A Fuzzy Classifier-Based Penetration Testing for Web Applications

J. K. Alhassan[1(✉)], Sanjay Misra[2], A. Umar[1], Rytis Maskeliūnas[3], Robertas Damaševičius[3], and Adewole Adewumi[2]

[1] Federal University of Technology, Minna, Nigeria
jkalhassan@futminna.edu.ng,
umarabdulkadir@hotmail.com
[2] Covenant University, Ota, Nigeria
{sanjay.misra,
adewole.adewumi}@covenantuniversity.edu.ng
[3] Kaunas University of Technology, Kaunas, Lithuania
{rytis.maskeliunas,robertas.damasevicius}@ktu.lt

Abstract. The biggest challenge of Web application is the inestimable losses arising from security flaws. Two approaches were advanced by a number of scholars to provide security to Web space. One of such approach is vulnerability assessment, which is a conscious effort to isolate, identify and recognize potentials vulnerabilities exploited by attackers. The second being the estimation and determination of level of risks/threats posed to Web applications by vulnerabilities obvious to the developer (or tester); this is generally referred to as penetration testing. Recently, there is Vulnerability Assessment and Penetration Testing (VAPT) that combined these two schemes to improve safety and effectively combat the menace of attackers on Web applications. This paper proposed Fuzzy Classifier-based Vulnerability and Assessment Testing (FCVAPT) model to provide security for sensitive data/information in Web applications. Cross Site Scripting (XSS) and Structured Query Language (SQL) injections were selected for evaluation of proposed FCVAPT model. FCVAPT model's classification performance for MSE, MAPE and RMSE were 33.33, 14.81% and 5.77% respectively. FCVAPT is considerably effective for detecting vulnerability and ascertaining the nature of threats/risks available to Web applications.

Keywords: Vulnerabilities assessment · Penetration testing
Fuzzy classifier-based · Web applications

1 Introduction

The number of active users of the Internet and Web applications is increasing over the years [1]. Web applications and services interface with the Internet resulting in higher degree of security risks. Consequently, the Web application server cannot be ignored, because, Web applications are completely open to global audience. Presently, there exist several approaches for managing security risk for Web applications such as firewall (hardening), defensive coding, monitoring and auditing [2]. Recently, majority of vulnerability assessment methods make use of classification techniques including artificial neural networks, fuzzy control systems and expert systems. In practice, ethical hackers or security experts across the globe usually examine security breaches and possible loopholes by means of Vulnerability Assessment and Penetration Testing (VAPT). VAPT provides an effective means of securing cyber assets [3].

It is believed that, there are over 72 million hosts on the Internet (www.isc.org/ds). By extension, these potentially mean that there could be billions of people within the network neighborhood making the security of network and web application users very difficult to be guarantee. This is due to the vastness of all sorts of people involved. An attacker needs not to be smart or skillful in order to carry out an attack on a specific target. Majority of bigger organizations and users of common computer networks focused investment through direct acquisition of appropriate security methods for computers and network systems.

The study adopted fuzzy classifier in order to categorize the presence of attack. Since fuzzy allow one to work in an uncertain and confusing situation or even an incomplete problem. Although, there are two types of fuzzy framework systems, the study adopted fuzzy inference rule as a platform for its rule base. This research undertook penetration testing of a certain Web application.

2 Related Literature

The use of Web application has increased over times as more services became accessible on the web. Day-by-day, there are more businesses adopting Web applications as means of carrying out transactions. Consequent upon this, the number of attacks on web applications increased tremendously. Studies have shown that the web applications continue to be the main object of attackers, which impacts on data, reputation or financial losses. But, many kinds of countermeasures are available to safeguard systems against attacks such as Intrusion detection system, firewall, and defensive coding [1].

2.1 Computer Vulnerability Analysis

A vulnerability analysis is the process of identifying and quantifying vulnerabilities in an environment. It is an in-depth evaluation of organization's posture, indicating weaknesses as well as providing the appropriate mitigation procedures required to either eliminate those weaknesses or reduce them to an acceptable level of risk [4]. Once a network is secured by fully patching it and deploying antivirus solutions, hackers might still be able to exploit a number of misconfigurations.

Some vulnerability scanners look for signs of known malware based on the computer's behaviour rather than actually scanning the files for known malware signatures. In some cases, this approach can help uncover issues that an antivirus might miss, especially if that malware is being protected by a rootkit. There are vulnerabilities caused by software. Some web services contain known exploits that allow a malicious attacker to use that script as a gateway to send emails, potentially using an organization to launch spam runs; SQL injection exploits might allow an attacker to get hold of usernames and passwords, or inserting his own username, or even to run code remotely. Likewise, the use of applications with known vulnerabilities can open an organization to targeted attacks. Malicious hackers might try to send people malicious payloads targeted at these vulnerable applications that, when triggered, would run the code the hacker would have embedded in the payload sent.

2.2 Web Vulnerability Assessment

[5] developed a Pixy, which is tool providing static analysis proficiencies for the purpose of detecting weaknesses in Web applications. Pixy is developed using the model of open source to enable it discover cross-site scripting weaknesses in PHP scripts. The PHP is prominent for building Web applications thereby causing serious concerns to security consultants. Data flow analysis concerned by experts as means of isolating vulnerabilities. Pixy and six other open sources Hypertext Preprocessor (PHP) solution were used to evaluate vulnerabilities in Web applications. PhpMyAdmin, PhpNuke and Gallery, 36 renowned susceptibilities were remodelled with 27 False Positives. In Simple PHP Blog, Serendipity and Yapig, 15 unacquainted vulnerabilities were identified as having 16 False Positives. However, the Pixy is ineffective for object orientated cases.

[6] developed model called Tainted Mode, which is the prominent input validation for detecting internal vulnerabilities. The author implemented the internal data flows assessment using classical tainted mode model. The information obtained from dynamic analysis is realized through computerized penetration test.

[7] introduced a scanner to discover injection weaknesses. This system conduct searches on websites in order to automatically identify XSS and SQL injection vulnerabilities. The two key components of this system are scanner and spider. Spider traverses the site and search for input points. Scanner begins with injection test and response exploration, which is made up of response analyst and rules author. The author uses VMware work station ACE comprising two hosts for the Web server and the defence server. The system was developed using PHP5 and MySQL, while, the attacks were executed by cURL module.

[8] proposed state violation attacks detection using WebScrab tool called BLOCK, which is a typical stateless application system. Also, the application behaviour model is attained from the client and application interactions (that is, the associations between responses, Web requests and session variables). BLOCK scheme uses two key aspects for identifying state violation based attacks. The training stage models the preferred behaviour through observation of structure of Web request/response, and the variable values corresponding to the session attacks free execution. Identification stage uses the attained model to test each inbound Web request and outbound response, and violation identified is noted.

[9] worked security challenges of Web services because of its distributed and open nature. In general, Web services are prone to cross-site scripting (XSS) attacks by exploiting vulnerabilities. The technique was developed based on fault injection and penetration testing to act out XSS attack across Web services. The author combines with Security Tokens and Web Service Security to effectually identify the sender and thereafter; offer access control authorization to the SOAP messages exchanged. The author conducted the penetration testing using soapUI vulnerability scanner tool, which

involves analysis of the behaviour in a specific situation for new errors or faults on Web services with WSInject.

[10] introduced an automatic black-box tool for recognize reflected XSS and keeping XSS weaknesses in Web applications. The tool depends on interactions of users to conduct its test more efficiently. Firstly, the records of user interactions are made. Thereafter, changes are effected on these interactions in case of attacks. Then, the entire transaction is started all over again on the system. The performance is evaluated and compared with Brup Spider, Spider, Acunetix and w3af on three applications of the Django framework from several setups. The outcomes reveal that the method can recognize more bugs than the listed open source and commercial tools.

2.3 Web Penetration Testing

Penetration testing is a notable practice for assessing a computer system security. At the start of 1970s, the Department of Defence (DoD) for the first time used the technique to demonstrate the security vulnerabilities in computer systems; thereafter it begun a project to develop programs to mitigate the exploitation of identified weaknesses and make systems more safe. On the whole, penetration tests are carried out by institutions in order to ensure the safety of services and information systems but isolate recognized security vulnerability to guard against dubious exploitation by the users [11].

Large data services corporations are generally worried about guarding information and sensitive data. The penetration test runs checks on the security schemes available in companies by means of numerous attacks simulations. The outcomes of this process is to determine areas of fresh infrastructure, application of software updates, installation of software, modification of user policies and application of security patches. The benefits of utilizing penetration testing include: prioritization of security risks, information protection, Financial Loss and security challenges [12].

Penetration test is a process of examining the security of computer or network systems through simulation of professional hacker attacks. The operations of hacker and penetration examiner are similar in many ways except that penetration testing is achieved with a certified professional undertaking a deal with an enterprise. The outcomes are presented as published and circulated reports. The aim of penetration test is to enhance data and information security. The reports of penetration testing comprise series of security information and weaknesses which are confidential and undisclosed to unauthorized individuals until all defects are corrected completely.

3 Methodology

The proposed system architecture comprises the test bed for antecedents, rule applier and consequents, and the checker as shown in Figure 1. The overall structure of the Web application vulnerability model is broken into three main components namely: the input, the FCVAPT Processing (or Test Bed) and the output as shown in Figure 1. The FCVAPT is the most complex component of proposed fuzzy classifier-based

VAPT because the codes, syntax, test bed and datasets are created and stored in order to assist the security experts and developers to ascertain the correct status of the Web applications and rate of susceptibility.

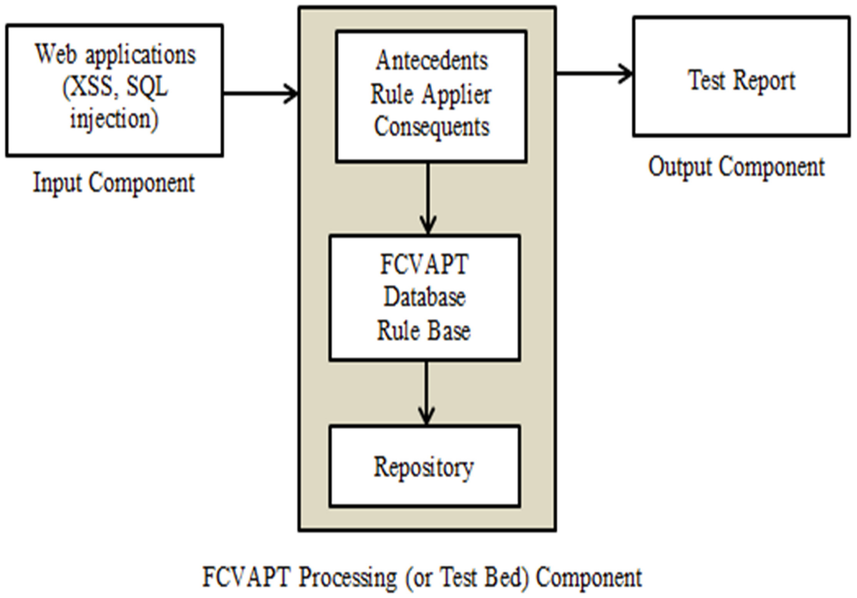
Web applications (XSS, SQL injection)
Input Component
Antecedents Rule Applier Consequents
FCVAPT Database Rule Base
Repository
FCVAPT Processing (or Test Bed) Component
Test Report
Output Component

Fig. 1. Structure of fuzzy classifier-based penetration testing

In Fig. 1, the input component collects the necessary dataset about known and unknown susceptibilities available on Web application referred to as the penetration stage. On the other hand, the VAPT processing or Test Bed is the assessment stage of the proposed technique which is concerned with the detection, recognition and classification of the defined and undefined vulnerabilities of the Web application under consideration.

3.1 Modeling of Threat Penetration Assessments

The risks/threats are normalized to fall within the range of 0–1 because, classification processes easily understand binary representations of entities. Using a scale of 0–1 for the rating score in each category in which the score of 0 depicts the least possible chance with smallest damage, and reverse is the case of rating score of 1 for Web application. Therefore, risk assignment of Web application is with the range of 0–1 in that higher values show the severity of risk to the Web application as shown in Eq. 1.

$$v_j = \frac{1}{q}\sum_{i=1}^{q} k_i \,. \tag{1}$$

where,

v is the vector for risk assessment score ratings for web application (such as worst or good), for $j = 1, 2, 3, \ldots q$
q is the total number of categories for risk analyzed and assessed,
k is the unique risk assessed and analysed components,
j is the matric of estimation or calculations for web application risk levels,
i is individual risk components measured to give the final risk score of the web application, which are damage, reproducibility, exploitability, users affected and discoverability, for $i = 1, 2, 3, \ldots q$.

Upon substitution of the various components described in Eq. 1, which are damage, reproducibility, exploitability, users affected and discoverability, then, it can be expressed as given by Eq. 2:

$$v = \frac{1}{q} \cdot k_1 + k_2 + k_3 + \cdots + k_{q-1} + k_q. \tag{2}$$

where,

q is the vector of individual assessment and analysis components, for $i = 1, 2, 3, \ldots q$.
v is vector of scores for low, fair or high rates and 0, 0.5 or 1 respectively.
k is individual vulnerability threats assessed.

3.2 Experimental Setup

The encoded signatures/features of the two selected Web application threats (or antecedents) are used to develop the truth table for expected fuzzy Rules Base as presented in Table 1.

Table 1. The expected fuzzy decision rules indices for the penetration testing

Rule	Antecedent A	Antecedent B	Consequent C
1	1	1	1
2	1	0.5	1
3	1	0	1
4	0.5	1	1
5	0.5	0.5	1
6	0.5	0	0.5
7	0	1	1
8	0	0.5	0.5
9	0	0	0

In Table 1, the maximum number of rules generated is determined by 3-by-3 membership degrees for the two Web application threats, which serve as antecedent A, antecedent B, and consequent C representing the vulnerability index. The membership function are built on the basis of three conditions for the antecedents such as Little, Much and Biggest. These degrees are arbitrary used to depict different conditions for the two antecedents, whereas the consequent can be described as Lowest, Cautious and Highest impact of vulnerability in Web applications understudies.

3.3 Performance Metrics

This paper adopts four metrics in validating the Fuzzy Classifier VAPT model in Web applications. These are the relative absolute error, mean absolute percentage error, mean square error and root mean square errors performance given by Eqs. 3, 4, 5 and 6.

The Relative Absolute Error (RAE) estimates the rate of incorrectness or deviation of penetration testing value as compared to expected value scores given by Eq. 3:

$$RAE = \left|\frac{x - \hat{x}}{x}\right| \times 100\% \tag{3}$$

$$MSE = \frac{1}{n}\sum\nolimits_{1}^{n} (x - \hat{x})^2 \tag{4}$$

$$MAPE = \frac{1}{n}\sum\nolimits_{1}^{n} \left|\frac{|x - \hat{x}|}{x}\right| (100\%) \tag{5}$$

$$RMSE = \sqrt{\frac{1}{n}\sum\nolimits_{1}^{n} (x - \hat{x})^2} \tag{6}$$

where,

x is the actual or expected penetration test outcome,
$\hat{x}$ is the predicted penetration test outcome,
n is sample size obtained.

4 Results and Discussion

The test bed is generated from the unique signatures encoded for XSS and SQL injections, which represented the antecedent for the classification machine deployed for the VAPT experiment setup I. The XSS signatures and their encoding produces the first antecedent (A). Similarly, the SQL injection signatures and their unique encoded schemes are used to generate the second antecedent (B) stored in the Repository and utilized by the Checker for the developed VAPT model developed.

The Rules Base significant component of the vulnerability assessment and penetration testing which effectively maps the two antecedents to produce logically unique consequents (that is, assesses threats/risks levels based on identified Web applications vulnerabilities. The antecedents were generated from the signatures values for SQL injections and XSS attacks propagated by malicious programmers or users. The fuzzy classifier is used to achieve this mapping mechanisms of antecedents and consequent, AND-logic operations is introduced as shown in Figure 2.

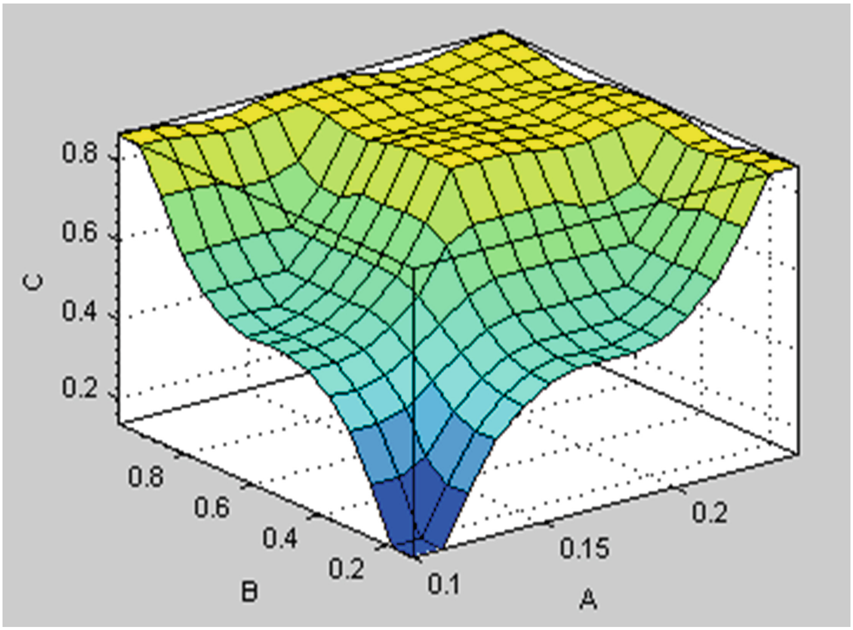

Fig. 2. The 3D representation of the variables A, B and C for VAPT model

In Fig. 2, the threats/risks recognition pattern is relatively stable for conditions between [0.60–0.80]. The recognition error rates for model when tested with 30% of datasets showed stability in the lines of curve for the variable C. The RAE is calculated to be 25% by means of Eq. 3. This implies about 75% precision for analyzed datasets during training of Checker.

This study developed a Vulnerability Assessment and Penetration Testing model by hybridizing fuzzy and intelligent classifiers whose outcomes are analyzed in Table 2.

Table 2. Fuzzy classifier based models outputs

Evaluation metric (%)	FCVAPT
RAE	14.815
MSE	33.333
MAPE	14.815
RMSE	5.774

In Table 2, the RAE determines the extent of closeness of observed outcomes to expected outcomes of the FCVAPT model. This reveals 14.81% deviation of observed variables from actual variables in the outputs realized in FCVAPT model setup. The results of other evaluation parameters significantly improved for the fuzzy classifier when deployed for vulnerability assessment and penetration testing. The performance of FCVAPT model calculated for MSE, MAPE and RMSE are 33.33%, 14.81% and 5.77%

respectively. The implication of these results is that FCVAPT model is capable of detecting vulnerability and estimating threats/risks level associated effectively with Web applications at a particular time.

5 Conclusion

This work identified two most prevalent threats/risks to Web application, which are propagated through malicious insertion of SQL and XSS injection codes. These SQL and XSS injections are discovered to be most dangerous security threats to Web applications due to their ability to be easily deployed malicious along genuine statements on different components such as system, application, operational, network, and physical.

This work proposed a FCVAPT model by introducing intelligent learning scheme known as fuzzy classifier. The model is developed to detect vulnerability in Web applications and concisely state threat or penetration level for recognized cases. The outcomes revealed that the RAE 14.81% deviation for established values of variables considered. Again, FCVAPT model's classification performance for MSE, MAPE and RMSE were 33.33, 14.81% and 5.77% respectively. These results imply that the model is effective for recognizing vulnerability and define the nature of threats/risks available to Web applications during VAPT.

Acknowledgements. We acknowledge the support and sponsorship provided by Covenant University through the Centre for Research, Innovation and Discovery (CUCRID).

References

1. Doshi, J., Trivedi, B.: Comparison of vulnerability assessment and penetration testing. Int. J. Appl. Inf. Syst. **8**(6), 51–54 (2015)
2. Ruse, M.E.: Modelling checking techniques for vulnerability analysis of web applications. Unpublished Ph.D. thesis, Department of Computer Science, Iowa State University, Ames, USA, pp. 1–90 (2013)
3. Aghariya, T.: Security testing on web application. Unpublished M.Eng. thesis, Department of Software Engineering, Charles Darwin University, Darwin, Australia, pp. 1–93 (2015)
4. Samant, N.: Automated penetration testing. Unpublished M.Sc. thesis, Department of Computer Science, San Jose State University, California, USA, pp. 1–69 (2011)
5. Jovanovic, N., Kruegel, K., Kirda, E.: Precise alias analysis for static detection of web application vulnerabilities. In: Proceedings of Programming Languages and Analysis for Security, New York, pp. 27–36. ACM press (2006)
6. Petukhov, A., Kozlov, D.: Detecting security vulnerabilities in web applications using dynamic analysis with penetration testing. Computing Systems Lab, Department of Computer Science, Moscow State University, pp. 1–120 (2008)
7. Chen, J.M., Wu, C.L.: An automated vulnerability scanner for injection attack based on injection point. In: Proceedings of International Computer Symposium, pp. 113–118 (2010)
8. Li, X., Xue, Y.: BLOCK: a black-box approach for detection of state violation attacks towards web applications. In: Proceedings of the 27th Annual Computer Security Applications Conference, pp. 247–256 (2011)

9. Salas, M.I.P., Martins, E.: Security testing methodology for vulnerabilities detection of XSS in Web Services and WS-Security. Electron. Notes Theor. Comput. Sci. **302**, 133–154 (2014)
10. McAllister, S., Kirda, E., Kruegel, C.: Leveraging user interactions for in-depth testing of web applications. In: Proceedings of Recent Advances in Intrusion Detection, pp. 191–210. Springer, Berlin, Heidelberg (2008)
11. Doupe, A.L.: Advanced automated web application vulnerability analysis. Unpublished Ph. D. thesis, Department of Computer Science, University of California, Santa Barbara, USA, pp. 1–227 (2014)
12. Shelly, D.A.: Using a web server test bed to analyse the limitations of web application vulnerability scanners. Unpublished M.Sc. thesis, Department of Computer Engineering, Virginia Polytechnic Institute and State University, Blacksburg, USA, pp. 1–98 (2010)