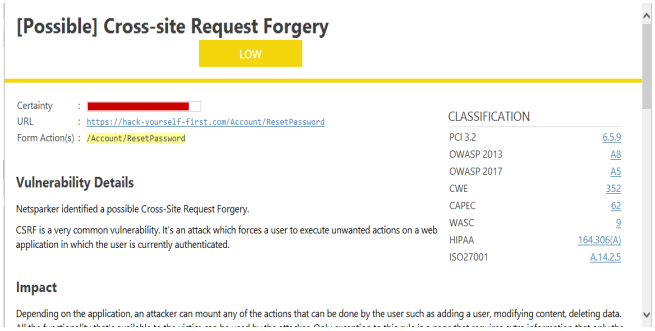

Fig. 4. CSRF Vulnerability.

D. Clickjacking

During vulnerability analysis of the target website, Netsparker detected a missing X-Frame-Options header which may result in clickjacking, as shown in Fig. 5.

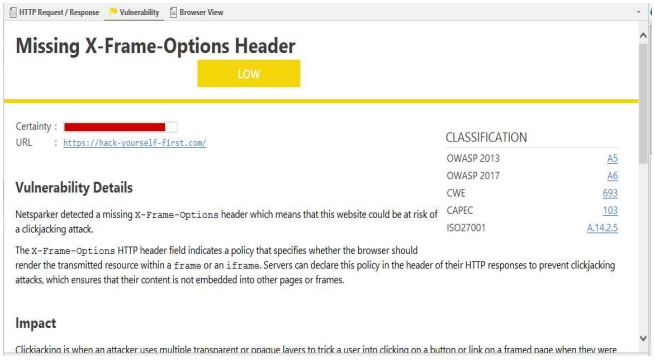

Fig. 5. Clickjacking.

V. Penetration Testing Phase

This section discusses the practical exploitation of the vulnerabilities in the target domain.

A. SQL Injection

According to the vulnerability assessment the vulnerability assessment tool has confirmed the presence of SQL injection. However, during the penetration testing phase, the same vulnerability could not be detected due to a server error caused by a failed conversion of a varchar value, as shown in Fig. 6. Since this vulnerability cannot be manually exploited, this detection of the vulnerability can be considered a false positive.

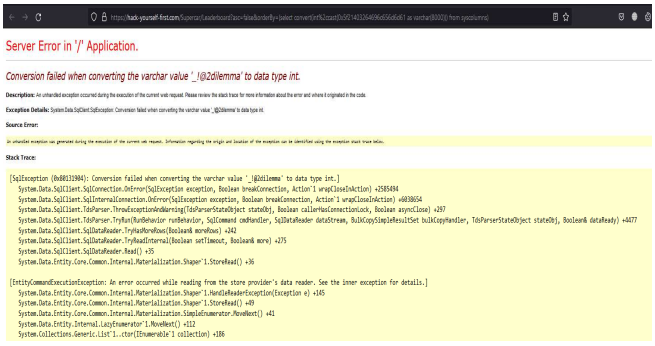

Fig. 6. Unsuccessful Exploitation of SQL Injection.

B. Reflected Cross-Site Scripting

As detected in the vulnerability assessment phase, there is a detection of cross-site scripting vulnerability, and it has been successfully exploited in the penetration testing phase. There is an exploitation of cookie in this process, so it is detected as high severity in accordance with DREAD, as shown in Fig. 7.

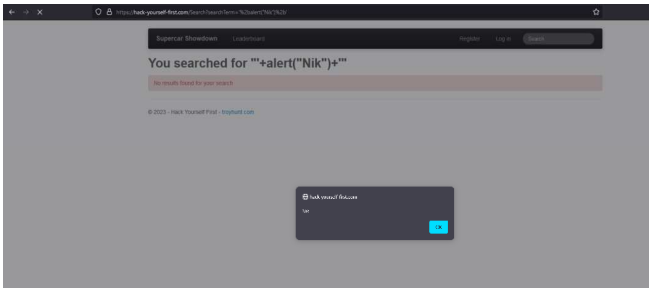

Fig. 7. Exploitation of XSS Vulnerability.

C. Cross-Site Request Forgery

The vulnerability assessment step detected CSRF vulnerability in the change password section, and as we can see, the vulnerability has been successfully exploited, as shown in Fig. 8.

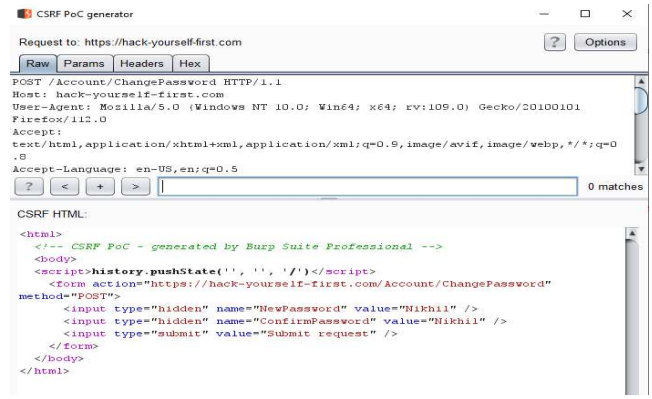

Fig. 8. Exploitation of CSRF.

Although the severity of this vulnerability was identified as low during the assessment, it is actually quite severe because attackers can take over the victim's account by simply sending an HTML file and clicking on a button. However, when it comes to penetration testing, it is said to be an impactful vulnerability and can be documented as high severity in accordance with DREAD.

D. Clickjacking

During the vulnerability assessment phase, a lack of an x-frame header was detected, leading to a clickjacking vulnerability. This vulnerability was effectively exploited during the penetration phase (as shown in Fig. 9) and is marked as medium severity in accordance with DREAD.

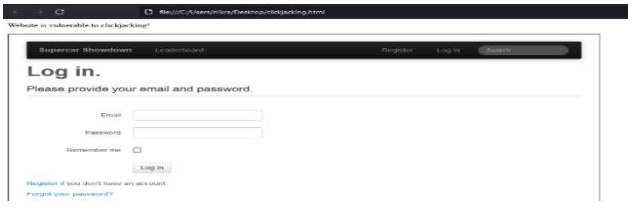

Fig. 9. Exploitation of Clickjacking.

E. URL Redirection

The original URL of the target website is: https://hack-yourself-first.com/Account/Login?ReturnUrl=%2fAccount%2fUserProfile%2f465

The manipulated URL is: https://hack-yourself-first.com/Account/Login?ReturnUrl=https://www.google.com

Upon logging in with the provided credentials, the user will be redirected to a new page, as shown in Fig. 10. The vulnerability assessment tool does not discover this vulnerability, it is manually identified during the penetration testing. While vulnerability assessment tools can only identify known parameters/vulnerabilities, a security expert can detect and exploit many unique flaws during penetration testing. According to the DREAD severity assessment, this vulnerability is classified as a high-level severity.

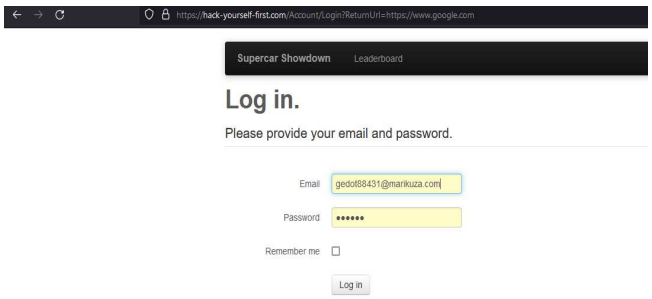

Fig. 10. Exploitation of URL Redirection.

F. Privilege Escalation

For this attack, we consider the following scenario: the attacker logs into the account and changes the cookie IsAdmin to true using a cookie editor or by right-clicking and inspecting element. This gives the attacker admin privileges, as shown in Fig. 11.

Such vulnerabilities cannot be exploited with the assistance of automated tools because they require careful examination, which the tool is not trained to do. A manual approach is necessary to find and exploit such significant vulnerabilities. Considering the potential impact and the ease of exploitation, this vulnerability is considered critical in accordance to DREAD.

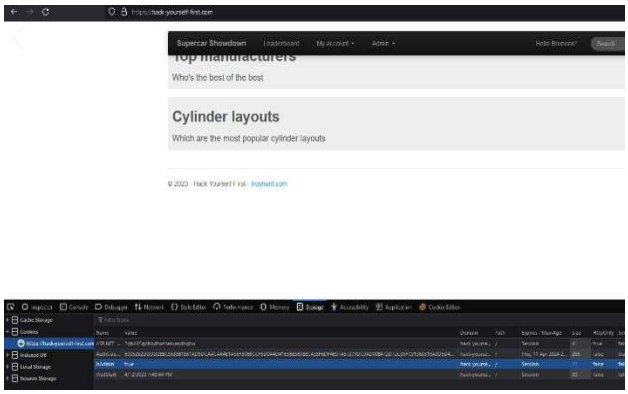

Fig. 11. Exploitation of Privilege Escalation.

G. Brute Force

The website's login page lacks rate limitation, making it vulnerable to brute force attacks that can lead to account takeover (as shown in Fig. 12). This vulnerability cannot be detected by a vulnerability assessment tool but can be exploited through either manual or automated penetration testing. According to the DREAD severity assessment, this vulnerability is considered to be of high severity.

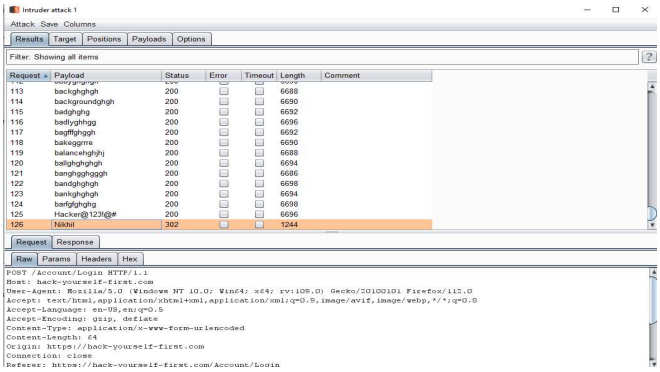

Fig. 12. Exploitation of Brute Force.

H. Indirect Object reference

It is a vulnerability that allows an attacker to gain access to another user's account simply by changing the user profile id number in the URL and even changing the other user's credentials, as shown in Fig. 13. Based on the severity of the parameters and impact (as per DREAD), it has been deemed a high-level threat.

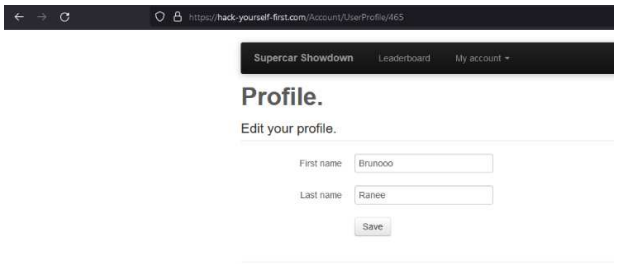

Fig. 13. Exploitation of Indirect Object Reference.

VI. COMPARATIVE ANALYSIS

As indicated by the vulnerability assessment, a SQL injection vulnerability was detected, which was also observed in the penetration testing exploitation phase. However, during the testing of the SQL injection vulnerability, it was discovered that the SQL injection was not present, and the given error was due to the failure to convert the varchar value. Therefore, it can be concluded that the vulnerability assessment tool can sometimes yield false positive results. Nonetheless, the tool was able to identify other vulnerabilities, such as XSS, clickjacking, and CSRF vulnerabilities, which can be exploited during the penetration phase.

During the penetration phase, more vulnerabilities that could pose a direct threat to the website or its users were discovered. Additional vulnerabilities discovered during the penetration phase include privilege escalation and indirect object reference, which resulted in the change of users' credentials, login page-based URL redirection, and brute force attack that resulted in account takeover.

In analyzing the results of the vulnerability assessment and penetration testing, it was discovered that the result of the vulnerability assessment is based on the vulnerability that can be exploited, whereas the result of penetration testing is based on the exploitation of vulnerability.

From TABLE 1, it is evident that while vulnerability assessment missed privilege escalation and IDOR vulnerabilities, penetration testing identified and successfully exploited these security risks, emphasizing the comprehensive nature of penetration testing in evaluating the real-world impact of vulnerabilities.

TABLE 1. COMPARATIVE ANALYSIS

Vulnerability	Vulnerability Assessment	Penetration Testing
SQL injection	Detected	False Positive Output
XSS	Detected	Exploited
CSRF	Detected	Exploited
Clickjacking	Detected	Exploited
URL Redirection	Not Detected	Detected and exploited
Privilege Escalation	Not detected	Detected and exploited
Brute Force	Not Detected	Detected and exploited
IDOR	Not detected	Detected and exploited

VII. CONCLUSION

Through both study and practical experimentation, it has been determined that manual penetration testing is more effective in terms of accuracy. Automated vulnerability assessment tools may sometimes generate false positive results, such as detecting SQL injection vulnerabilities when there are none. Automated scanning tools are outdated as they only illustrate the possibility of false positives in automated scanning. However, testing should not rely solely on automated scanning. The combination of automated scanning and manual penetration testing yields better results. Manual penetration testing is more precise since it is based on the tester's practical skills and knowledge. Penetration testers who possess skills and knowledge of automation tools perform better attacks to penetrate a website. Using the manual testing approach, security experts are able to discover and exploit various types of attacks such as cross-site scripting, SQL injection, clickjacking, authentication bypass, and CSRF attacks. In order to save time and resources, automation testing is utilized to detect vulnerabilities in a website. The vulnerability assessment tool proves useful during the penetration phase as it helps discover which parameters and URLs are vulnerable to certain types of vulnerabilities. This makes it easier for security experts to penetrate and exploit these vulnerabilities.

REFERENCES

[1] Shinde, P.S. and Ardhapurkar, S.B., "Cyber security analysis using vulnerability assessment and penetration testing", World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave), pp. 1–5, 2016.

[2] Garcia, M.L., "Vulnerability assessment of physical protection systems". 1st ed, Elsevier, 2005.

[3] Curphey, M. and Arawo, R., "Web application security assessment tools", IEEE Security & Privacy, 4(4), pp.32-41, 2006.

[4] Xynos, K. et al., "Penetration testing and vulnerability assessments: A professional approach", International Cyber Resilience Conference. pp.126-132, 2010.

[5] Nweke, L.O. and Wolthusen, S., "A review of asset-centric threat modelling approaches", International Journal of Advanced Computer Science and Applications (IJACSA), 11 (2), pp. 1-6, 2020.

[6] Setiawan, E.B. and Setiyadi, A., "Web vulnerability analysis and implementation", In IOP conference series: materials science and engineering, vol. 407, no. 1, pp. 012081, 2018.

[7] Holm, H., Sommestad, T., Almroth, J. and Persson, M., "A quantitative evaluation of vulnerability scanning", Information Management & Computer Security, 19(4), pp.231-247, 2011.

[8] Ten, C.W., Liu, C.C. and Manimaran, G., "Vulnerability assessment of cybersecurity for SCADA systems", IEEE Transactions on Power Systems, 23(4), pp.1836-1846, 2008.

[9] Shah, S. and Mehtre, B.M., "An overview of vulnerability assessment and penetration testing techniques", Journal of Computer Virology and Hacking Techniques, 11(1), pp. 27–49, 2015.

[10] Goel, J.N. and Mehtre, B.M., "Vulnerability Assessment & Penetration Testing as a Cyber Defence Technology", Procedia Computer Science, 57, pp. 710–715, 2015.

[11] Singh, N., Meherhomji, V. and Chandavarkar, B.R., "Automated versus Manual Approach of Web Application Penetration Testing", 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), pp. 1–6, 2020.

[12] Felderer, M., Büchler, M., Johns, M., Brucker, A.D., Breu, R. and Pretschner, A., "Security testing: A survey", In Advances in Computers, vol. 101, pp. 1-51, 2016.

[13] Pasquale, F., "The Black Box Society: The Secret Algorithms that Control Money and Information", Harvard University Press, 2015.

[14] Mahmood, R., Esfahani, N., Kacem, T., Mirzaei, N., Malek, S. and Stavrou, A,. "A white-box approach for automated security testing of Android applications on the cloud", 7th International Workshop on Automation of Software Test (AST), pp. 22-28, 2012.

[15] Chan, M.Y. and Cheung, S.C., "Applying white box testing to database applications", Hong Kong University of Science and Technology, Department of Computer Science, Tech. Rep. 1999.

[16] Khan, M.E. and Khan, F., "A comparative study of white box, black box and grey box testing techniques", International Journal of Advanced Computer Science and Applications, 3(6), 2012.

[17] Sutton, M., Greene, A. and Amini, P., "Fuzzing: brute force vulnerability discovery", 1st ed, Pearson Education, 2007.

[18] Nguyen, H.Q., "Testing applications on the Web: Test Planning for Internet-based systems", 2nd ed, John Wiley & Sons, 2001.

[19] Nagpure, S. and Kurkure, S., "Vulnerability Assessment and Penetration Testing of Web Application", International Conference on Computing, Communication, Control and Automation. pp. 1–6, 2017.

[20] Sinha, S., "Header Injection and URL Redirection", Bug Bounty Hunting for Web Security: Find and Exploit Vulnerabilities in Websites and Applications. Berkeley, CA: Apress, pp. 79–96, 2019.

[21] Gautam, T. and Jain, A., "Analysis of brute force attack using TG—Dataset", SAI Intelligent Systems Conference (IntelliSys), pp. 984-988, 2015.