


Vulnerability Assessment of Angolan University Web Applications

Emanuel Mateus and Carlos Serrão ^a

Instituto Universitário de Lisboa (ISCTE-IUL), ISTAR_Iscte, Lisboa, Portugal

Keywords: Web Applications, Security, Vulnerability Assessment, Angolan Universities, OWASP, CWE.


Abstract: Vulnerability assessment is one of the technical procedures that can help prevent serious security breaches, which, when exploited, can undermine brand credibility and or the continuity of a business. Universities hold and process important relevant and sensitive student and staff information appealing to attackers and might affect the organisations' credibility if such information is disclosed. This work presents a study conducted to assess the security status of the Angolan universities' web applications, identifying the most frequent security vulnerabilities and their criticality, based on OWASP Top 10 and CWE Top 25 references to identify and validate the findings discovered during the automatic vulnerability assessment process.

1 INTRODUCTION

In the current context, the Internet is no longer a "simple" worldwide network of computers. Instead, it has become an actual content and business platform where several companies develop their business engagement activities and gain advantages over their competitors. Web applications on the World Wide Web (WWW) have become the front-facing image of organisations in cyberspace, collecting and processing data from customers, users, staff, business partners and others. Web applications can be defined as computer programs that run inside a web browser, where the code to be executed is fetched from a remote location (Salini & Kanmani, 2012). Throughout this evolution of the Internet and WWW, organisations web sites endured several transformations, moving from simple sets of static HTML pages to complex full-featured applications to implement and take essential business functions.

In Angola, similarly to what happens in other parts of the world, university web applications have transformed in the past years. These web applications are currently based on open-source content management systems (CMS) solutions such as Joomla, WordPress, and Umbraco. In many cases, these CMS also serve as a web portal that provides access to the university Enterprise Resource Planning (ERP) systems accessible from anywhere in the world, using a web browser. Therefore, these

applications are critical and strategic for the management of universities. However, given their web nature, they are exposed to the risks that the Internet presents. Therefore, ensuring their security and trust are some of the most critical success factors of these applications because they are open doors for threat actors (Huang et al., 2017). The complexity, modularity and inclusion of third-party libraries in the development and distribution of web applications make it difficult to ensure that they are entirely vulnerability-free: software bugs that might lead to security vulnerabilities, improper systems configuration or any other vulnerabilities that, when adequately exploited by an attacker might compromise the application and data security. According to a recent study conducted by Acunetix, 63% of all analysed web applications have medium severity vulnerabilities, while 26% contain high severity vulnerabilities (Acunetix, 2020). Therefore, organisations need to look with rigour to the security of these web applications to prevent the loss of sensitive data and avoid business reputation and drop in revenue. Web application security is the process of protecting websites and online services against different security threats that exploit vulnerabilities in an application's code (M. B. Shuaibu & Ibrahim, 2017). Considering this, we have conducted a study to understand how vulnerable their web applications are and the most common vulnerabilities found in the context of the significant Angolan universities. This

^a <https://orcid.org/0000-0002-4847-2432>

study's objective was to create the necessary security awareness on organisations about the risks they expose their organisations' information to and alert them to the essential mitigations that need to be accomplished to mitigate such risks. The present work identified the most frequent security vulnerabilities in Angolan universities' web applications and found that many of these flaws could be eliminated if the evaluated institutions introduced the regular practice of vulnerability and intrusion tests in their technical and administrative procedures. The study is also innovative, because to the best of our knowledge, this is the first time this type of security study is conducted on Angolan universities.

This work is organised into five different sections. The first section of this paper introduces the topic and builds the context for the problem that the authors address. The following section presents relevant related information, specifically on vulnerability assessment and frameworks and applicability to web application security. The third section of this paper introduces and describes the used methodology applied to conduct this work. The fourth section presents the most relevant results obtained after following the methodology selected. Finally, the last section presents the major conclusions of this work.

2 VULNERABILITY ASSESSMENT AND FRAMEWORKS

This part of the article presents an overview of related works conducted in this field, on what refers to the vulnerability assessment activities and organisations that are international benchmarks, and a set of frameworks that can be used to conduct security assessments.

2.1 Vulnerability Assessment

Vulnerability is defined as any weakness that an attacker can exploit to make an information asset susceptible to damage. A web application vulnerability assessment is an in-depth analysis of the building functions, systems, and site characteristics to identify the weaknesses of the system, the sufficiency of existing security measures (if any), the lack of redundancy and the duration of recovery of the operation of an attack (Liu et al., 2012)

Vulnerability analysis involves discovering a subset of data input failures with which a malicious user can exploit logic errors in an application and

make it unsafe (Sparks et al., 2007). This process can be done independently or integrated as one of the steps of more extensive intrusion testing.

Multiple organisations and works exist that deal with this topic of web application vulnerability and security scanning. One of the most relevant organisations that deal with web application security is OWASP (Open Web Application Security Project). This non-profit organisation is dedicated to developing tools that enable application security development practices to increase the overall application security and creating and maintaining documentation to create developers' application security awareness. One of the most relevant OWASP documents is the OWASP Top Ten that contains a consensual and curated list where the ten (10) most critical and frequent web application security risks are identified (OWASP Foundation, 2015). The most up to date risk list include the following: A1: Injection; A2: Broken Authentication; A3: Sensitive Data Exposure; A4: XML External Entities; A5: Broken Access Control; A6: Security Misconfiguration; A7: Cross-Site Scripting (XSS); A8: Insecure Deserialization; A9: Using Components with Known Vulnerabilities; and A10: Insufficient Logging & Monitoring. Another relevant organisation is the MITRE Corporation, a US government-funded non-profit institution whose primary purpose is computer security. MITRE has created and keeps a list of the most frequent software security vulnerabilities (MITRE, 2020b). In addition, MITRE implements the Common Weakness Enumeration (CWE), a helpful tool that should be considered when assessing an application's vulnerabilities. It uses a common language to identify and describe vulnerabilities and present ways to mitigate and prevent their occurrence (MITRE, 2020c). Yearly, MITRE also produces a Top 25 list of the most common identified weaknesses (Howard, 2009; MITRE, 2020a).

In the scientific literature, it is also possible to find multiple works where authors present and discuss web application security approaches and some global studies conducted to access this security. Some of these studies use a more global approach, for example, considering the generic web applications security analysis for the whole country. At the same time, some other works adopt a more sectorial approach. For instance, in (Teodoro & Serrão, 2011) and (Teodoro & Serrão, 2010) the authors propose a process for improving the web applications quality through security assessments while applying the same methodology to assess the security of the most relevant web applications the country. Furthermore, other authors propose a similar approach to evaluate

the security of e-government sites (Zhao & Zhao, 2010). Finally, some other authors provide a comprehensive comparison between different vulnerability assessment methods and penetration testing (Doshi & Trivedi, 2015; B. M. Shuaibu et al., 2015; ur Rehman et al., 2017).

2.2 Web Applications Security Assessment Frameworks

In the current context, several methodologies and standards related to intrusion testing and vulnerability assessment can be used to assess the security of web applications. These methodologies include, for instance, the NIST SP 800-115 Technical Guide to Information Security Testing and Assessment (Scarfone et al., 2008), Information Systems Security Assessment Framework (ISAAF) (Rathore et al., 2006), The Open Information Security Services Group (OISSG) (Shanley & Johnstone, 2015), the Open-Source Security Testing Methodology Manual (OSSTMM) (Prandini & Ramilli, 2010) and the Penetration Testing Execution Standard (PTES) (Dinis & Serrao, 2014; Haubris & Pauli, 2013).

These organisations and authors clarify that the purpose of guidelines for intrusion testing is to provide a basic outline of what a penetration test is and outline the steps that should be followed during its execution. However, (Haubris & Pauli, 2013) clarify that the different existing guidelines have several advantages and disadvantages and that the choice should be based on the purpose of the test to be performed.

In work described in this paper, the authors selected the usage of the methodology proposed by NIST SP 800-115 (Scarfone et al., 2008). This selection justifies the methodology simplicity, objectiveness, flexibility, recursive activities, and above all, because it is a guide with widely accepted recommendations.

This methodology provides practical recommendations and technical and administrative procedures for conducting vulnerability assessment tests on computer applications and systems. In practice, it identifies four steps that need to be undertaken: Planning, Discovery, Attack and Report. These four phases cover the entire security assessment lifecycle, starting with planning the assessment activities until presenting the results to the client.

3 WORK METHODOLOGY

As referred, this work adopted the NIST SP 800-115, which is composed of four different stages. The first three stages are described in this section: planning, discovery and attack.

3.1 Planning

In the planning phase, the rules are identified, the approval of the audit by the entities is finalised and documented, and the targets of the tests are set. No actual testing occurs at this stage. This work was carried out for 24 months, only during the weekends, not compromising or overloading the regular application's operations. Four (4) different web application scanners were used to conduct this analysis. These web application scanners are a set of automated scripts that crawl through a web application while looking for security problems. For this work, one commercial web application scanner was selected (Pentest-Tools.com¹), while the other three (3) had an open-source licensing (OWASP ZAP², Skipfish³ and OpenVAS⁴). To standardise the web applications vulnerability assessment findings, all results were grouped according to the risk classification and identification proposed by the OWASP Top Ten. This approach allows the detection and evaluation of the most frequent security vulnerabilities of web applications, to analyse and quantify the degree of criticality of the vulnerabilities of the web applications used by Angola universities and, in the end, classify the degree of the security risk of web applications based on the findings.

For the present study, ten (10) Angola universities were selected that had a web application in operation at the time of the study, excluding other institutions that had intermittent web applications or had only presence on social networks and without any official website. For privacy's sake, it is not possible to disclose the name of the universities that were targeted by our testing processes. However, the primary data sources were obtained following the intrusion tests of higher education institutions in Angola, recognised by the guardianship institution, the Ministry of Higher Education and Science and Technology of Angola, published in 2020 (MINISTÉRIO DA CIÊNCIA E TECNOLOGIA, 2020).

¹ <https://pentest-tools.com/>

² <https://www.zaproxy.org>

³ <https://github.com/spinkham/skipfish>

⁴ <https://www.openvas.org>

3.2 Data Collection and Analysis

This section corresponds to the discovery phase of the methodology. The discovery phase consists of two different parts. The first part is the start of actual testing and covers the collection and verification of information. The identification of services and network ports is carried out to identify potential targets. In addition to identifying the port and service, other techniques collect information on the target network. The second part of the discovery phase is vulnerability analysis, which compares services, applications, and operating systems evaluated using a vulnerabilities knowledgebase. This procedure begins with validating the URLs of the university web application, and from this, it lists all existing links in the domain to perform its evaluation. During the evaluation period, for each link found, the existence of forms was automatically verified, which in turn were subjected to different types of tests to validate the degree of vulnerabilities they presented.

Secondary data sources were obtained using WhatWeb⁵, an open-source application distributed as a package of the Debian-derived Linux distribution Kali Linux⁶. The data was obtained using four tools previously identified. To minimise the false positives, all the automated tools reported findings were manually confirmed through manual inspection. The tests were properly authorised by the involved institutions and followed an ethical approach. A black box security assessment was followed, and all the tests were exclusively carried through the Internet, so some types of data that could be obtained by carrying out more exhaustive and intrusive tests that could cause partial breakdowns or stopping of services or computer systems were ignored to safeguard the business continuity of the involved institutions. WhatWeb was the first tool to be used in the tests. WhatWeb is an open-source tool and has over 1700 plug-ins, giving it an excellent capacity for different web technologies. This tool can be used to recognise or collect data such as IP address, geographic location of servers, and information about content management systems (CMS), blog platforms, identification of Javascript or CSS libraries versions, description of programming languages used in the development of specific applications, types of web servers and their versions.

The OWASP ZAP, Skipfish, OpenVAS and Pentest-Tools.com (PT) performed automated web applications security scanning. These tools were executed from the outside of the institution to be

audited. They were scanning for security vulnerabilities, which can compromise the normal functioning of applications in the short, medium, or long run. These applications also can detect the incorrect configuration of different applications, network assets, and end-of-life applications. Also, some of these tools have features that allow the

Table 1: List of vulnerabilities/features considered in the analysis.

ID	Vulnerability/Feature
CWE-79	Improper Neutralisation of Input During Web Page Generation ("Cross-site Scripting")
CWE-787	Out-of-bounds Write
CWE-20	Improper Input Validation
CWE-125	Out-of-bounds Read
CWE-119	Improper Restriction of Operations within the Bounds of a Memory Buffer
CWE-89	Improper Neutralisation of Special Elements used in an SQL Command ("SQL Injection")
CWE-200	Exposure of Sensitive Information to an Unauthorised Actor
CWE-416	Use After Free
CWE-352	Cross-Site Request Forgery (CSRF)
CWE-78	Improper Neutralisation of Special Elements used in an OS Command ("OS Command Injection")
CWE-190	Integer Overflow or Wraparound
CWE-22	Improper Limitation of a Pathname to a Restricted Directory ("Path Traversal")
CWE-476	NULL Pointer Dereference
CWE-287	Improper Authentication
CWE-434	Unrestricted Upload of File with Dangerous Type
CWE-732	Incorrect Permission Assignment for Critical Resource
CWE-94	Improper Control of Generation of Code ("Code Injection")
CWE-522	Insufficiently Protected Credentials
CWE-611	Improper Restriction of XML External Entity Reference
CWE-798	Use of Hard-coded Credentials
CWE-502	Deserialization of Untrusted Data
CWE-269	Improper Privilege Management
CWE-400	Uncontrolled Resource Consumption
CWE-306	Missing Authentication for Critical Function
CWE-862	Missing Authorisation

⁵ <https://morningstarsecurity.com/research/whatweb>

⁶ <https://www.kali.org>

management of the vulnerabilities encountered over some time and have a security flaw verification system that will enable them to make recommendations on the technical procedures that must be performed to mitigate or eliminate the detected vulnerabilities. Data collection was the main objective of validating the existence of web application security flaws that could compromise organisations' security. As such, ten of the twenty-five (10/25) security vulnerability features chosen are part of a broad consensus on the most critical security risks for web applications as identified in the OWASP Top 10. The remaining 15 (fifteen) features are part of the CWE/SANS Top 25, which lists the 25 (twenty-five) elements that are classified as being the most dangerous software failures (MITRE, 2020a). It is important to refer that the twenty-five (25) characteristics listed below have several sub-characteristics that were not listed here for the sake of space (Table 1).

The vulnerability assessment process was performed based on passive and non-intrusive tests that allowed the identification and evaluation of the different types of vulnerabilities found in web applications. The OWASP ZAP tool was used as the main instrument to validate the mandatory features listed. This tool was the only one of the three used to perform the tests proposed on all websites successfully.

4 RESULTS AND DISCUSSION

This section of this paper describes both the attack and reporting stages of the methodology. Performing an attack is at the heart of any penetration test. At this stage, scanning vulnerabilities identified in the previous phase, trying to exploit them. If an attack is successful, the vulnerability will be confirmed, and preventive measures will be taken to mitigate the security breach. The reporting phase co-occurs with the other three steps of the intrusion test. In the planning phase, technical and administrative procedures are documented in the report, while in the stages of discovery and attack, the findings are recorded and maintained in preliminary reports. After the test, a report should be prepared to describe the results, presenting a risk classification and providing guidance on mitigating the weaknesses discovered.

4.1 Classification

There are multiple approaches to risk analysis. Some risk computing approaches involve considering the

threats, vulnerabilities and impact (Liu et al., 2012; Lubis & Tarigan, 2017): Risk = Threat * Vulnerability * Impact. OWASP defines an adaptable model, simplified and applicable to web applications, where the highlight goes to probability and impact factors: Risk = Probability * Impact. In this work, we have selected the second method, which estimates the probability of occurrence of vulnerability exploitation and estimates the impact caused. Both elements are used to calculate the degree of general severity risk that can qualitatively be evaluated as low, medium or high, and quantitatively the values range from 0 to 9: Low risk (0 - 3), medium risk (3 - 6) and high risk (6 - 9).

4.2 Results

The sequential use of the four different web application security scanners allowed the identification of more than one thousand (1310) web application security flaws in Angolan universities. All the various scanning tools were able to identify at least one security vulnerability. The Skipfish scanner identified the most significant number of security issues (1016), while the PT scanner only identified almost eighty (76) security issues (Table 2).

It is also possible to check the qualitative risk level according to the university's web application testing and the tool used to conduct the tests (Figure 1). For example, in Table 3, it is possible to identify that multiple different web application tools identify that five universities web applications have a high-security risk level.

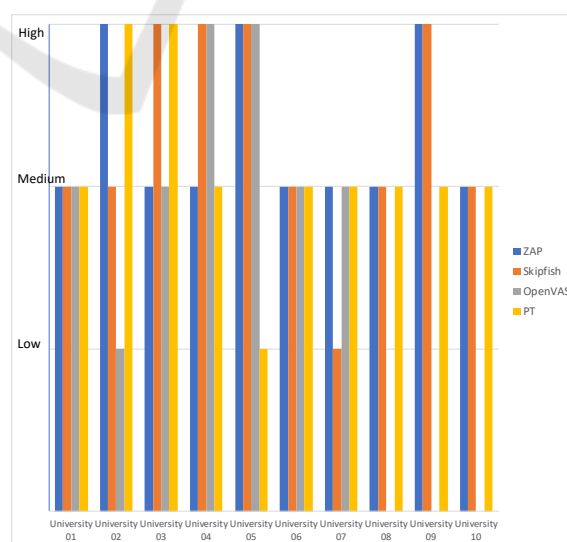


Figure 1: OWASP security risk level per university and per tool.

Table 2: List of vulnerabilities identified by tool.

Tool	Vulnerabilities			Total
	Low	Medium	High	
ZAP	71	19	5	95
Skipfish	143	810	63	1016
OpenVAS	9	112	2	123
PT	40	33	3	76
				1310

While looking at the detail of the identified vulnerabilities (Figure 2), it is possible to conclude that only one of the categories of OWASP Top Ten risks was not detected in any of the analysed sites by the selected tools - the use of XML with external entities (XXE). However, sensitive data exposure, cross-site scripting, and inadequate security configuration were the categories of vulnerabilities that most contributed to the increase in the number of security problems found by presenting 704 and 22 (742), three hundred and ninety-six (396) and one hundred and forty-one (141) issues respectively.

Table 3: OWASP qualitative security risk level according to the university tested and tool used.

Universities	Qualitative Evaluation			
	ZAP	Skipfish	OpenVAS	PT
University 01	Medium	Medium	Medium	Medium
University 02	High	Medium	Low	High
University 03	Medium	High	Medium	High
University 04	Medium	High	High	Medium
University 05	High	High	High	Low
University 06	Medium	Medium	Medium	Medium
University 07	Medium	Low	Medium	Medium
University 08	Medium	Medium	N/A	Medium
University 09	High	High	N/A	Medium
University 10	Medium	Medium	N/A	Medium

	A1:2017-Injection	A2:2017-Broken Authentication	A3:2017-Sensitive Data Exposure	A4:2017-XML External Entities (XXE)	A5:2017-Broken Access Control	A6:2017-Security Misconfiguration	A7:2017-Cross-Site Scripting (XSS)	A8:2017-Insecure Deserialisation	A9:2017-Using Components with Known Vulnerabilities	A10:2017-Insufficient Logging & Monitoring
ZAP	5	0	56	0	0	12	39	0	2	1
Skipfish	53	0	663	0	0	12	306	0	0	0
OpenVAS	2	0	0	0	0	92	29	2	0	0
PT	3	1	23	0	6	25	22	0	0	0
Total	63	1	742	0	6	141	396	2	2	1

Figure 2: Number of vulnerabilities detected by tool according to OWASP Top Ten.

If, on the one hand, sensitive data exposure can be considered to be low-severity security flaws, failures

related to the Injection category are considered to be high risk; this means that the sixty-three (63) high-risk security flaws that were found in this study is explored, a malicious entity can immediately compromise the system or web application. On the other hand, in the opposite direction, there are low-risk failures where the category of vulnerabilities related to sensitive data exposure falls.

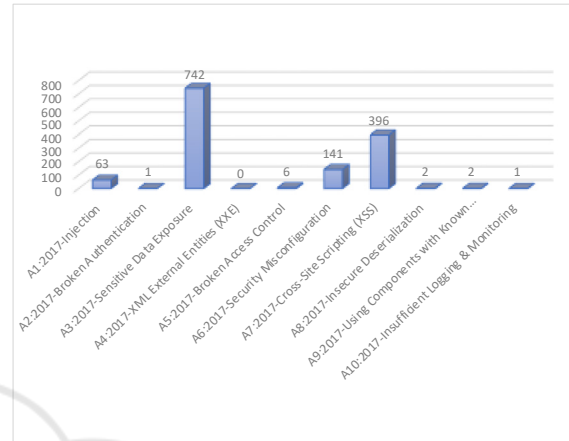


Figure 3: Number of vulnerabilities detected according to the OWASP Top Ten.

Figure 4 illustrates the vulnerabilities discovered according to the web scanners used where it is possible to verify that the use of multiple web scanners, which have different levels of parameterisation, sensitivity and vulnerability arrest algorithms, allowed on the one hand to identify common security flaws that were reported by all scanners and on the other hand some security flaws were only discovered by two or even a single security scanner.

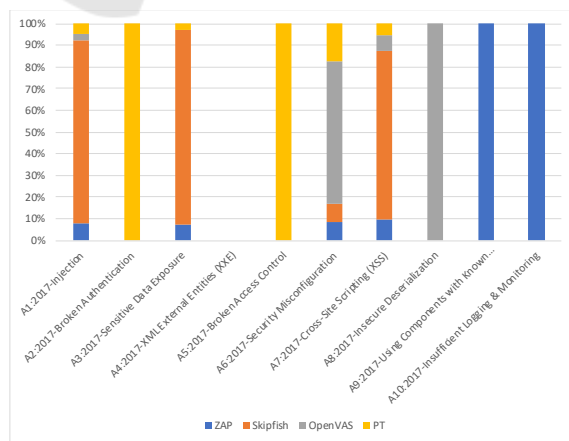


Figure 4: Percentage of vulnerabilities detected according to the tool that was used.

From Figure 4, it is also possible to observe that the ZAP scanner was the only one that detected security flaws related to the use of components with known vulnerabilities and logging and insufficient monitoring. Moreover, this was the only scanner with built-in capabilities to detect such security problems on web applications. Similarly, the PT scanner was the only one that detected security flaws directly linked to the failed access control authentication (Broken Access Control). All web scanners converge on identifying security flaws related to Injection, Poor Security Configuration, and Cross-Site Scripting. During the study, it was also possible to find that only two (2) of the ten (10) universities maintain their degree of security risk from when the study was conducted while the other universities see their security risk level degrading.

5 CONCLUSIONS

The number of security flaws identified in this study and the degree of vulnerability of web applications ranging from medium to high risk demonstrates that the web applications of Angolan universities, in general, are not secure. It was also possible to conclude that 70% of applications communicate through unsafe channels due to the absence of security certificates (and therefore without SSL/TLS) and use JavaScript libraries with known vulnerabilities, thus being exposed to cross-site scripting attacks. The present study also found that all web applications provide too much information about their ecosystems (operating systems, server versions and applications), which can enhance the increase in the degree of vulnerabilities if the systems are not up-to-date and bug-free. Moreover, this information leakage can provide more clues for attackers to compromise those applications and the information they hold. Another relevant aspect of this study is discovering the relationship between the security flaws found with the various content management systems (CMS) that universities use. That is, the lack of updating of these applications, their components and JavaScript libraries increase the risk of security breaches and makes all dependent applications vulnerable since this type of applications are interconnected with the others. In some cases, they serve as a web portal to access other resources or applications. The usage of different types of scanners, with varying settings of parameterisation in terms of sensitivity, depth and aggressiveness, allowed to identify of security flaws that another web scanner could have ignored and consequently to obtain a false

sense of security due to the absence of certain types of vulnerabilities that exist in the application but that the scanner used was not able to identify. For this reason, it is recommended to use several web scanners simultaneously when assessing the vulnerabilities of web applications. It is also possible to conclude that the security flaws discovered in this study could be avoided if the targeted educational institutions adopted penetration testing in the development, installation, configuration, and maintenance of network and/or application assets. This study listed open-source tools (framework guides) that allow security vulnerabilities and intrusion testing to be carried out. These tools are pretty effective, as it can be said that if Angolan universities regularly perform penetration tests in their applications and or ecosystems, the security flaws identified here would be discovered and eliminated. Consequently, the web applications would be safer. To the best of our knowledge, this was the first time that a security evaluation of the web applications of the Angola universities took place, and the conclusions reached depict a dark scenario in terms of web application security. Therefore, there are essential steps that need to take place at the Angola universities to include good application security practices in the implementation, configuration and deployment of these university web applications that need to consider the appropriate security risks and adopt measures to mitigate them. The work described in this paper should be cyclically repeated over time to measure the security maturity of such web applications, allowing for the establishment of proper secure by default environments.

REFERENCES

- Acunetix. (2020). *Web Application Vulnerability Report 2020*. <https://www.acunetix.com/white-papers/acunetix-web-application-vulnerability-report-2020/>
- Dinis, B., & Serrao, C. (2014). External footprinting security assessments: Combining the PTES framework with open-source tools to conduct external footprinting security assessments. *International Conference on Information Society (i-Society 2014)*, 313–318. <https://doi.org/10.1109/i-Society.2014.7009066>
- Doshi, J., & Trivedi, B. (2015). Comparison of vulnerability assessment and penetration testing. *Int. J. Appl. Inf. Syst.*, 8(6), 51–54.
- Haubris, K. P., & Pauli, J. J. (2013). Improving the efficiency and effectiveness of penetration test automation. *2013 10th International Conference on Information Technology: New Generations*, 387–391.

- Howard, M. (2009). Improving Software Security by Eliminating the CWE Top 25 Vulnerabilities. *IEEE Security Privacy*, 7(3), 68–71. <https://doi.org/10.1109/MSP.2009.69>
- Huang, H.-C., Zhang, Z.-K., Cheng, H.-W., & Shieh, S. W. (2017). Web application security: threats, countermeasures, and pitfalls. *Computer*, 50(6), 81–85.
- Liu, C., Tan, C.-K., Fang, Y.-S., & Lok, T.-S. (2012). The security risk assessment methodology. *Procedia Engineering*, 43, 600–609.
- Lubis, A., & Tarigan, A. (2017). Security Assessment of Web Application Through Penetration System Techniques. *Jend. Gatot Subroto Km*, 4(100), 296–303.
- MINISTÉRIO DA CIÊNCIA E TECNOLOGIA. (2020). *LISTA DAS INSTITUIÇÕES DE INVESTIGAÇÃO CIENTÍFICA E DESENVOLVIMENTO TECNOLÓGICO*. http://ciencia.ao/redes/rede_instituicoes.pdf
- MITRE. (2020a). *MITRE - 2021 CWE Top 25 Most Dangerous Software Weaknesses*.
- MITRE. (2020b). *MITRE - Common Vulnerabilities Enumeration*.
- MITRE. (2020c). *MITRE - Common Weakness Enumeration*.
- OWASP Foundation. (2015). Category:OWASP Top Ten Project - OWASP. *OWASP™ Foundation*.
- Prandini, M., & Ramilli, M. (2010). Towards a practical and effective security testing methodology. *The IEEE Symposium on Computers and Communications*, 320–325.
- Rathore, B., Brunner, M., Dilaj, M., Herrera, O., Brunati, P., Subramaniam, R., Raman, S., & Chavan, U. (2006). Information systems security assessment framework (issaf). *Draft 0.2 B, 1*, 2006.
- Salini, P., & Kanmani, S. (2012). Security requirements engineering process for web applications. *Procedia Engineering*, 38, 2799–2807.
- Scarfone, K. A., Souppaya, M. P., Cody, A., & Orebaugh, A. D. (2008). *Sp 800-115. technical guide to information security testing and assessment*. National Institute of Standards & Technology.
- Shanley, A., & Johnstone, M. N. (2015). *Selection of penetration testing methodologies: A comparison and evaluation*.
- Shuaibu, B. M., Norwawi, N. M., Selamat, M. H., & Al-Alwani, A. (2015). Systematic review of web application security development model. *Artificial Intelligence Review*, 43(2), 259–276.
- Shuaibu, M. B., & Ibrahim, R. A. (2017). Web application development model with security concern in the entire lifecycle. *2017 4th IEEE International Conference on Engineering Technologies and Applied Sciences (ICETAS)*, 1–6.
- Sparks, S., Embleton, S., Cunningham, R., & Zou, C. (2007). Automated vulnerability analysis: Leveraging control flow for evolutionary input crafting. *Twenty-Third Annual Computer Security Applications Conference (ACSAC 2007)*, 477–486.
- Teodoro, N., & Serrão, C. (2010). Web applications security assessment in the Portuguese world wide web panorama. In *Communications in Computer and Information Science: Vol. 72 CCIS*. https://doi.org/10.1007/978-3-642-16120-9_17
- Teodoro, N., & Serrão, C. (2011). Web application security: Improving critical web-based applications quality through in-depth security analysis. *International Conference on Information Society, i-Society 2011*.
- ur Rehman, H., Nazir, M., & Mustafa, K. (2017). Security of web application: state of the art. *International Conference on Information, Communication and Computing Technology*, 168–180.
- Zhao, J. J., & Zhao, S. Y. (2010). Opportunities and threats: A security assessment of state e-government websites. *Government Information Quarterly*, 27(1), 49–56.