Received: 07 Sept 2023 | Revised: 24 Apr 2024 | Accepted: 02 Aug 2024 | Published: 23 Aug 2024
Volume 2(2) Pages 1-28 DOI: https://doi.org/10.56279/jicts.v2i2.52

JICTS
Journal of ICT Systems

Assessment of Vulnerabilities in Student Records Web-Based Systems of Public and Private Higher Learning Institutions in Tanzania

Wilbard G. Masue[1], Daniel Ngondya, Tabu S. Kondo

Department of Computer Science and Engineering, The University of Dodoma, Dodoma, Tanzania

[1]Corresponding author
Email: wilbard.masue@udom.ac.tz

Funding information

This study was supported by the University of Dodoma

Higher Learning Institutions (HLIs) increasingly use web-based systems to manage data, including website content, academic results, and financial records. These systems improve service delivery to stakeholders but expose HLIs to various vulnerabilities. Web-based systems at HLIs are frequently compromised due to such vulnerabilities. This study aimed to assess the vulnerabilities of Student Records Web-Based Systems (SRWBS) in private and public HLIs in Tanzania using black-box testing. Two automatic vulnerability scanners were employed: OWASP Zed Attack Proxy (ZAP), an open-source tool, and Acunetix, a proprietary tool. The study assessed the vulnerabilities in the SRWBS of three private and five public HLIs in Tanzania. Findings revealed 28 vulnerabilities, including Broken Authentication, Session Management, Security Misconfiguration, Sensitive Data Exposure, Cross-Site Request Forgery (CSRF), and Cross-Site Scripting (XSS). Public HLIs had an average vulnerability rate of 44.2%, while private HLIs were vulnerable at 37%. This indicates that public HLIs are generally more at risk. Efforts to secure web-based systems should prioritize addressing the most common vulnerabilities identified in this study.

Keywords

Cyber security assessment

Web system vulnerabilities

Vulnerability assessment

Vulnerability scanning

1. Introduction

The higher education sector in Tanzania is a tactical agent for development of the country [1]. Most Higher Learning Institutions (HLIs) across the world are managing their task in web-based systems [2]. These web-based systems in HLIs store potential information. With advancement of information and communication technologies (ICT), the number of hackers has been increasing [3]. The kind of attacks performed by hackers in HLIs systems includes stealing data, performing unauthorized data modification or deletion [4-8].

Developers and Administrators of web-based systems have been knowingly or unknowingly overlooking security issues during development

and use of web-based systems, from planning to testing and deployments phases [9-10]. As a result, attackers have continued to exploit vulnerabilities in developed web-based systems and compromise them while disrupting their operations.

The HLIs web-based systems around the world have been compromised by attackers due to presence of vulnerabilities. Jablon [11] reported the way hackers had stolen credit card numbers, research works as well as student and employee social security numbers from the university systems and networks in the United State of America (USA). Also, in 2017, two Tanzanian HLIs websites for Open University of Tanzania and the University of Dar es salaam were reported to have been hacked, momentarily bringing down and making them inaccessible [1].

A number of studies have been conducted regarding security vulnerabilities of web-based systems in Tanzania's different sectors. The studies include vulnerability assessment of e-government websites, government ministry websites and admission system for HLIs in Arusha integrated to Tanzania Commission for University (TCU) Application Programming Interface (API) [12–16]. From the existing studies, it is evident that no study has attempted to assess security vulnerabilities for SRWBS of public and private HLIs in Tanzania. It is interesting to study vulnerabilities associated with SRWBS due to the fact that they are used to store and process financial and academic results. The students' motivation to tamper with their financial and academic results cannot be underestimated. Furthermore, students in HLIs are heavy users of computers and online services, which makes them more likely to commit cyber-crimes as observed in [17-18].

The work by Kundy and Eva [14] assessed the security threats of the selected HLIs focused on two private HLIs in Arusha. This study selected two private HLIs, namely University of Arusha and Tumaini University of Arusha (Makumira). Purposive sampling was used to select the HLIs, with questionnaire and interview as data collection tools. The study did not carry out any scanning test to establish vulnerabilities of the systems. Furthermore, the selected HLIs limit generalization of the results.

Another work by Mtakati and Sengati [13] focused on five HLIs in Arusha in assessing vulnerabilities of admission system integrated to TCU Application Programming Interface (API), considering with both private and public HLIs where the sample selection was purposive and hence it cannot be generalized for HLIs in Tanzania [13]. Additionally, the study included vulnerability scanning, but it did not specify the criteria for selecting the scanned vulnerabilities.

Likewise, another study focused on assessing security vulnerabilities of Modular Object-Oriented Dynamic Learning Environment (MOODLE) operation in 8 HLIs in different regions in Tanzania, where the sample were selected using purposive sampling that cannot yield a result which can be generalized for Tanzania [16]. The assessment of Moodle is not relevant to most of SRWBS in Tanzania as most SRWBS are in-house developed by Tanzania developers while the Moodle is open-source tool and the nature of coding and configurations are different.

It can be seen that most of the studies focused on vulnerability assessment for websites, admission systems and networks of HLIs as well as e-learning platforms. Few studies in Tanzania assessed vulnerabilities of HLIs website and systems rather than SRWBS. This study assessed the vulnerabilities of SRWBS for HLIs in Tanzania and determined the common vulnerability of SRWBS for HLIs in Tanzania.

The study was conducted using the theory on information security [19]. The theory has four key

elements namely information, resources, threats and controls. The theory applies to the study in the way that SRWBS have potential information and resources that requires the use of appropriate control measures in mitigating threats and security vulnerabilities on it.

The study contributions are as follows: firstly, this paper sought to identify and analyze prevalent vulnerabilities present in SRWBS across both private and public HLIs in Tanzania. Secondly, this paper examines the underlying causes contributing to the manifestation of vulnerabilities within SRWBS in the Tanzanian higher education landscape. Lastly, it proposes effective mitigation strategies aimed at addressing and mitigating vulnerabilities identified within SRWBS of HLIs in Tanzania.

The rest of this paper is organized as follows: the next section contains materials and methods employed, explaining procedures and approaches adopted to undertake the vulnerability assessment of SRWBS for private and public HLIs in Tanzania. Then, the results and discussion section follows which present results and their discussions obtained from the vulnerability assessment process. Finally, the conclusion and recommendations section follows.

2. Materials and Methods

Experimental black-box testing was used to assess vulnerabilities of SRWBS for private and public HLIs in Tanzania and was conducted using computers installed with vulnerability scanning tools, namely Open Webs Application Security Project Zed Attack Proxy (OWASP ZAP) and Acunetix. The tools work by intercepting packets between a web-browser and a web-application and then comparing with pre-existing vulnerabilities. With widespread application of Artificial Intelligence (AI), the tools can be made smarter by even uncovering vulnerabilities that have not been established yet. However, AI algorithms are computationally intensive and require lots of datasets [20].

The aforementioned tools were used to collect and analyze vulnerabilities from the SRWBS. This approach provided the primary data of the common vulnerabilities of SRWBS for HLIs in Tanzania. The approach was based on external black-box testing, implying that the vulnerabilities were scanned outside the network of selected HLIs SRWBS. This is because the black-box test mimics how external hackers search for flaws that could be used to launch attacks by using tools that automatically find weaknesses in web-based systems. Kikude [21], Shrivastava et al. [22], Mantra et al. [5], Murah & Ali. [23], Elisa [12], Begum et al. [24], Moniruzzaman et al. [25], Farah [2] and Kondoro & Mtebe [15] have assessed web-based system vulnerabilities using the black-box test.

2.1 Sample Selection

The research was held in eight HLIs in Tanzania (three private HLIs and five public HLIs) which were coded to HLI_A, HLI_B, HLI_C, HLI_D, HLI_E, HLI_F, HLI_G and HLI_H respectively for privacy purposes. The selected study locations from five regions in Tanzania represented the HLIs in Tanzania which uses SRWBS. This study selected HLIs samples based on proportionate stratified sampling which is probabilistic sampling techniques which divide the population into two strata namely, public HLIs SRWBS and private HLIs SRWBS [26].

Bhalerao and Kadam [27] in their research paper, proposed that the sample size for a small population with less than 1000 elements should be at least 10%, and for those with a large population greater than 1000 elements should be at least 20%. Based on this fact, selecting HLIs under this study out of 76 HLIs in Tanzania, the sample size was

10% of the population from public and private HLIs respectively. Five (5) samples were selected from public HLIs SRWBS and 3 samples were selected from private HLIs SRWBS, respectively. A total of 8 SRWBS for HLIs in Tanzania were assessed against the common web-based system vulnerabilities.

From each stratum, Simple Random Sampling (SRS) technique using the fishbowl method was used to select the HLIs included in the study. With fishbowl technique, two tables were created for private HLIs and public HLIs. The table for private HLIs had 28 HLIs and the table for public HLIs had 48 HLIs. The HLIs were numbered from number one to twenty-eight for private HLIs and from number one to forty-eight for public HLIs. Then two bowls of small rolled pieces of papers numbered as per the HLIs in the two tables for public HLIs and private HLIs were prepared. Five pieces of papers were randomly drawn from the public HLIs bowl and three pieces of papers were randomly drawn from private HLIs bowls. After drawing the random piece of papers for public and private HLIs, the numbers were followed to mark their names where the sample for public HLIs and private HLIs were obtained. The coded list of selected HLIs based on the proportionate stratified sampling technique has been shown in Table 1.

Table 1. Selected HLIs.

S/N	HLI code	Category	Location
1	HLI_A	Public	Dodoma
2	HLI_B	Public	Dodoma
3	HLI_C	Public	Arusha
4	HLI_D	Public	Dar es salaam
5	HLI_E	Public	Morogoro
6	HLI_F	Private	Dodoma
7	HLI_G	Private	Iringa
8	HLI_H	Private	Arusha

2.2 Vulnerability Scanners Selection

The HLIs SRWBS web-based system vulnerabilities were assessed using OWASP ZAP and Acunetix vulnerability scanners. The reason for selection of these two vulnerability scanners is to combine the power of mostly used and best open source and commercial vulnerability scanners so as to improve the vulnerabilities detections such that some may get detected by open-source tool and the others may be detected by commercial tools or both [25-27]. OWASP ZAP has been selected because it is an open source and free software since it has mostly been used by previous scholars and proved to detect most web-based system vulnerabilities. Also, Acunetix, has been selected from proprietary and commercial software since it has capabilities to detect up to 6000 web-based system vulnerabilities and has been used mostly in other previous works including but are not limited to works by Elisa [12], Khoury et al. [29], El Idrissi et al. [30] and Murah & Ali [23].

2.3 Vulnerabilities Assessment Setups

OWASP ZAP version 2.11.1 and Acunetix version 11 were installed in HP EliteBook G3 with Windows 10 Pro (x64) operating system. Following the installation of OWASP ZAP, Mozilla Firefox was installed. This browser was configured with a ZAP proxy, enabling OWASP ZAP to intercept any active pages launched by OWASP ZAP onto Mozilla for the purpose of detecting vulnerabilities based on the entered target URL. The computer connected to the internet using Ethernet cable was used to perform the vulnerabilities assessment activity. In order to capture directories created utilizing AJAX technologies, OWASP ZAP vulnerability assessments were performed using the AJAX spider for target directory crawling. The test of the scanning and assessment was based on Automatic Explore mode.

Likewise, the necessary URLs of SRWBS for HLIs were created as the target in Acunetix for vulnerabilities assessment after logging into Acunetix using email address and password. After downloading and installing Acunetix (where during installation will create a user account with email and password), one should open the Acunetix web interface to find the interface for login. To initiate vulnerability assessment under Acunetix, it needs to select the target and click Scan by choosing scanning options namely Scan Type, Report Type, and Schedule. After completing the selection then Create Scan button is clicked; when the assessment is completed, it will show the report soon after it finishes the scanning.

In summary, the following steps was used in undertaking vulnerability assessment using Acunetix tool:

i. Step 1: Adding a target SRWBS Before scanning.;
ii. Step 2: Launching a Scan Now that you've added your SRWBS for the security scanning;
iii. Step 3: Reviewing Scan Results;
iv. Step 4: Integrating with Issue Tracking Tool which show detected vulnerability summary in colored boxed where you will click and view the vulnerabilities under specific category;
v. Step 5: Creating a Scan Report where you can export it in HTML or pdf.

3. Results

Figure 1 to Figure 8 display various screens of OWASP ZAP vulnerability results for HLI_A to HLI_H produced by assessing the SRWBS vulnerabilities of HLIs in Tanzania. The screen display alerts that were detected, and their further analysis were used to create a list of web-based system vulnerabilities found in HLIs SRWBS in Tanzania. For the tools setting and scanning steps on the other hand are found in Figures 15-18, in appendix section, for further clarifications.

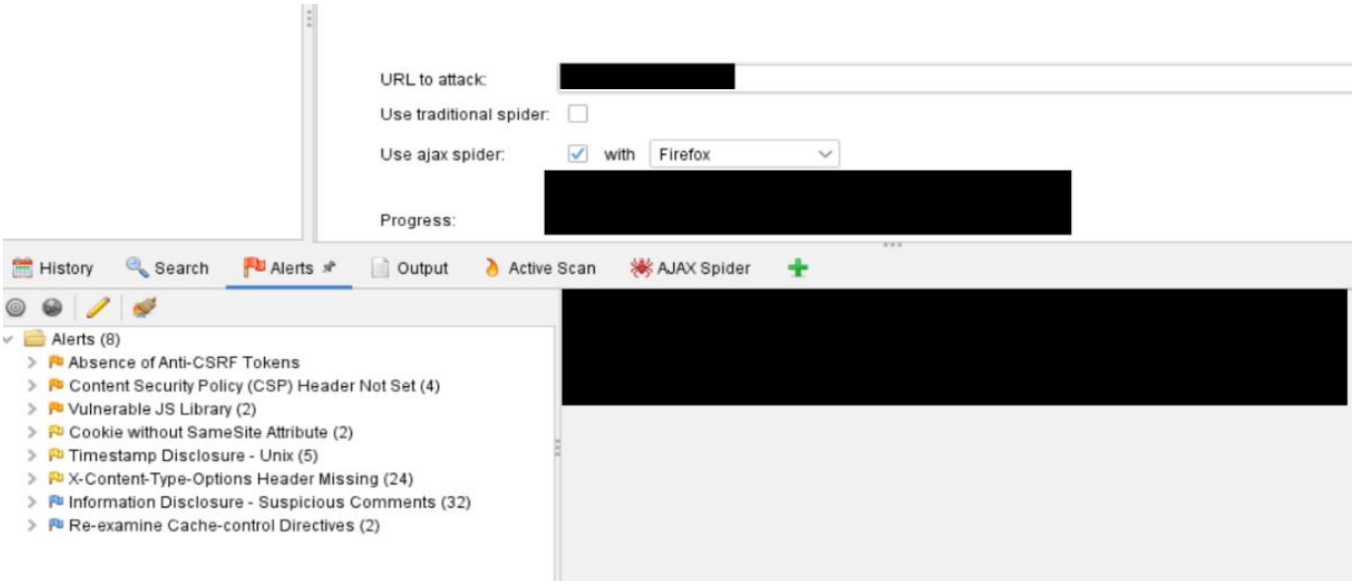

Figure 1. Results of OWASP ZAP for SRWBS of HLI_A.

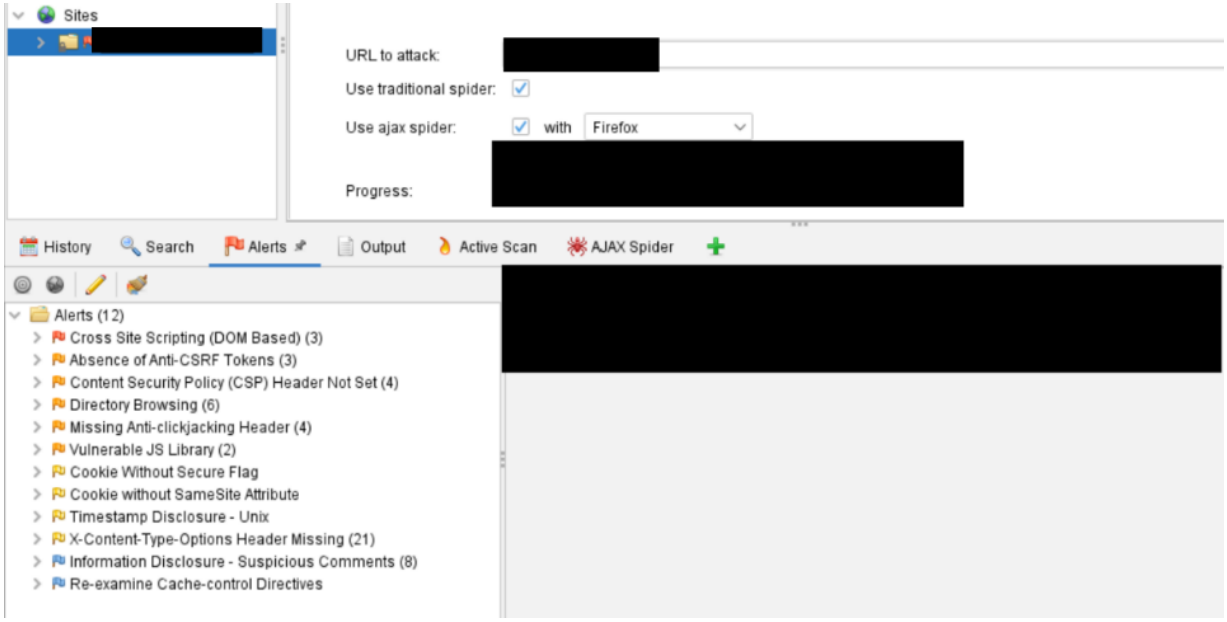

Figure 2. Results of OWASP ZAP for SRWBS of HLI_B.

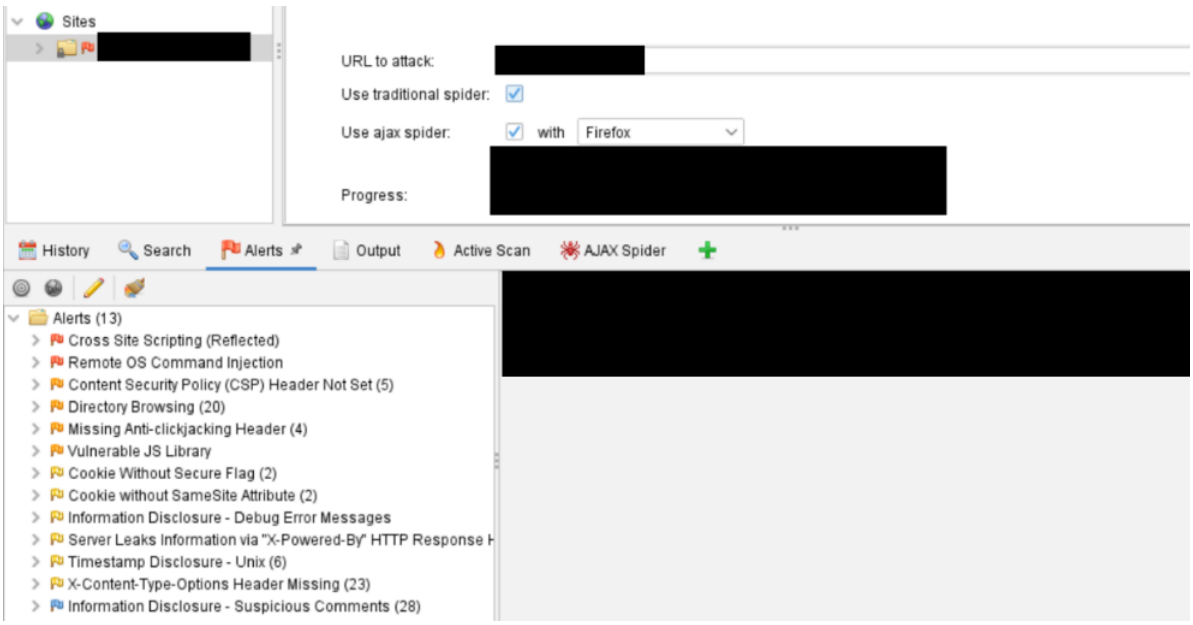

Figure 3. Results of OWASP ZAP for SRWBS of HLI_C.

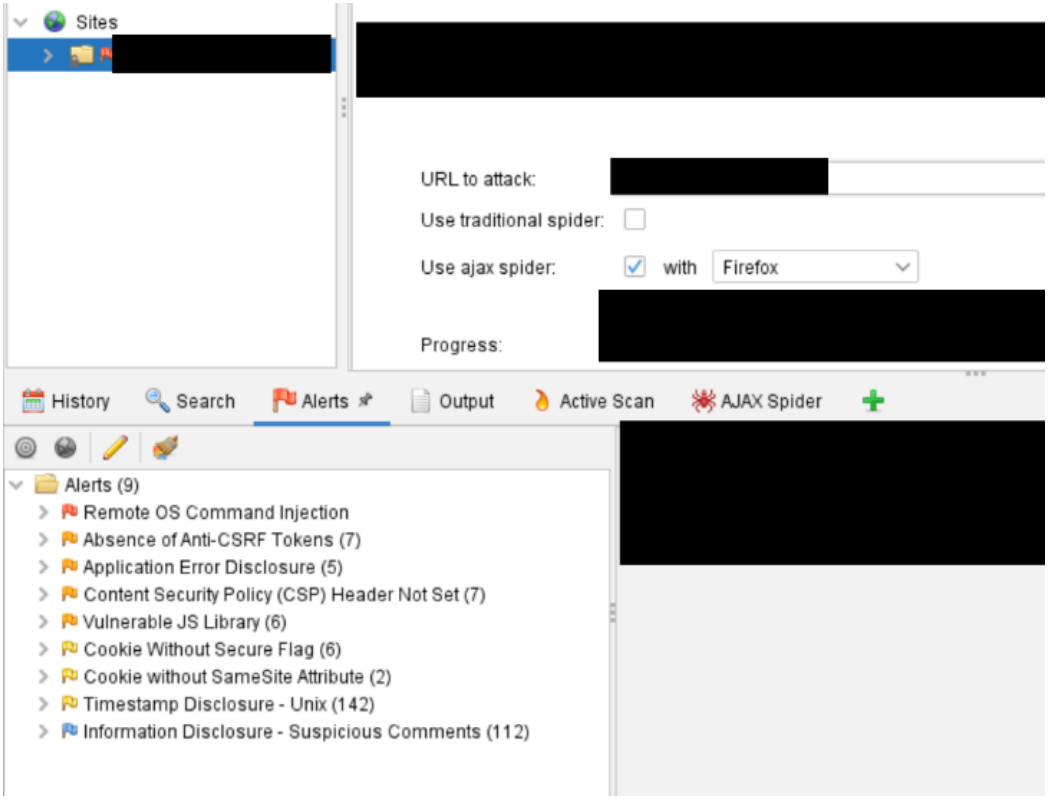

Figure 4. Results of OWASP ZAP for SRWBS of HLI_D.

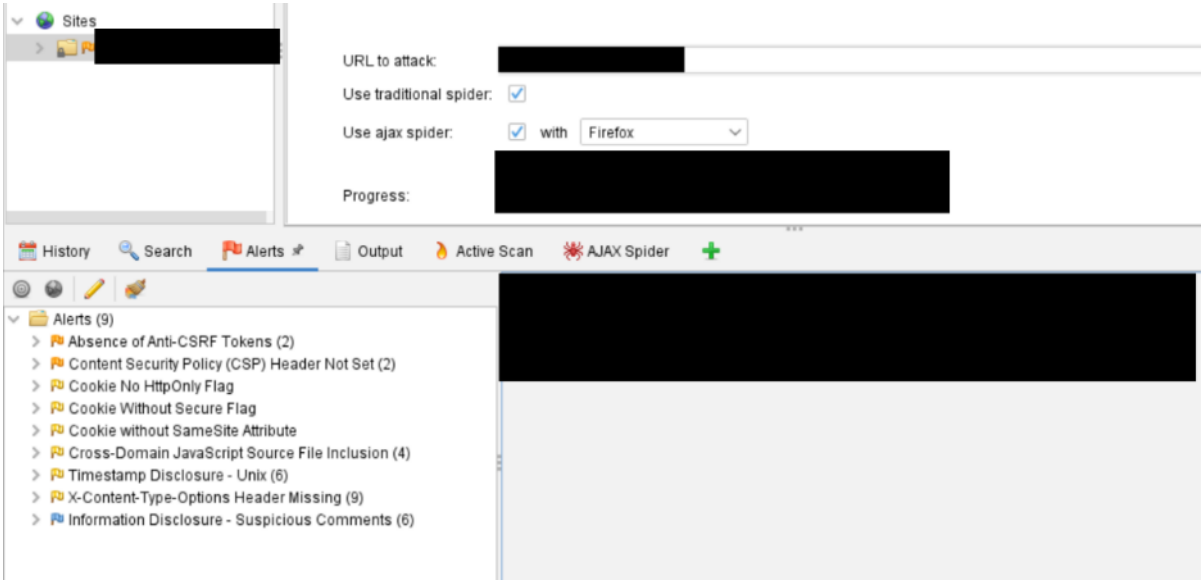

Figure 5. Results of OWASP ZAP for SRWBS of HLI_E.

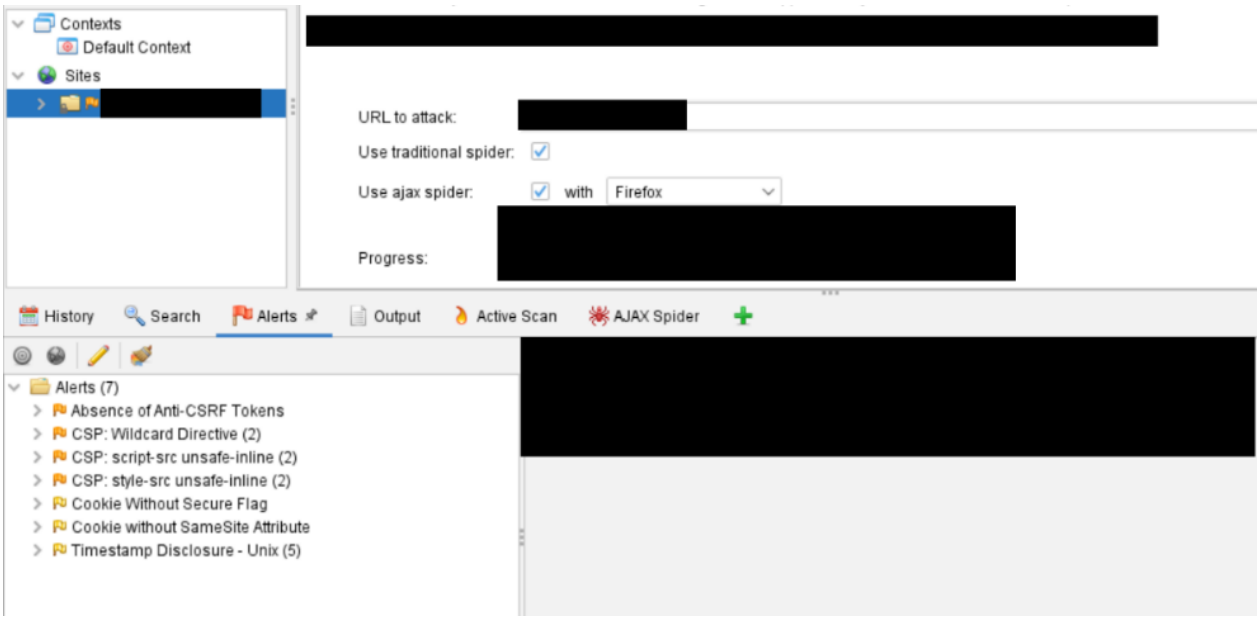

Figure 6. Results of OWASP ZAP for SRWBS of HLI_F.

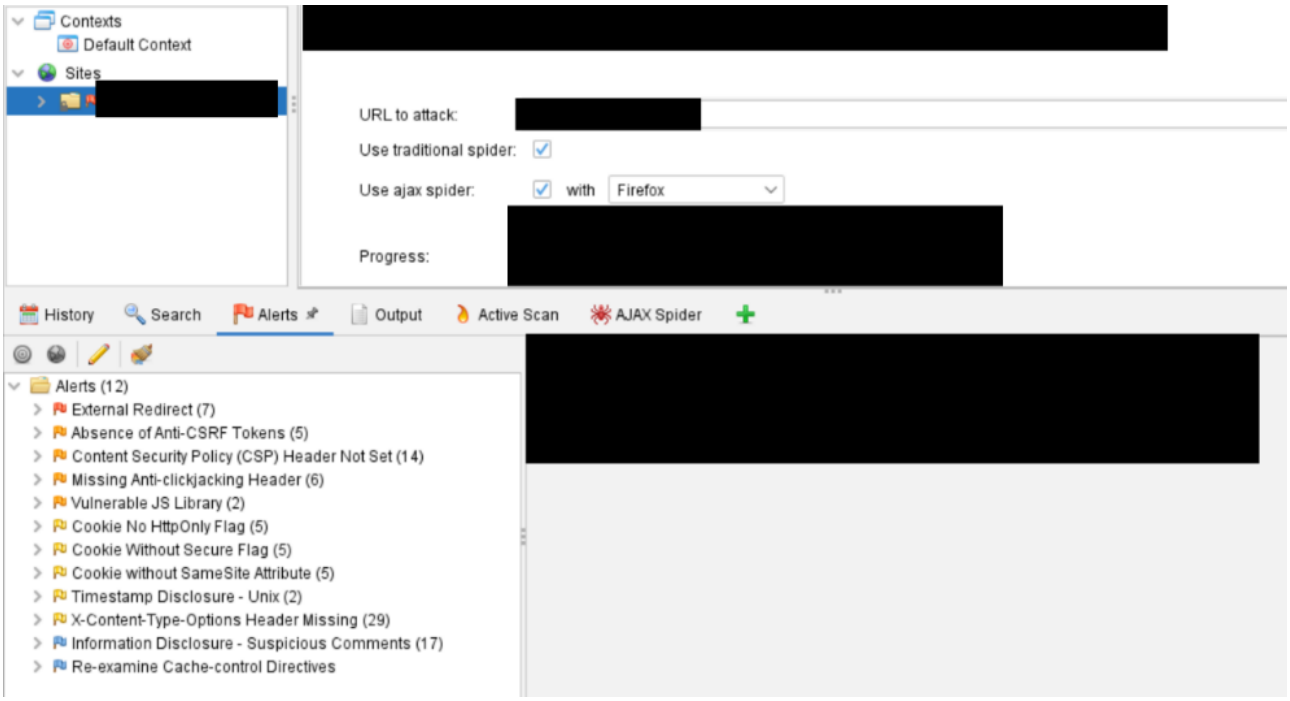

Figure 7. Results of OWASP ZAP for SRWBS of HLI_G.

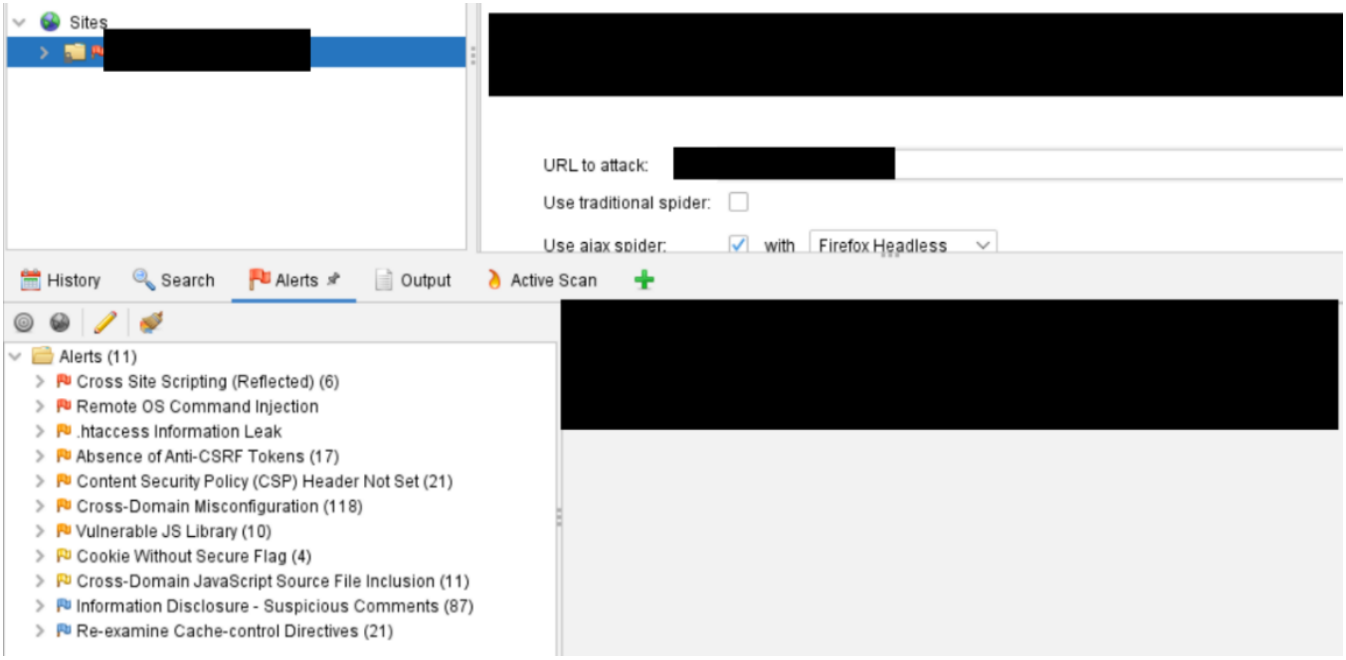

Figure 8. Results of OWASP ZAP for SRWBS for HLI_H.

3.1 OWASP ZAP Security Alerts

The OWASP ZAP web-based system vulnerability scanner was used to evaluate all HLIs SRWBS for vulnerabilities, and it appears that each HLIs SRWBS contained a unique set of security alerts, as shown in Figure 1 to Figure 8. The alerts found include Cross Site Scripting (DOM based and reflected), absence of anti-CSRF tokens, Content Security policy (CSP) header not set, Directory browsing, missing anti-clickjacking header, Vulnerable JS Library, Cookies without Secure Flag, Cookies without Same Site attribute, Timestamp Disclosure Unix, X-Content type options header missing, Information Disclosure suspicious comments, Re-examine cache control directives, Cookie no HttpOnly flag, external redirect, CSP Wildcard Directive, CSP script-src unsafe inline, CSP style-src unsafe inline, Cross Domain JavaScript Source File Inclusion, Remote OS command injection, Server Leaks, information via X-Powered-By HTTP Response header, htaccess information leak application error message disclosure and Cross Domain Misconfigurations.

Moreover, Figure 9 to Figure 12 present different screens of the Acunetix web-based system vulnerability scanner. The screens show the targets configured for vulnerability assessment and detected vulnerabilities in terms of severity levels namely High, Medium, Normal and Low.

Furthermore, Acunetix WVS detected a significant number of security flaws, as illustrated in Figure 9 to Figure 12. A list of vulnerabilities discovered throughout the evaluation process is presented by Acunetix. Some of them underwent additional analysis, with the most well-known vulnerabilities saving as representations.

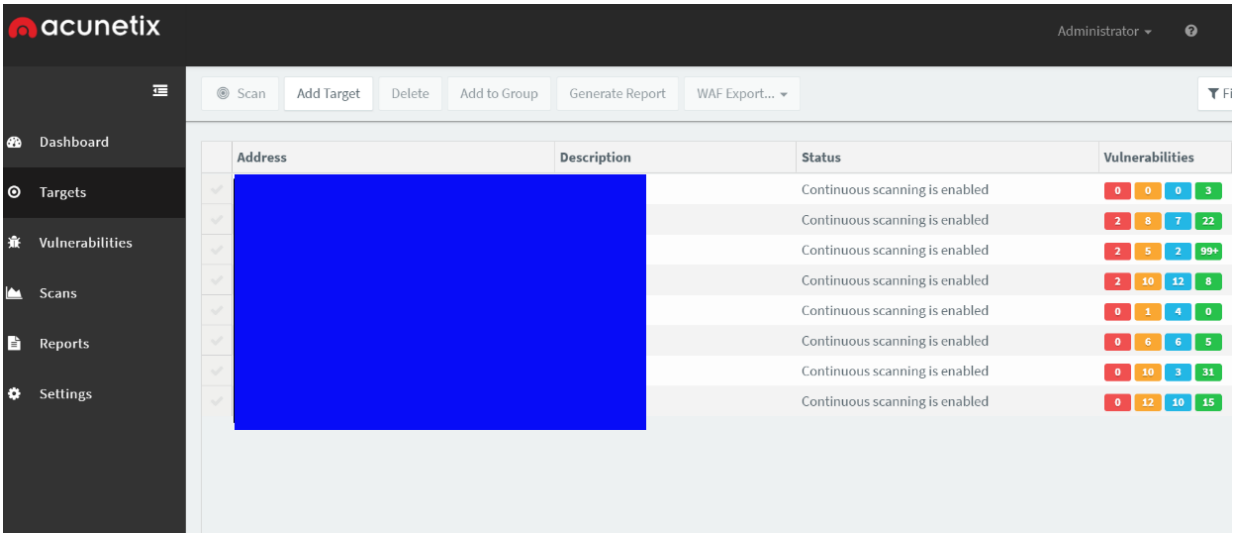

Figure 9. Configured targets for vulnerability scanning in the Acunetix scanner.

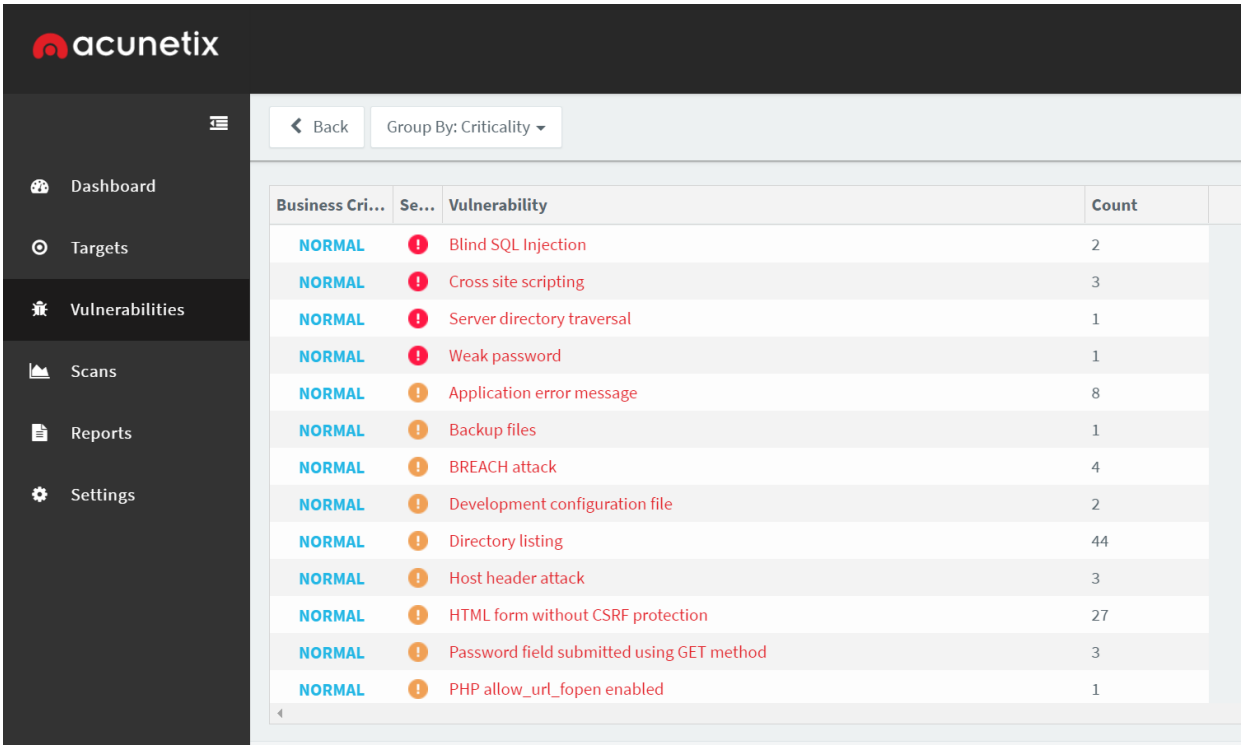

Figure 10. Critical vulnerabilities detected in SRWBS of HLIs in Tanzania.

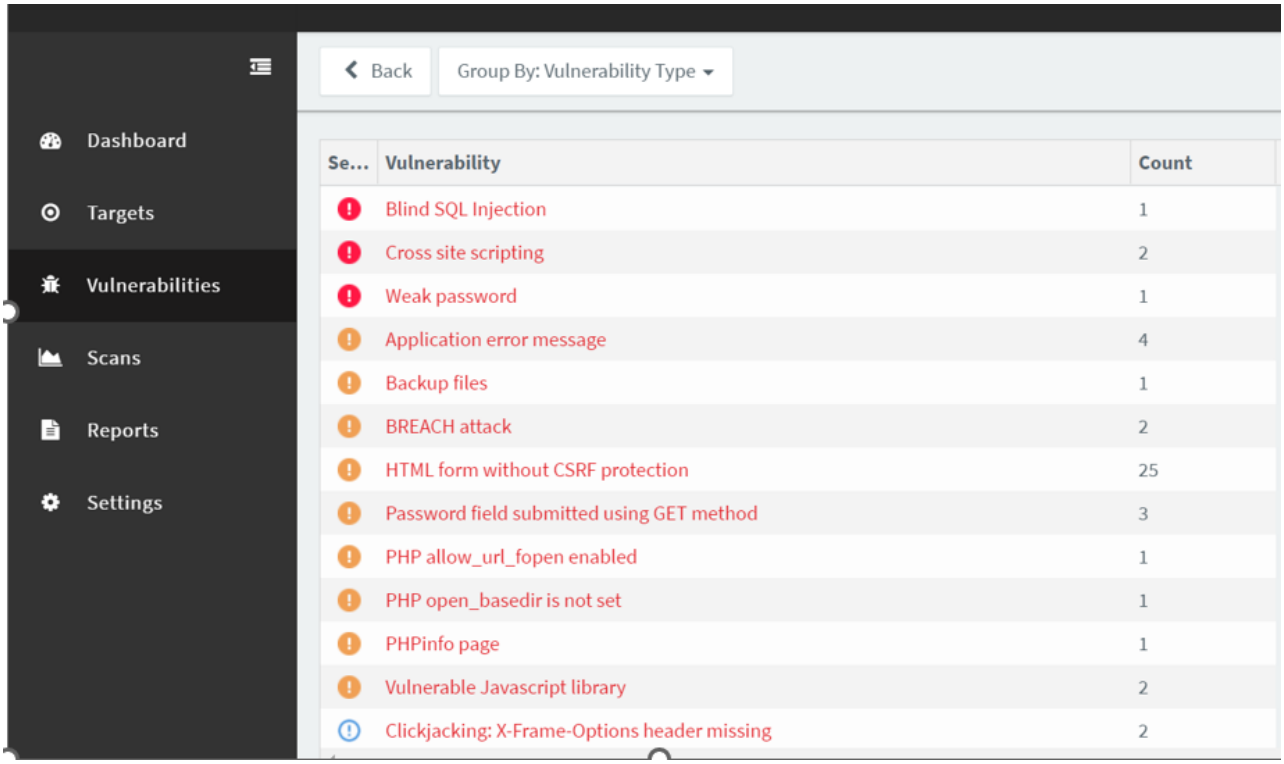

Figure 11. Detected vulnerabilities in SRWBS of HLIs in Tanzania by types (page 1).

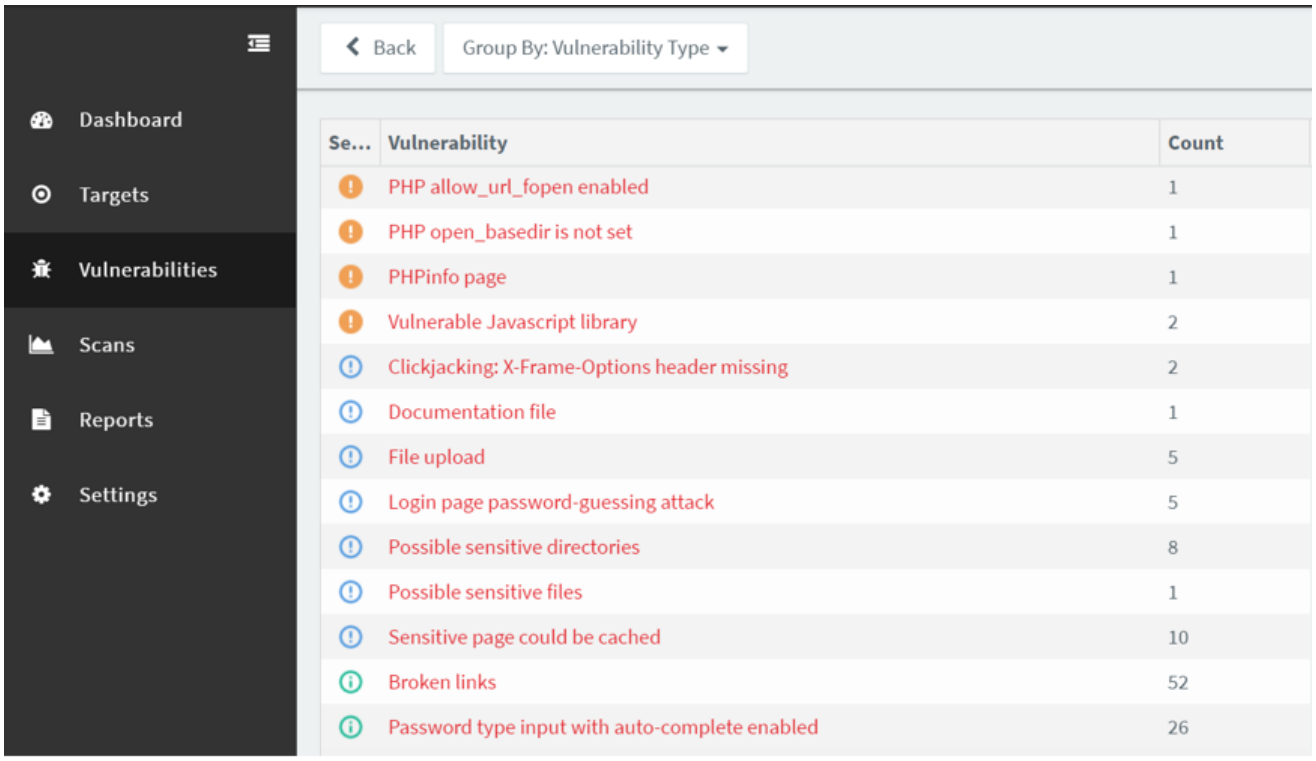

Figure 12. Detected vulnerabilities in SRWBS of HLIs in Tanzania by types (page 2).

3.2 Vulnerabilities Detected by Acunetix WVS

The Acunetix WVS identified Blind SQLi, XSS, Server directory traversal, weak passwords, application error message disclosure, backup file disclosure, BREACH attack, development configuration file disclosure, directory listing, HTML form without CSRF protection, and password field submitted using GET method as vulnerabilities.

4. Discussion

4.1 Vulnerabilities Assessment Findings

Table 2 presents the results of 28 web-based vulnerabilities from the state of the art. It consists of rows for vulnerabilities and column for HLIs SRWBS. Tick (✓) symbol was used to indicate presence of vulnerability while cross (✕) symbol was used to indicate absence of vulnerability.

The severity levels for each vulnerability detected in the HLIs in Tanzania were presented in Table 4. The severity level shows the measure of risk the vulnerabilities have to the HLIs SRWBS. The severity levels are high, medium and low. The most risk vulnerability range from critical, high, medium to the low severity level which is the least risk security level.

When the vulnerabilities in SRWBS of each HLI's were assessed using the OWASP ZAP and Acunetix vulnerability scanner, 28 vulnerabilities were found. Broken Access Control, Security Misconfiguration, Broken Authentication and Session Management and Sensitive Data Exposure were detected in all 8 HLIs SRWBS. CSRF and Vulnerable JS Libraries were detected in 7 HLIs SRWBS. CSRF and Using Components with Known Vulnerabilities appeared in 6 HLIs SRWBS. XSS have appeared in 5 HLIs SRWBS. Insecure Design, Broken Link, and, Vulnerable and Outdated Components vulnerabilities were detected in 4 HLIs SRWBS. Clickjacking, BREACH Attack and Login Page Password Guessing Attack were detected in 3 of the HLIs SRWBS. Blind SQL Injection, Directory & Path Traversal, Cross Domain Misconfiguration (CORS) and Remote OS Command Injection were detected in 2 of the HLIs SRWBS. Host header attack, Cryptographic Failures, Application Error Disclosure, File Upload, Weak Password, Software and Data Integrity Failures and External URL Redirect were detected in 1 of the HLIs SRWBS.

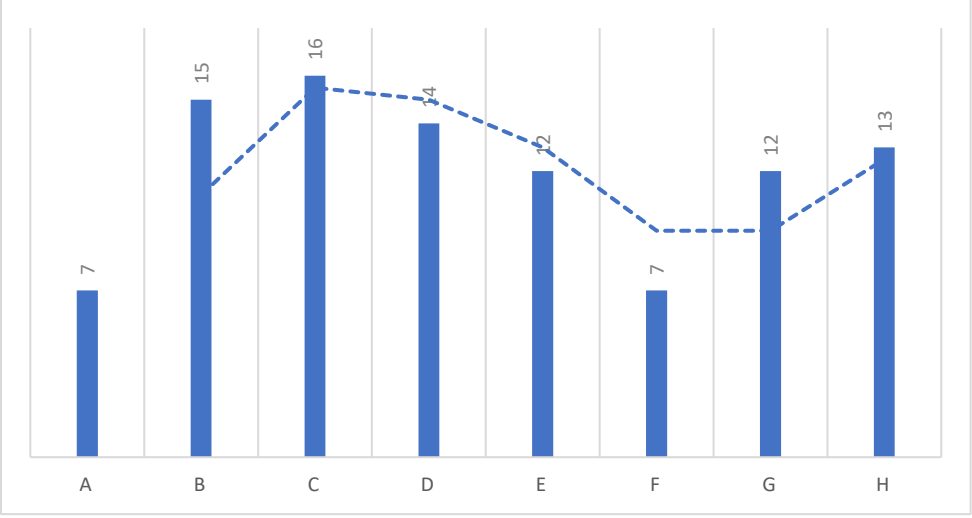

Figure 13. Number of vulnerabilities detected in SRWBS of HLIs in Tanzania.

Table 2. List of Vulnerabilities Detected in HLIs SRWBS in Tanzania.

S/N	Vulnerabilities	SRWBS of HLIs in Tanzania									
		A	B	C	D	E	F	G	H	COUNT	PERCENTAGE
1	Broken Authentication and Session Management	✓	✓	✓	✓	✓	✓	✓	✓	8	100
2	Broken Access Control	✓	✓	✓	✓	✓	✓	✓	✓	8	100
3	Security Misconfiguration	✓	✓	✓	✓	✓	✓	✓	✓	8	100
4	Sensitive Data Exposure	✓	✓	✓	✓	✓	✓	✓	✓	8	100
5	Vulnerable JS Libraries	✓	✓	✓	✓	✓	✗	✓	✓	7	87.5
6	CSRF	✓	✓	✗	✓	✗	✓	✓	✓	6	75
7	Using Components with Known Vulnerabilities	✗	✓	✓	✓	✓	✗	✓	✓	6	75
8	XSS	✗	✗	✓	✓	✓	✗	✗	✗	5	62.5
9	DOM based XSS	✗	✓	✗	✗	✗	✗	✗	✗		
10	Reflected XSS	✗	✗	✓	✗	✗	✗	✗	✓		
11	Insecure Design	✗	✓	✗	✓	✓	✓	✗	✗	4	50
12	Broken Link	✓	✓	✗	✓	✗	✗	✓	✗	4	50
13	Vulnerable and Outdated Components	✗	✓	✓	✓	✓	✗	✗	✓	4	50
14	Click Jacking	✗	✓	✓	✗	✓	✗	✗	✗	3	37.5
15	BREACH Attack	✗	✓	✓	✗	✓	✗	✗	✗	3	37.5
16	Login Page Password Guessing Attack	✗	✗	✓	✗	✓	✗	✓	✗	3	37.5
17	Remote OS Command Injection	✗	✗	✓	✓	✗	✗	✗	✓	3	37.5
18	Blind SQL Injection	✗	✗	✓	✗	✗	✗	✓	✗	2	25
19	Directory Traversal and Path Traversal	✗	✓	✓	✗	✗	✗	✗	✗	2	25
20	Cross-Domain Misconfiguration (CORS)	✗	✗	✓	✗	✗	✗	✗	✓	2	25
21	Application Error Disclosure	✗	✗	✗	✓	✗	✗	✗	✗	1	12.5
22	File Upload	✗	✗	✗	✓	✗	✗	✗	✗	1	12.5
23	Weak Password	✗	✗	✗	✗	✗	✗	✓	✗	1	12.5
24	External URL Redirect	✗	✗	✗	✗	✗	✗	✓	✗	1	12.5
25	Host header attack	✗	✗	✗	✗	✗	✓	✗	✗	1	12.5
26	Cryptographic Failures	✗	✓	✗	✗	✗	✗	✗	✗	1	12.5
27	Cross-Domain JavaScript File Inclusion	✗	✗	✗	✗	✗	✗	✗	✓	1	12.5
28	Software and Data Integrity Failures	✗	✗	✗	✗	✗	✗	✗	✓	1	12.5
Total		**7**	**15**	**16**	**14**	**12**	**7**	**12**	**13**		
Percentages		**25.1**	**51.7**	**55.1**	**48.2**	**41.3**	**25.1**	**41.3**	**44.8**		
Average		**44.2**					**37**				

Table 3. Comparison between vulnerability detected by Acunetix and OWASP ZAP.

S/N	Vulnerabilities	Status of detected vulnerability using Acunetix								Status of detected vulnerability using OWASP ZAP							
		A	B	C	D	E	F	G	H	A	B	C	D	E	F	G	H
1	Broken Authentication and Session Management	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
2	Broken Access Control	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
3	Security Misconfiguration	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
4	Sensitive Data Exposure	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
5	Vulnerable JS Libraries	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓
6	CSRF	✓	✓	✗	✓	✗	✓	✓	✓	✓	✓	✗	✓	✗	✓	✓	✓
7	Using Components with Known Vulnerabilities	✗	✓	✓	✓	✓	✗	✓	✓	✗	✓	✓	✓	✓	✗	✓	✓
8	XSS	✗	✗	✓	✓	✓	✗	✗	✗	✗	✗	✓	✓	✓	✗	✗	✗
9	DOM based XSS	✗	✓	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
10	Reflected XSS	✗	✗	✓	✗	✗	✗	✗	✓	✗	✗	✓	✗	✗	✗	✗	✓
11	Insecure Design	✗	✓	✗	✓	✓	✓	✗	✗	✗	✓	✗	✓	✓	✓	✗	✗
12	Broken Link	✓	✓	✗	✓	✗	✗	✓	✗	✓	✓	✗	✓	✗	✗	✓	✗
13	Vulnerable and Outdated Components	✗	✓	✓	✓	✓	✗	✗	✓	✗	✓	✓	✓	✓	✗	✗	✓
14	Click Jacking	✗	✓	✓	✗	✓	✗	✗	✗	✗	✓	✓	✗	✓	✗	✗	✗
15	BREACH Attack	✗	✓	✓	✗	✓	✗	✗	✗	✗	✓	✓	✗	✓	✗	✗	✗
16	Login Page Password Guessing Attack	✗	✗	✓	✗	✓	✗	✓	✗	✗	✗	✓	✗	✓	✗	✓	✗
17	Remote OS Command Injection	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✓	✓	✗	✗	✗	✓
18	Blind SQL Injection	✗	✗	✓	✗	✗	✗	✓	✗	✗	✗	✓	✗	✗	✗	✓	✗
19	Directory Traversal and Path Traversal	✗	✓	✓	✗	✗	✗	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗
20	Cross-Domain Misconfiguration (CORS)	✗	✗	✓	✗	✗	✗	✗	✓	✗	✗	✓	✗	✗	✗	✗	✓
21	Application Error Disclosure	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗
22	File Upload	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗
23	Weak Password	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗
24	External URL Redirect	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✓	✗
25	Host header attack	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗
26	Cryptographic Failures	✗	✓	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
27	Cross-Domain JavaScript File Inclusion	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✓
28	Software and Data Integrity Failures	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✓

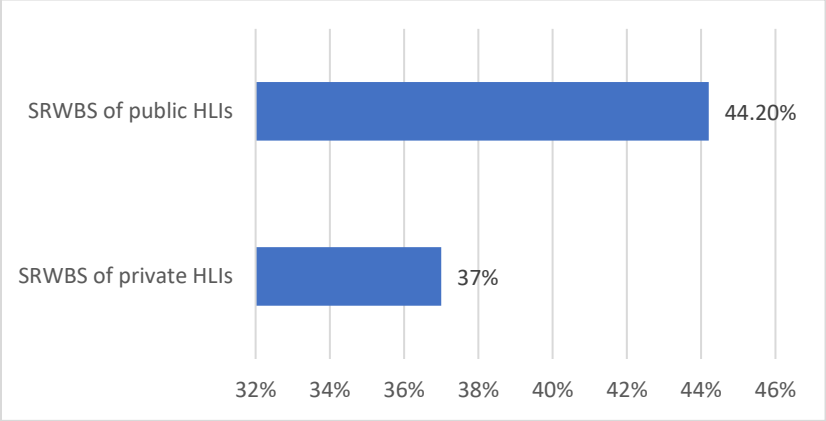

Figure 14. Average percentage of SRWBS vulnerabilities for public and private HLIs in Tanzania

Table 3 presents the comparison of vulnerability detected by Acunetix and OWASP ZAP tools. It shows that most vulnerabilities were detected by each tool in the SRWBS of the 8 HLIs. This indicates that significant attention was given to cross-validation during the vulnerability assessment, as the results obtained from the Acunetix tool were validated using the OWASP ZAP tool. Only few SRWBS of HLIs resulted into different vulnerability detection between Acunetix tool and OWASP ZAP tool. For instance, in HLI_D and HLI_H, Acunetix tool failed to detect the remote command execution vulnerability where OWASP ZAP tool managed to detected it. Also, HLI_G Acunetix managed to detect weak password vulnerability while OWASP ZAP failed to detect it.

In the practice of vulnerability assessment, cross-validation using multiple tools is essential for ensuring a comprehensive and reliable security evaluation. By employing both Acunetix and OWASP ZAP, we managed to enhance the detection accuracy and coverage of potential vulnerabilities within SRWBS system. During the assessment under this study, it was found that most vulnerabilities were detected by both tools, highlighting their effectiveness and alignment in identifying security issues. However, OWASP ZAP identified one additional vulnerability (remoted command execution) that was not detected by Acunetix where Acunetix identified one additional vulnerability (weak password) that was not detected by OWASP ZAP. This discrepancy emphasizes the value of cross-validation, as relying on a single tool could result in overlooked vulnerabilities. Therefore, the integration of Acunetix and OWASP ZAP in a cross-validation approach provides a more thorough and dependable assessment, reinforcing the overall security posture of the system.

The findings from this cross-validation exercise also underscore the unique strengths of each tool. Acunetix is known for its comprehensive scanning capabilities and user-friendly interface, making it a popular choice for many organizations. OWASP ZAP, on the other hand, is renowned for its flexibility and extensive customization options, which can be particularly advantageous in identifying complex or less common vulnerabilities. The fact that OWASP ZAP and Acunetix both detected an additional vulnerability demonstrates how these tools can complement each other. By leveraging the strengths of both tools, organizations can achieve a more holistic understanding of their security landscape. This layered approach not only improves the detection rate but also enhances the overall resilience of the system against potential threats. Consequently, incorporating multiple tools into the vulnerability assessment process is a prudent strategy for maintaining a robust security posture.

Table 4. SRWBS of HLIs in Tanzania vulnerabilities and their severity levels.

	Vulnerabilities	SRWBS of HLIs in Tanzania															
		A		B		C		D		E		F		G		H	
		Status	Severity	Status	Severity	Status	Severity	Status	Severity	Status	Severity	Status	Severity	Status	Severity	Status	Severity
1	Broken Authentication and Session Management	✔	Low	✔	Low	✔	low	✔	Low	✔	Low	✔	Low	✔	Low	✔	Low
2	Broken Access Control	✔	Low	✔	Low	✔	Medium	✔	Low	✔	Low	✔	Low	✔	Low	✔	Low
3	Security Misconfiguration	✔	Medium	✔	Medium	✔	Medium	✔	Medium	✔	Medium	✔	Medium	✔	Medium	✔	Medium
4	Sensitive Data Exposure	✔	Medium	✔	Medium	✔	Medium	✔	Medium	✔	Low	✔	Medium	✔	Medium	✔	Low
5	Vulnerable JS Libraries	✔	Medium	✔	Medium	✔	High	✔	Medium	✔	Medium	✘	-	✔	High	✔	Medium
6	CSRF	✔	High	✔	High	✘	-	✔	High	✘	-	✔	High	✔	High	✔	High
7	Using Components with Known Vulnerabilities	✘	-	✔	Medium	✔	Medium	✔	Medium	✔	Medium	✘	-	✔	Medium	✔	Medium
8	XSS	✘	-	✘	-	✔	Critical	✔	High	✔	high	✘	-	✘	-	✘	-
9	DOM based XSS	✘	-	✔	Critical	✘	-	✘	-	✘	-	✘	-	✘	-	✘	-
10	Reflected XSS	✘	-	✘	-	✔	Critical	✘	-	✘	-	✘	-	✘	-	✔	High
11	Insecure Design	✘	-	✔	Medium	✘	-	✔	Medium	✔	Medium	✔	Medium	✘	-	✘	-
12	Broken Link	✔	Low	✔	Low	✘	-	✔	Low	✘	-	✘	-	✔	Low	✘	-
13	Vulnerable and Outdated Components	✘	-	✔	Medium	✔	Medium	✔	Medium	✔	Medium	✘	-	✘	-	✔	Medium
14	Click Jacking	✘	-	✔	Medium	✔	Medium	✘	-	✔	Medium	✘	-	✘	-	✘	-
15	BREACH Attack	✘	-	✔	High	✔	High	✘	-	✔	High	✘	-	✘	-	✘	-
16	Login Page Password Guessing Attack	✘	-	✘	-	✔	Medium	✘	-	✔	Medium	✘	-	✔	Medium	✘	-
17	Remote OS Command Injection	✘	-	✘	-	✔	High	✔	High	✘	-	✘	-	✘	-	✔	High
17	Blind SQL Injection	✘	-	✘	-	✔	Critical	✘	-	✘	-	✘	-	✔	Critical	✘	-
18	Directory Traversal and Path Traversal	✘	-	✔	Medium	✔	Critical	✘	-	✘	-	✘	-	✘	-	✘	-
19	Cross-Domain Misconfiguration (CORS)	✘	-	✘	-	✔	Medium	✘	-	✘	-	✘	-	✘	-	✔	High
20	Application Error Disclosure	✘	-	✘	-	✘	-	✔	Medium	✘	-	✘	-	✘	-	✘	-
21	File Upload	✘	-	✘	-	✘	-	✔	Medium	✘	-	✘	-	✘	-	✘	-
22	Weak Password	✘	-	✘	-	✘	-	✘	-	✘	-	✘	-	✔	Critical	✘	-
23	External URL Redirect	✘	-	✘	-	✘	-	✘	-	✘	-	✘	-	✔	High	✘	-
24	Host header attack	✘	-	✘	-	✘	-	✘	-	✘	-	✔	High	✘	-	✘	-
25	Cryptographic Failures	✘	-	✔	Medium	✘	-	✘	-	✘	-	✘	-	✘	-	✘	-
26	Cross-Domain JavaScript File Inclusion	✘	-	✘	-	✘	-	✘	-	✘	-	✘	-	✘	-	✔	Low
27	Software and Data Integrity Failures	✘	-	✘	-	✘	-	✘	-	✘	-	✘	-	✘	-	✔	Low
28	Software and Data Integrity Failures	✘	-	✘	-	✘	-	✘	-	✘	-	✘	-	✘	-	✔	Low

The vulnerability assessment findings show that 100% of selected SRWBS for HLIs in Tanzania are susceptible to Broken Access Control, Security Misconfiguration, Broken Authentication and Session Management and Sensitive Data Exposure. Also, the findings show that 87.5% of selected SRWBS for HLIs in Tanzania are susceptible to Vulnerable JS Libraries while 75% of SRWBS for HLIs in Tanzania are susceptible to CSRF and Using Components with Known Vulnerabilities. Moreover, the findings show that 62.5% of selected SRWBS for HLIs in Tanzania are susceptible to XSS vulnerabilities. In addition to that, the findings revealed that 50% of the selected SRWBS for HLIs in Tanzania are susceptible to Vulnerable and Outdated Components vulnerabilities, Broken Link and Insecure Designs. The findings also show that 37.5% of chosen SRWBS for HLIs are susceptible to Clickjacking, BREACH Attack, Remote OS Command Injection and Login Page Password Guessing Attack. Furthermore, the findings show that 25% of selected SRWBS for HLIs are susceptible to Cross Domain Misconfiguration (CORS), Blind SQL Injection and Directory and Path Traversal vulnerabilities. Last but not least, the findings show that 12.5% of selected SRWBS for HLIs are susceptible to Cryptographic Failures, Application Error Disclosure, File Upload, Weak Password, External URL Redirect, Software and Data Integrity Failures, Host header attack and Cross-Domain JavaScript File Inclusion.

From Figure 14, the study found that public SRWBS of HLI in Tanzania are more susceptible to vulnerabilities than private SRWBS of HLI in Tanzania. On average, the public SRWBS were found susceptible to vulnerabilities by 44.2% while private SRWBS were found susceptible to vulnerabilities by 37%. This implies that public SRWBS have 44.2% chance to be found with web-based system vulnerabilities, while private SRWBS have 37% chance to be found with web-based system vulnerabilities.

Despite the SRWBS of public HLIs to be more vulnerable than SRWBS of private HLI by average percentages, the SRWBS of HLIs_A from public HLIs was among the two HLIs with lowest number (7) of vulnerabilities detected as shown in Figure 13. The reason has been attributed to its strong cybersecurity posture where the HLIs has a dedicated cybersecurity unit with cybersecurity specialists together with highly trained developers with considerations of proper secure software development skills.

Also, the study found that 50% of selected SRWBS for HLIs are vulnerable to XSS, 25% are vulnerable to SQLi, 75% of selected SRWBS for HLIs are vulnerable to CSRF, 37.5% are vulnerable to OS command injection, 25% are vulnerable to Directory and Path Traversal and, 100% of the selected SRWBS for HLIs are vulnerable to Security misconfigurations and, Broken authentication & Session management vulnerabilities. Also, 1% of chosen SRWBS for HLIs are vulnerable to File Inclusion vulnerability.

The findings under this study are the generalizations of web-based system vulnerabilities for SRWBS of HLIs in Tanzania. The findings are specific for SRWBS of HLIs environments in Tanzania. Hence, this study provides general findings of what web-based system vulnerabilities exist in SRWBS of HLIs in Tanzania. With the reported 28 vulnerabilities detected from SRWBS of HLIs in Tanzania, HLIs in Tanzania may assess their SRWBS against these vulnerabilities in order to improve their SRWBS security.

Vulnerability assessment of website in Bangladesh had assessed 34 websites and, out of them, 19 were found vulnerable and 15 were secure [25]. Comparing this study result with our findings,

it shows that security is a challenging issue in web-based system in both developing and developed countries and more efforts are required to address it.

Study conducted by Helmiawan et al. [31] show that the heart of the web of organizations and universities that do research depends on the individual SQL database, and thus there is a high risk and there is a medium risk, but it is not easy to secure SQL database. Hackers often will damage the SQL database to perform copy, edit and delete the database. This should be taken as a turning point to improve the security of SRWBS in Tanzanian HLIs against SQLi to secure and ensure safety of SRWBS databases and potential data.

Also, the study align with work of Ulven [32] which conducted a Systematic Review of Cybersecurity Risks in Higher Education across the globe. The study showed most HLIs have cyber security risks in which hacking risk caused by vulnerabilities in HLIs systems are the root cause. And thus, assessment of SRWBS vulnerabilities for HLIs in Tanzania was necessary in understanding of top vulnerabilities and recommending mitigation measures.

4.2 Evolving Cybersecurity Challenges facing HLIs in Tanzania based on the Vulnerability Assessment

From the detected vulnerabilities obtained by vulnerability assessment, HLIs in Tanzania are facing various cyber security challenges such as the use outdated software. This is among the most common threats that HLIs are facing in their systems. The consequences on using outdated software can be severe, including data breaches, financial loss, and reputational damage.

Additionally, third-party vulnerabilities are a concern. This is due to the fact that many of the development frameworks, libraries, supportive programming engines, and plugins used are created by third-party vendors. These may contain vulnerabilities that HLI developers may find difficult to address promptly [30-31].

Moreover, by analyzing the detected vulnerabilities, lack of cyber security awareness, lack of cyber security policy, lack of cyber security units and shortage of financial and human resource budget may also be the major challenges available in HLIs which should be addressed to improve the security of HLIs systems. These challenges are linked due to the fact that no any SRWBS was found secure from vulnerabilities and among the challenges causing vulnerabilities are these that have been mentioned [14, [32-33].

4.3 Impact of Vulnerabilities in SRWBS for HLIs in Tanzania

Web-based system vulnerabilities have caused various worse impacts in education sectors and thus in SRWBS as well. For instance, SQLi vulnerability may result in cyber-attacks which may cause unauthorized deletion, modification or copy of SRWBS databases and compromise the integrity of SRWBS. Also, XSS, CSRF and SQLi may enable attackers to gain unauthorized access to SRWBS accounts and stealing of SRWBS data. Exploitation of SQLi and XSS may enable attacker to modify SRWB students and staff information, such as results, financial records, and registrations status [3, 37].

Also, presence of weak cryptographic protocol may enable attacker to read confidential information on transit or at rest in the SRWBS. Moreover, vulnerabilities, such as using components with known vulnerabilities and using outdated components, may expose the system to attackers and help them to gain unauthorized access and even installation of malicious files in the SRWBS infrastructures. Furthermore, vulnerabilities may affect the reputation of the HLIs especially when the attacker will replace

SRWBS system pages with bad contents, such as phonograph or insulting contents [1] .

Last but not least, Vulnerabilities in SRWBS of HLIs in Tanzania has a lot of harmful impacts. SRWBS should address each vulnerabilities using a proper way so as to ensure that attackers are not raising any exploits on them and causing such a harmful impact written in this paper and even those that have not been written.

4.4 Mitigation Measures for the Detected Vulnerabilities in SRWBS for HLIs in Tanzania

Before presenting recommendations for mitigation measures of web-based system vulnerabilities in SRWBS of HLIs in Tanzania, the causes of the 28 detected vulnerabilities were presented in the Table 5. The presented causes were derived from the expert knowledge and observation of reports in the vulnerability scanning tools used during vulnerability assessment [23, 35–41].

Table 5. Causes of vulnerabilities detected in SRWBS of HLIs in Tanzania.

S/N	Vulnerabilities	Causes
1	Broken Authentication and Session Management	• poorly implemented authentication and session management functions. • username and password haven't been invalidated adequately during logout
2	Broken Access Control	• weak or non-implementation of access control in the target application • developers fail to restrict access to certain functions based on a user's role
3	Security Misconfiguration	• Occurs when services are deployed by developers / server administrators with insecure default settings
4	Sensitive Data Exposure	• Misconfiguration errors. • No encryption or weak encryption • Insecure passwords • Unsecure webpages (lack of SSL/TLS) • Providing excessive permissions to users who don't need them and a lack of visibility into who has access to what files, empowers users to access and share data without any accountability
5	Vulnerable JS Libraries	• when a web application uses outdated or unpatched JavaScript libraries
6	CSRF	• when a web server receives a malicious request from a trusted browser without authenticating the request with a security token.
7	Using Components with Known Vulnerabilities	• coding errors, • design flaws, or • configuration issues in the component
8	XSS	• Executing unvalidated user inputs in application • Output to the browser not properly escaped before being displayed
9	DOM based XSS	• Executing unvalidated user inputs in application
10	Reflected XSS	• Output to the browser not properly escaped before being displayed
11	Insecure Design	• Failure to integrate security during application designs
12	Broken Link	• deleting or renaming pages without updating the links to them, • migrating your site to a new platform or domain without proper redirection, and • typing errors or misspellings in the link URL or anchor text.
13	Vulnerable and Outdated Components	• using software components that are no longer being supported for development.

S/N	Vulnerabilities	Causes
14	Click Jacking	• Use of HTML frames or iframes without proper security.
15	BREACH Attack	• Weak and Stolen Credentials. Compromised passwords
16	Login Page Password Guessing Attack	• Weak and Stolen Credentials. Compromised passwords
17	Remote OS Command Injection	• when an application passes unsafe (unvalidated) user supplied data (forms, cookies, HTTP headers etc.) to a system shell
18	Blind SQL Injection	• insufficient input validation to filter SQL statements in input fields.
19	Directory Traversal and Path Traversal	• when the attacker is able to read files on the web server outside of the directory of the website due to lack of developer restrictions.
20	Cross-Domain Misconfiguration (CORS)	• improper configuration of CORS headers
21	Application Error Disclosure	• improper error handling for example, when developer failed to restrict display of pages in application that contain too verbose error messages, with potential source code disclosure or other sensitive information like the internal web server configuration, credentials of API keys, resource's location or any other user's data during occurrences of application errors.
22	File Upload	• Insufficient input file validation
23	Weak Password	• Use of dictionary words and phrases as password
24	External URL Redirect	• Insipient user input validation for example when an application allows a user to control a redirect or forward to another URL
25	Host header attack	• Inadequate host header validation
26	Cryptographic Failures	• When no encryption or use of weak encryption algorithms
27	Cross-Domain JavaScript File Inclusion	• when your web application loads JavaScript files from an external domain without proper validation
28	Software and Data Integrity Failures	• Insufficient access control when an attacker can modify or delete data in an unauthorized manner.

The following measures can be taken by HLIs to mitigate the detected vulnerabilities so as to improve the security of their SRWBS and they should work on the vulnerabilities in the order of their severity levels from critical to low.

Developers should conduct input sanitizations to address XSS and SQLi vulnerabilities. SRWBS developers should sanitize inputs entered by SRWBS users in the SRWBS to make sure no malicious input is processed and executed in the SRWBS. Instead, they should sanitize and filter input before processing and ensure clean inputs are supplied to the SRWBS [38, 40-44].

Server managers should perform proper and secure configuration of operational infrastructures, such as the webserver, database server and the frameworks and libraries of the SRWBS. This study has shown that most of SRWBS and their infrastructures are not well configured in such a way there are vulnerabilities related to misconfigurations [5, 7, 16]. The configurations task should be well handled by both server administrators and SRWBS developers.

Developers should Update hardware, software, libraries and plugins for the SRWBS of HLIs to mitigate vulnerabilities caused by use of outdated software and the use of components with known vulnerabilities. The use of outdated components has contributed to vulnerabilities old software, libraries and plugins. Replacing old hardware and regularly updating software, libraries and plugins in web server and SRWBS frameworks will help to mitigate these vulnerabilities [16, 48]. This will also help to avoid using components with known vulnerabilities in SRWBS operational infrastructures.

Developers should enable anti-CSRF token in SRWBS source codes to prevent SRWBS against CSRF vulnerabilities. Some of the SRWBS have been found with CSRF vulnerabilities. The use of web development framework, such as Laravel, Yii, Code ignitor and Django both support use of anti-CSRF token in protecting the systems from CSRF vulnerabilities. Developers should make sure they enable this feature in order to utilize its benefit in defending their SRWBS against the CSRF vulnerability [2], [7], [41], [49].

Developers should ensure proper error handling to avoid leaking sensitive data to unauthorized users. Some of the SRWBS display error which leaks sensitive information about the SRWBS backend. Proper error handling will help to mitigate this vulnerability [10] .

Developers and server managers should employ standard recommended secure cryptographic protocols to ensure confidentiality of data at rest and in transit of the SRWBS. It has been seen that some of the SRWBS are using weak cryptographic protocols in encrypting their sensitive data [47-48]. Developers should use the current strong cryptographic protocols recommended by regulatory authorities such as eGa (Tanzania electronic government agency), TZ-CERT (Tanzanian Computer Emergency Response Team), ITU (International Standard Organizations) and other International Security agencies.

Developers should implement strong access control policy, which include the use of complex and strong unique password, and may include the use of two factor authentications in critical user accounts. Some of the SRWBS have been found with weak password vulnerability [49-50]. Developers should enforce complex and strong password as per the good password standards

Developers should include codes to prevent clickjacking by writing codes that instruct browsers via HTTP headers on frame killings. Clickjacking occurs when a popup window with a button appears on the browser where in the button another button is launched by attackers to redirect user to malicious site or in forcing user to enable malicious activities. By including codes to prevent clickjacking attack the SRWBS will be assured safe and secure [10], [39-40].

The SRWBS should be redesigned following secure software design and secure architecture to mitigate vulnerabilities caused by insecure design [54]. All SRWBS should be developed using latest development frameworks using the latest stable programming language versions available. Some of SRWBS have been developed using web development frameworks but not latest thing that had caused existence of vulnerabilities in their systems.

Additionally, frequent cybersecurity trainings should be provided to the Tanzanian HLIs' custodians of the SRWBS in order to keep them up to date on the most recent security threats. This includes, but is not limited to, developers, system administrators, server managers, and security specialists and all system users including students and instructors [14, 46, 52-53]. It will also help to mitigate vulnerabilities in the SRWBS of the HLIs in Tanzania and raise security awareness among ICT employees in relation to those systems.

Moreover, HLIs in Tanzania should regularly conduct vulnerability assessment in their SRWBS include code reviews to ensure that vulnerabilities are identified and mitigated properly [2], [20], [54-55]. Conducting regularly vulnerability assessment and penetration testing will help to address vulnerabilities and assure that the SRWBS are secure and well protected from cybersecurity risks

5. Conclusion and Recommendations

In this study, the selected SRWBS of the 8 HLIs in Tanzania have been assessed using OWASP ZAP and Acunetix vulnerability scanners. The assessments were conducted using automatic mode where 28 vulnerabilities, such as Broken Authentication and Session Management, SQLi, XSS, OS command injection and CSRF, were detected with different severity levels ranging from critical, high, medium and low.

Taking the consideration of the selected SRWBS of the 8 HLIs in Tanzania, it has been

shown that most of the HLIs have not yet taken the issue of security as a serious concern since none of the HLIs have been found free from vulnerabilities. It has been observed that SRWBS of private HLIs in Tanzania are less vulnerable with an average of 37% than those of the public HLIs which are vulnerable by average of 44.2%.

The SRWBS of private and public HLIs in Tanzania should take a concern of secure software development, which involves security consideration during planning phase, requirement phase, design, coding phase, testing phase, deployment and maintenance phase. Taking this into consideration will mitigate most of vulnerabilities at large since every phase of development will take care of security at large.

As the number of essential systems in HLIs and other areas in Education sector increase in Tanzania, the future work is to assess vulnerabilities of admission systems for private and public HLIs integrated into TCU and NACTE (National Council of Technical Education) so as to come up with generalized results of vulnerabilities in admission systems in Tanzania. This will help to improve the security of admission system for HLIs in Tanzania.

Furthermore, an innovative methodology that can be undertaken in the future in place of black-box testing is the utilization of machine learning (ML) and artificial intelligence (AI) techniques.

This approach involves training ML models on large datasets of known vulnerabilities and attack patterns to develop robust testing frameworks. These ML-driven testing frameworks can autonomously identify potential vulnerabilities in SRWBS and other software systems without relying solely on predefined test cases. By continuously learning from new data and adapting to evolving threat landscapes, ML-based black-box testing can provide more comprehensive and accurate assessments of system security. Furthermore, the integration of reinforcement learning (RL) algorithms can enable testing agents to dynamically adjust their testing strategies based on feedback received during testing iterations.

Acknowledgement
Authors pay special thanks to the University of Dodoma management for supporting this work.

CONTRIBUTIONS OF CO-AUTHORS

Wilbard G. Masue	[ORCID: 0000-0001-6333-9456]	Conceived the idea and wrote the paper
Daniel Ngondya	[ORCID: 0000-0003-4267-6351]	Conducted review and language editing
Tabu S. Kondo	[ORCID: 0000-0002-0222-4951]	Conducted review and language editing

Appendix

Figure 15. Acunetix login page.

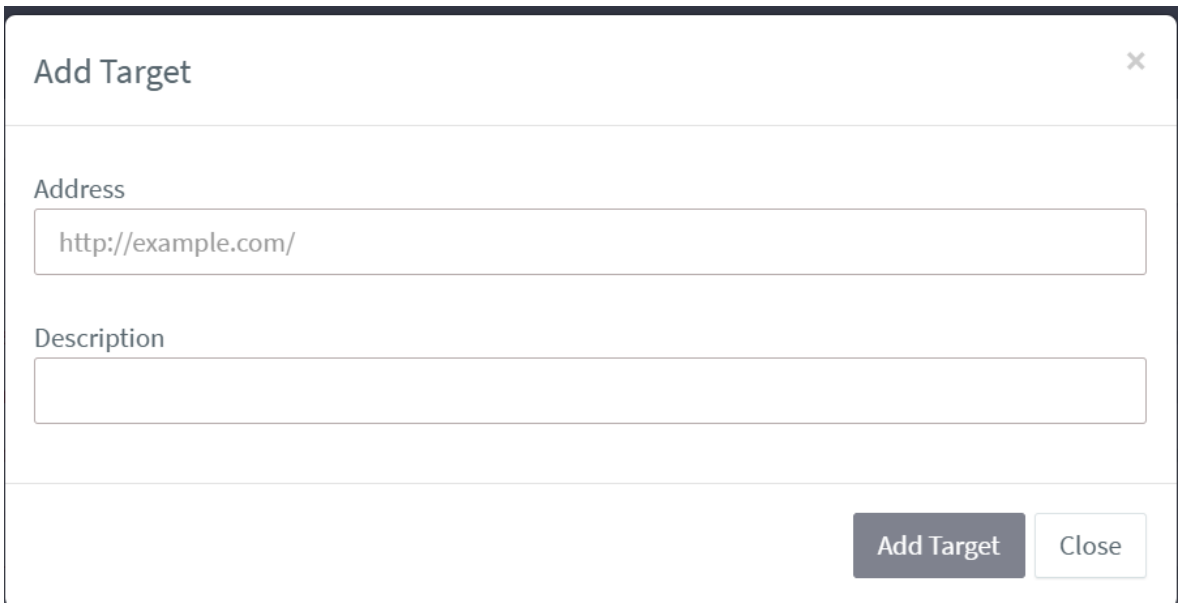

Figure 16. Screen for creating target in Acunetix.

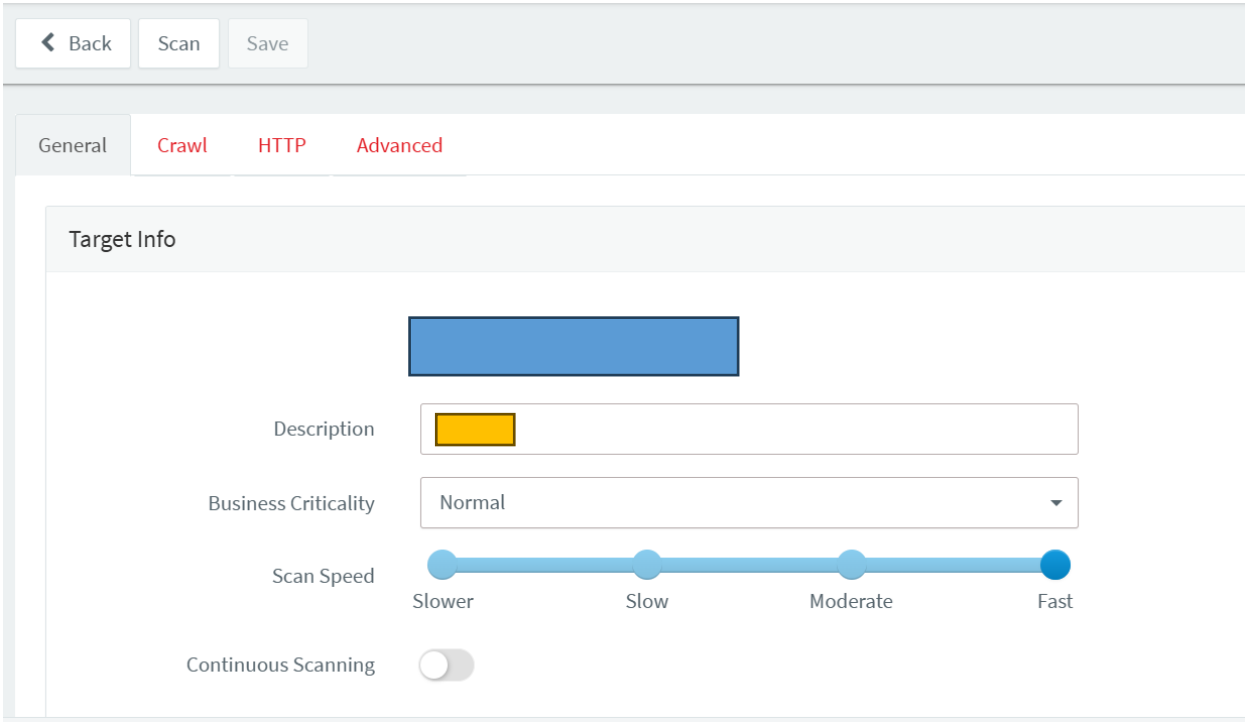

Figure 17. Screen for initiating the vulnerability scanning in Acunetix.

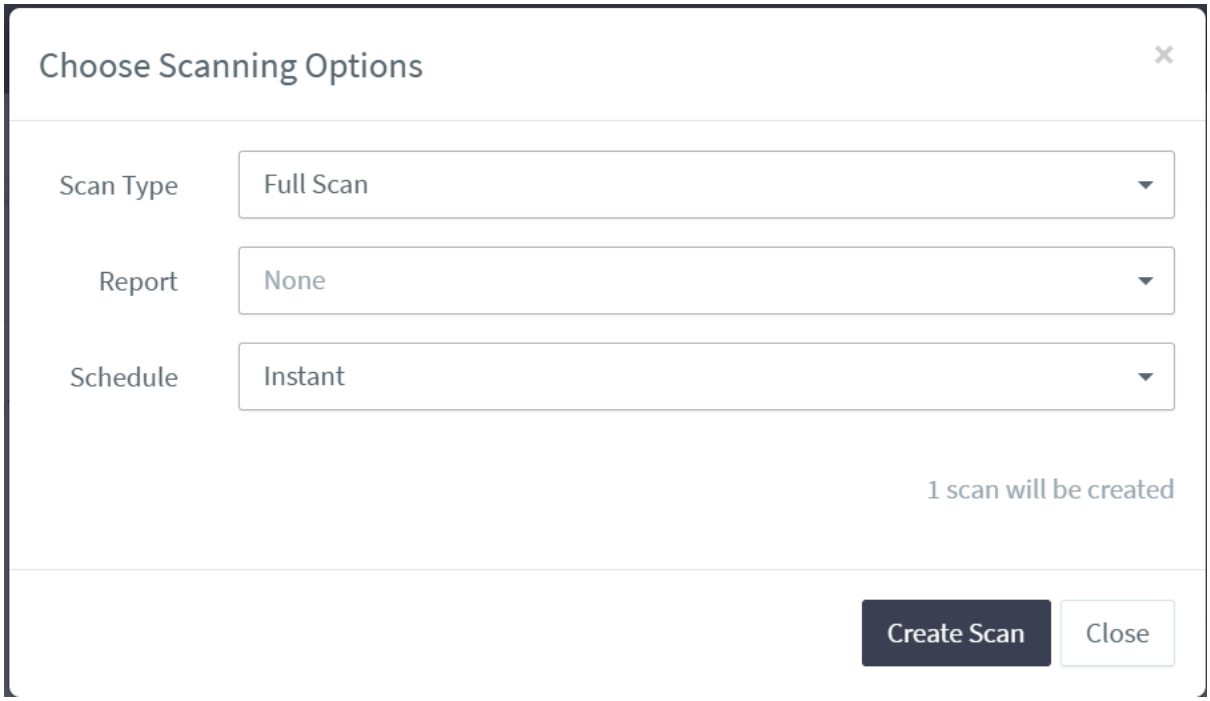

Figure 18. Screen for choosing scanning options in Acunetix.

REFERENCES

[1] M. Mshangi, *To enhance security of information systems in Tanzania: the case of education sector in Tanzania (Doctoral dissertation)*, Open University of Tanzania, 2020.

[2] T. Farah, M. Shojol, M. Hassan, and D. Alam, *Assessment of vulnerabilities of web applications of Bangladesh: A case study of XSS & CSRF*, in 2016 6th International Conference on Digital Information and Communication Technology and Its Applications, DICTAP 2016, 2016, no. November 2017, pp. 74–78.

[3] S. Coughlan, S*tudents blamed for university and college cyber-attacks*, BBC, 2018. https://www.bbc.com/news/education-45496714

[4] Y. S. Jang and J. Y. Choi, *Detecting SQL injection attacks using query result size*, Comput. Secur. Sci., vol. 44, pp. 1–15, 2014, doi: 10.1016/j.cose.2014.04.007.

[5] I. G. N. Mantra, M. S. Hartawan, H. Saragih, and A. A. Rahman, *Web vulnerability assessment and maturity model analysis on Indonesia higher education*, in Procedia Computer Science, 2019, vol. 161, pp. 1165–1172. doi: 10.1016/j.procs.2019.11.229.

[6] Z. Xu, Q. Hu, and C. Zhang, *Why computer talents become computer hackers: Start with talent and skills driven by curiosity and hormones, constrained only by moral values and judgment*, Commun. ACM, vol. 56, no. 4, pp. 64–74, 2013.

[7] B. Chen, P. Zavarsky, R. Ruhl, and D. Lindskog, *A study of the effectiveness of CSRF guard*, in Proceedings - 2011 IEEE International Conference on Privacy, Security, Risk and Trust and IEEE International Conference on Social Computing, PASSAT, 2011, pp. 1269–1272.

[8] M. Bhatia and D. J. K. Maitra, *E-learning Platforms Security Issues and Vulnerability Analysis*, in 2018 International Conference on Computational and Characterization Techniques in Engineering & Sciences (CCTES), 2018, pp. 276–285.

[9] R. Adebiaye, *Mitigating Vulnerability Risks in Cybersecurity Using Predictive Measures*, Int. J. Adv. Sci. Res. Dev., vol. 4, no. 10, p. 12, 2017.

[10] B. Rexha, A. Halili, K. Rrmoku, and D. Imeraj, *Impact of secure programming on web application vulnerabilities*, in 2015 IEEE International Conference on Computer Graphics, Vision and Information Security, CGVIS 2015, 2016, pp. 61–66.

[11] R. Jablon, *University of California victim of nationwide hack attack*, abcnews, 2021. https://abcnews.go.com/Technology/wireStory/university-california-victim-nationwide-hack-attack-76847800 (accessed Mar. 10, 2022).

[12] N. Elisa, *Usability, Accessibility and Web Security Assessment of E-government Websites in Tanzania*, Int. J. Comput. Appl., vol. 164, no. 5, pp. 42–48, 2017.

[13] B. Mtakati and F. Sengati, *Cybersecurity posture of higher learning institutions in Tanzania*, no. May, 2021.

[14] E. D. Kundy, *Cyber Security Threats in Higher Learning Institutions in Tanzania, a Case of University of Arusha and Tumaini University Makumira*, Olva Acad. Res., vol. 2, no. 3, pp. 1–38, 2019.

[15] A. Kondoro and J. Mtebe, *Investigating Secure Implementation of Government Web based Systems in Tanzania*, Int. J. Comput. Appl., vol. 182, no. 10, pp. 6–14, 2018.

[16] S. Ally, *Secure Software Deployment : Investigating the Security Vulnerabilities of MOODLE LMS in Public Higher Learning Institutions in Tanzania*, vol. 5, no. 3, pp. 53–61, 2016.

[17] L. Muniandy, B. Muniandy, and Z. Samsudin, *Cyber Security Behaviour among Higher Education Students in Malaysia*, J. Inf. Assur. Cybersecurity, vol. 2017, pp. 1–13, 2017, doi: 10.5171/2017.800299.

[18] C. D. Marcum, G. E. Higgins, M. L. Ricketts, and S. E. Wolfe, *Hacking in High School: Cybercrime*

Perpetration by Juveniles, Deviant Behav., vol. 35, no. 7, pp. 581–591, 2014, doi: 10.1080/01639625.2013.867721.

[19] C. A. Horne, A. Ahmad, and S. B. Maynard, *A theory on information security*, Proc. 27th Australas. Conf. Inf. Syst. ACIS 2016, pp. 1–12, 2016.

[20] S. Zeadally, *Harnessing Artificial Intelligence Capabilities to Improve Cybersecurity*, IEEE Access 2020.

[21] W. Kikude, *Network vulnerabilities analysis : a case study of the college of informatics and virtual education (CIVE) (Master's dissertation)*., The University of Dodoma, 2019.

[22] A. Shrivastava, S. Choudhary, and A. Kumar, *XSS vulnerability assessment and prevention in web application*, in Proceedings on 2016 2nd International Conference on Next Generation Computing Technologies, NGCT 2016, 2017, no. October, pp. 850–853.

[23] M. Z. Murah and A. A. Ali, *Web assessment of Libyan government e-Government services*, Int. J. Adv. Comput. Sci. Appl., vol. 9, no. 12, pp. 583–590, 2018.

[24] A. Begum, M. M. Hassan, T. Bhuiyan, and M. H. Sharif, *RFI and SQLi based local file inclusion vulnerabilities in web applications of Bangladesh*, Int. Work. Comput. Intell., no. December, pp. 21–25, 2017.

[25] M. Moniruzzaman, F. Chowdhury, and M. S. Ferdous, *Measuring Vulnerabilities of Bangladeshi Websites*, in 2nd International Conference on Electrical, Computer and Communication Engineering, ECCE 2019, 2019, no. February.

[26] Ranjit_Kumar, *Research_Methodology_A_Step-by-Step_G. Chennai*, SAGE Publications Ltd, 2011.

[27] S. Bhalerao and P. Kadam, *Sample size calculation*, Int. J. Ayurveda Res., vol. 1, no. 1, p. 55, 2010.

[28] S. Bairwa, B. Mewara, and J. Gajrani, *Vulnerability Scanners: A Proactive Approach to Assess Web Application Security*, Int. J. Comput. Sci. Appl., vol. 4, no. 1, pp. 113–124, 2014.

[29] N. Khoury, P. Zavarsky, D. Lindskog, and R. Ruhl, *An analysis of black-box web application security scanners against stored SQL injection*, in Proceedings - 2011 IEEE International Conference on Privacy, Security, Risk and Trust and IEEE International Conference on Social Computing, PASSAT/SocialCom 2011, 2011, pp. 1095–1101.

[30] S. El Idrissi, N. Berbiche, F. Guerouate, and M. Sbihi, *Performance evaluation of web application security scanners for prevention and protection against vulnerabilities*, Int. J. Appl. Eng. Res., vol. 12, no. 21, pp. 11068–11076, 2017.

[31] M. Agreindra Helmiawan, E. Firmansyah, I. Fadil, Y. Sofivan, F. Mahardika, and A. Guntara, *Analysis of Web Security Using Open Web Application Security Project 10*, 2020.

[32] J. B. Ulven and G. Wangen, *A systematic review of cybersecurity risks in higher education*, Futur. Internet, vol. 13, no. 2, pp. 1–40, 2021, doi: 10.3390/fi13020039.

[33] B. Zhao et al., *A large-scale empirical analysis of the vulnerabilities introduced by third-party components in IoT firmware*, ISSTA 2022 - Proc. 31st ACM SIGSOFT Int. Symp. Softw. Test. Anal., pp. 442–454, 2022, doi: 10.1145/3533767.3534366.

[34] A. Cobleigh, M. Hell, L. Karlsson, O. Reimer, J. Sönnerup, and D. Wisenhoff, *Identifying, Prioritizing and Evaluating Vulnerabilities in Third Party Code*, Proc. - IEEE Int. Enterp. Distrib. Object Comput. Work. EDOCW, vol. 2018-Octob, pp. 208–211, 2018, doi: 10.1109/EDOCW.2018.00038.

[35] H. M. Hassan, D. M. Reza, and M. A.-A. Farkhad, *An Experimental Study of Influential Elements on Cyberloafing from General Deterrence Theory Perspective Case Study: Tehran Subway Organization*, Int. Bus. Res., vol. 8, no. 3, pp. 91–98, 2015.

[36] M. Bada, B. Von Solms, and I. Agrafiotis, *Reviewing national cybersecurity awareness for users*

and executives in Africa, Int. J. Adv. Secur., vol. 12, no. 1 and 2, pp. 108–118, 2019.
[37] J. Maranga, *Emerging Issues in Cyber Security for Institutions of Higher Education*, Int. J. Comput. Sci. Netw., vol. 8, no. 4, August 2019, 2019.
[38] S. K. Lala, A. Kumar, and T. Subbulakshmi, *Secure web development using OWASP guidelines*, Proc. - 5th Int. Conf. Intell. Comput. Control Syst. ICICCS 2021, no. Iciccs, pp. 323–332, 2021, doi: 10.1109/ICICCS51141.2021.9432179.
[39] OWASP, *OWASP Top 10 Vulnerabilities*, 2021. https://owasp.org/Top10/ (accessed Aug. 24, 2021).
[40] P. D. Pacey and J. H. Purnell, *OWASP_Top_10 vulnerabilities 2017*,OWASP, 2017.
[41] A. K. Priyanka and A. Sqlmap, *Web ApplicationVulnerabilities : Exploitation and Prevention*, IEEE Xplore, 2020.
[42] W. Qianqian and L. Xiangjun, *Research and design on Web application vulnerability scanning service*, in Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS, 2014, pp. 671–674.
[43] H. Shahriar, *Security vulnerabilities and mitigation techniques of web applications*, in SIN 2013 - Proceedings of the 6th International Conference on Security of Information and Networks, 2013, p. 459.
[44] I. Yusof and A. S. K. Pathan, *Mitigating Cross-Site Scripting Attacks with a Content Security Policy*, Computer (Long. Beach. Calif)., vol. 49, no. 3, pp. 56–63, 2016.
[45] I. Medeiros, N. Neves, and M. Correia, *DEKANT: A static analysis tool that learns to detect web application vulnerabilities*, in ISSTA 2016 - Proceedings of the 25th International Symposium on Software Testing and Analysis, 2016, pp. 1–11.
[46] H. Kour, *Tracing out Cross Site Scripting Vulnerabilities in Modern Scripts*, Int. J. Adv. Netw. Appl., vol. 07, no. 05, pp. 2862–2867, 2016.
[47] Y. F. G. M. Elhakeem and B. I. A. Barry, *Developing a security model to protect websites from cross-site scripting attacks using ZEND framework application*, in Proceedings - 2013 International Conference on Computer, Electrical and Electronics Engineering: "Research Makes a Difference", ICCEEE 2013, 2013, pp. 624–629.
[48] E. A. Altulaihan, A. Alismail, and M. Frikha, *A Survey on Web Application Penetration Testing*," Electron., vol. 12, no. 5, 2023, doi: 10.3390/electronics12051229.
[49] A. Masood, "*Cyber security for service oriented architectures in a Web 2.0 world: An overview of SOA vulnerabilities in financial services*, in 2013 IEEE International Conference on Technologies for Homeland Security, HST 2013, 2013, pp. 1–6.
[50] C. R. Harrell, M. Patton, H. Chen, and S. Samtani, *Vulnerability assessment, remediation, and automated reporting: Case studies of higher education institutions*, 2018 IEEE Int. Conf. Intell. Secur. Informatics, ISI 2018, pp. 148–153, 2018, doi: 10.1109/ISI.2018.8587380.
[51] S. Kausar, X. Huahu, A. Ullah, Z. Wenhao, and M. Y. Shabir, *Fog-Assisted Secure Data Exchange for Examination and Testing in E-learning System*, Mob. Networks Appl., vol. 28, no. 2, pp. 673–689, 2023, doi: 10.1007/s11036-019-01429-x.
[52] C. Joshi and U. K. Singh, *Information security risks management framework – A step towards mitigating security risks in university network*, J. Inf. Secur. Appl., vol. 35, pp. 128–137, 2017.
[53] S. Kumar, R. Mahajan, N. Kumar, and S. K. Khatri, *A study on web application security and detecting security vulnerabilities*, in 6th International Conference on Reliability, Infocom Technologies and Optimization: Trends and Future Directions, ICRITO, 2018, vol. 2018-Janua, pp. 451–455.
[54] A. C. Perera, K. Kesavan, S. V. Bannakkotuwa, C. Liyanapathirana, and L. Rupasinghe, *E-*

commerce (WEB) application security: Defense against reconnaissance, in Proceedings - 2016 16th IEEE International Conference on Computer and Information Technology, CIT 2016, 2016 6th International Symposium on Cloud and Service Computing, IEEE SC2 2016 and 2016 International Symposium on Security and Privacy in Social Netwo, 2017, pp. 732–742. doi: 105.

[55] I. Alshourbaji et al., *An Approach To Weigh Cybersecurity Awareness Questions In Academic Institutions Based On Principle Component Analysis: A Case Study Of Saudi Arabia*, Int. J. Sci. Technol. Res, vol. 10, no. 04, pp. 319–326, 2021.

[56] N. Ahmed, M. R. Islam, U. Kulsum, M. R. Islam, M. E. Haque, and M. S. Rahman, *Demographic factors of cybersecurity awareness in Bangladesh*, in 2019 5th International Conference on Advances in Electrical Engineering, ICAEE 2019, 2019, no. June 2018, pp. 685–690.

[57] G. Deepa, P. S. Thilagam, F. A. Khan, A. Praseed, A. R. Pais, and N. Palsetia, *Black-box detection of XQuery injection and parameter tampering vulnerabilities in web applications*, Int. J. Inf. Secur., vol. 17, no. 1, pp. 105–120, 2018.

[58] P. Xiong and L. Peyton, *A model-driven penetration test framework for web applications*, in PST 2010: 2010 8th International Conference on Privacy, Security and Trust, 2010, pp. 173–180.