International Journal of Advanced Technology and Engineering Exploration, Vol 8(77)
ISSN (Print): 2394-5443 ISSN (Online): 2394-7454
http://dx.doi.org/10.19101/IJATEE.2020.762187

Research Article

Development of a browser extension for web application vulnerability detection, avoidance, and secure browsing (VDAS)

Alya Geogiana Buja*, Nurul Syahirah Khairuddin, Noor Afni Deraman, Khyrina Airin Fariza Abu Samah, Mohd Nor Hajar Hasrol Jono and Nor Aimuni MD Rashid
Universiti Teknologi MARA Cawangan Melaka, Faculty of Computer & Mathematical Sciences, Malaysia

Received: 11-October-2021; Revised: 10-April-2021; Accepted: 12-April-2021

Abstract
This paper presents the development of a browser extension for web application vulnerability detection, avoidance and secures browsing. Number of attacks on websites are increasing from time to time. This attack can be happened because of the vulnerabilities exists in application code, perhaps missing validation during the development. Therefore, the aim of this extension is to detect the web application vulnerabilities, which indirectly can provide a secure browsing environment to avoid Internet users from being compromised by attackers. There are four types of web application vulnerabilities considered during the development of the Vulnerability Detection, Avoidance, and Secure Browsing (VDAS) namely Cross Site Scripting (XSS), Structured Query Language injection (SQLi), Local File Inclusion (LCI), and Remote Command Execution (RCE). The VDAS is designed based on data mining approaches. There are five phases involved in developing the VDAS; preliminary study, requirement analysis, system design, system development and system testing. The accuracy of the developed extension was successfully tested and validated by using Vega. In this study, the VDAS was only applied on Google Chrome. Hence, further work is recommended to ensure that the VDAS can be applied on other browsers as well.

Keywords
Browser, Cyber-attack, Cyber security, Extension.

1.Introduction
With the advent of web technologies, the world cannot get away from dealing with various vulnerabilities either in websites or web applications. New technologies developed have introduced new possibilities for exploitations. This problem continues to emerge due to most of web-based application developers often lack background knowledge of internet security. In addition, they are less aware of flaws or weaknesses in the system's design, implementation and security procedures that might result in the system to be easily compromised. Moreover, developers are commonly forced to focus on the functionality for the end-user rather than considering the security analysis of the application developed. Besides, working under pressure in order to accomplish the task has led to poorly developed web applications due to limitation of time for checking the vulnerabilities and flaws inside the codes [1].

Web application vulnerabilities occur because they need to interact with multiple users across multiple networks while at the same time, hackers easily manipulate their accessibility [2, 3]. Attackers can use such flaws to perform Cross-Site Scripting (XSS) in order to capture the user's session identifier and impersonate the user in the application. Cross-Site Scripting (XSS) remains one important cause.

A browser extension is a piece of software that improves the functionality of browsers. The browser can be easily added the extension in order to perform extra functions. Hence, the current study proposes a browser extension which can be implemented on Google Chrome that is capable to enhance security of browsing activities. The two objectives of this study are i) to develop a browser extension for detecting web application vulnerabilities and ii) to evaluate the functionality and accuracy of the developed browser extension.

This paper reviews, web application vulnerabilities, types of web vulnerability scanners and web

*Author for correspondence

application vulnerability detection approaches in Section 2. Section 3 describes the methodology of this study. Findings and results from all tests that were conducted are presented and discussed in Section 4. Section 5 discusses the output of the study and Section 6 concludes this study.

2.Literature review

This section reviews all topics related to this project such as web application vulnerabilities, vulnerability scanners and related works.

2.1Web application vulnerabilities

The Structured Query Language injection (SQLi) is still the most dangerous class of vulnerability in the latest Open Web Application Security Project (OWASP). SQLi is based on injecting strings into database queries. Attackers are capable of executing malicious SQL statements which they can take control of a web application's database server that is commonly known as Relational Database Management System (RDBMS). Attackers may also modify by either adding or deleting the record in a database when they able to inject the SQL injection.

Cross Site Scripting (XSS) is a form of attack whereby users are tricked to run a page that has vindictive content codes [4] that transcendently happens utilizing JavaScript and when a web application utilizes inputs gained from web users without appropriately approving them. When the malicious script code is infused, it will be placed in the web application server prior to exploitation by the attackers. XSS may force to genuine security infringement, including the change of passwords, log out or gather users' data such as their login credentials, sessions, and cookies [5].

Local File Inclusion (LFI) is a path used by attackers to attach files on a server via Hypertext Transfer Protocol (HTTP) request utilizing a web browser to manipulate [6]. The weakness happens when a user provides information without filtering the information given to an inclusive type, for examples, include () and require (). When attackers find out the availability of this vulnerability, they will figure out the list of sensitive files such as /etc/passwd and /etc/user with the intention to misuse them.

Remote Code Execution (RCE) is an attack wherein the attacker's intention will probably infuse their own code, on the host operating system through a weak application or when the application passed a risky user supplied data. This form of attack permits the attacker to insert his own code which the application will further perform.

2.2Vulnerability scanner

A vulnerability scanner is a software application that assesses security vulnerabilities in networks or host systems and creates a bunch of scan results that are frequently utilized to set vulnerability evaluation particularly during the process of development [7–10]. Usually, vulnerability scanner detects vulnerabilities that are originated from vendors, system administration activities, or general activities by users. Vulnerabilities that are originated from vendors are commonly software bugs, web application vulnerabilities, vulnerable services, missing operating system patches, and insecure default configurations.

A network-based scanner is generally executed with respect to a single machine that scans various different hosts on the network by detecting critical vulnerabilities, for example, misconfigured firewalls, risk-related to vendor-supplied software, vulnerable web servers, and dangers related to network and the administration of systems. There are several types of network-based scanners which are port scanners, application scanners, and vulnerability scanners [11]. Port scanners are used to scan and determine the list of open network ports, operating systems and services offered in remote systems [12]. Next, application scanners are used to track weakness of the specific application on the network that can bring high risks to the system. Finally, web vulnerability scanners are likely as port scanners, but they automatically evaluate web sites for finding exploitable SQLi and XSS vulnerabilities [13]. There are several open source tools of port scanners such as Rohrmann [14].

Host-based scanner is implemented on the target computer or remotely controlled by authenticated account access to the target computer. It can detect signs that an attacker has already compromised a system, including looking for suspicious file names, unexpected new system files or device files, and unexpected privileged programs [15].

2.3Related works

Several studies on web application vulnerability detection have been described in this section including web application vulnerability detection the use of the Boyer-Moore string matching algorithm, Rational Unified Process (RUP) model and data mining.

The Boyer-Moore String Matching is utilized to recognize and settle various concerns with respect to

web application vulnerabilities [16–18]. Regular issues of them are false positive and false negatives. False positive takes place when there is a mistake in web applications; notwithstanding, they cannot discover any weakness though the error. In the interim, false negatives are scanners that cannot discover any weakness in web applications and illuminate the user that the web is secure in spite of the fact that there might be weaknesses.

Figure 1 shows the process of the Boyer-Moore String Matching Algorithm. For the SQLi Module, Buffer Overflow Module, Cross Site Scripting, and Cross Site Request Forgery, they will firstly get the Uniform Resource Locator (URL) from the user, scan the server and then operate the process before giving the result. Each of the modules will complete their process from getting the URL, scanning the server and giving out the results before passing them to the next module within the implementation of the matching process using the Boyer-Moore String Matching Algorithm.

The RUP Model is a model executed for recognizing web application vulnerabilities to uncover the presence of errors that exist during a program execution [19]. This is on the grounds that few web application vulnerabilities occur because of input validation and sanitization problems such as SQLi and XSS. Some web designers are not secure-mindful of their web application development which adds to various vulnerable web applications keep consistently expanding. The RUP Model uses the RUP-based Framework, which connects with four phases of the system which are inception, elaboration, construction and transition. The inception will capture the requirements and analyse them. The components that are used in RUP - based models are crawler component, web application attack component, analysis module, and generate reports.

Web crawler acts as an Internet bot that is able to copy all web pages that are visited by users. In this RUP-based Framework vulnerability tool, the web crawler will be used for crawling and gathering all web information and their links into the website. Web crawler gathers information from the web server and at the same time it simulates the attacking code and creates a request to the web server. The web server responds directly to analyse and identify the vulnerability.

Data mining is a technique used to process a large amount of data stored in the data warehouse. This technique has been used for extracting useful information [20]. In this case, data mining was used to detect and clarify vulnerabilities in the code. Data mining starts from gathering data, processing the data, building the model, testing for assessment of the model and finally evaluating the result using the knowledge.

There are two phases of using data mining algorithm for web application vulnerability detection. Firstly, the learning phase and followed by the classification phase. The learning phase extracts the features and defines the rule. Then, the classification phase has a classifier that orders given codes, either vulnerable or non-vulnerable, in view of the feature vectors. The location of SQLi utilizing the Artificial Intelligence (AI) approach [21−25] is proposed in this procedure whereby during the learning phase process, a preparation of the data set is perused by the application from text files. At that point, the information is passed to the learning method of the classifier which will later produce feature vectors from the data obtained by utilizing the strategies separation and tokenizing. The application peruses the test dataset and groups it utilizes the features separated from the learning process in the classification process.

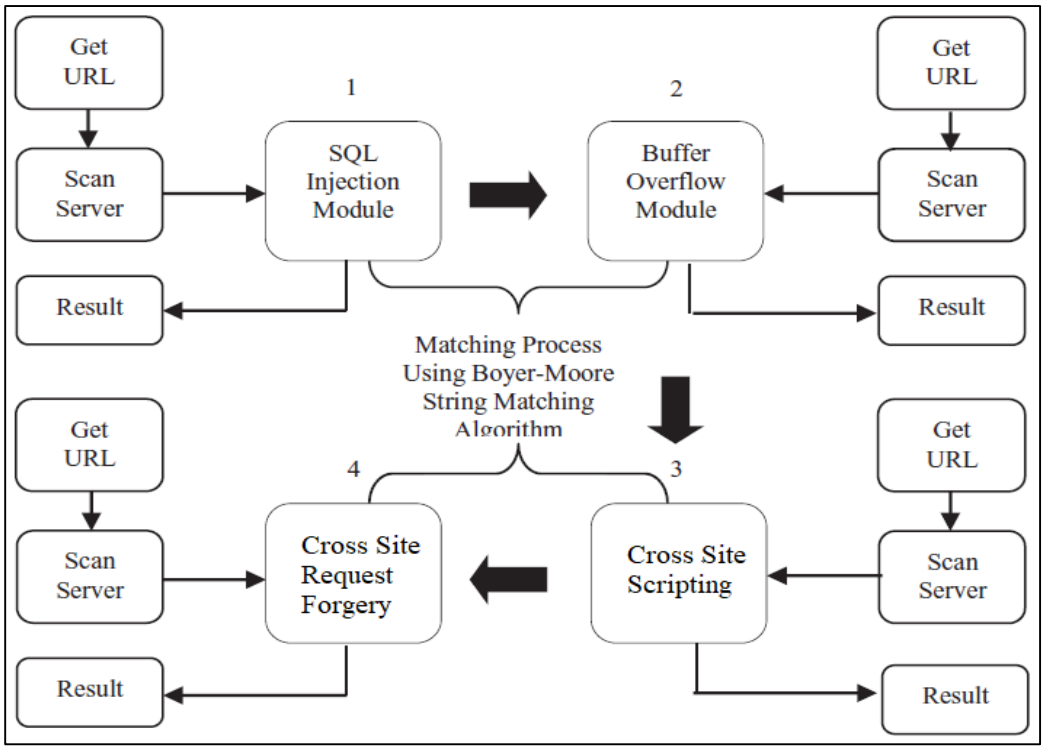

Figure 1 Process of Boyer-Moore string matching algorithm (Source: [16–18])

3.Methods

In order to execute this study, there are five phases involved, namely; preliminary study, requirement analysis, system design and development, and system testing.

3.1Preliminary study

The main activity in this phase is reviewing all literature and research work related to the study. The literature reviewed and analysed were not limited to web application vulnerabilities and techniques of the detection.

3.2Requirement analysis

The second phase of this study is requirement analysis. In this phase, all software and hardware requirement needed for the system development were discovered and identified. A sample of Uniform Resource Locator (URL) that was used for testing was determined. In addition, the most important activity is to identify the existing tool or software for detecting web application vulnerabilities that can be used for result validation.

3.3System design and development

In this phase, the system architecture and user interface were designed. *Figure 2* shows the VDAS architecture. By and large, a vulnerability scanner is comprised of four fundamental parts, namely Scan Engine, Scan Database, Report Module, and User Interface [26].

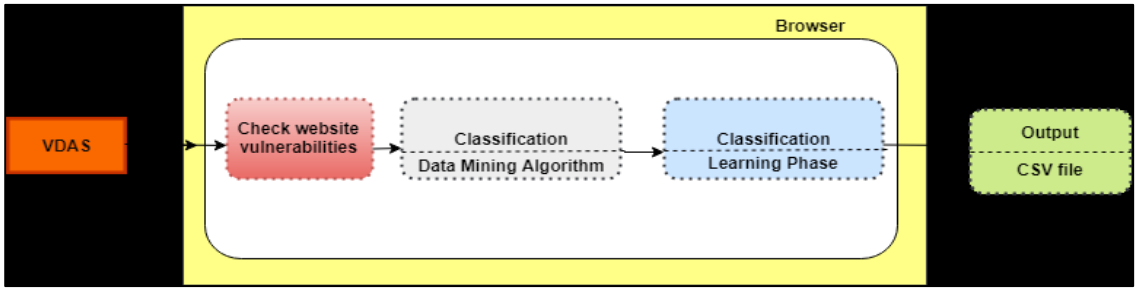

Figure 2 The VDAS model

The VDAS is designed based on data mining, which is used to detect and clarify vulnerabilities in the code to confirm the truth of the vulnerabilities detected. Data mining starts by gathering data, processing the data, building the model, testing for assessment of the model, and finally evaluating the result using knowledge. There are two phases of using data mining algorithm for web vulnerability detection. Firstly, the learning phase and followed by the classification phase. The learning phase extracts the features and defines the rule. Then, the classification phase has a classifier that orders given codes, either vulnerable or non-vulnerable, in view of the feature vectors.

The location of SQLi utilizing the AI approach is proposed in this procedure whereby during the learning phase process, a preparation of the data set is perused by the application from text files. At that point, the information is passed to the learning method of the classifier which will later produce feature vectors from the data obtained by utilizing the strategies separation and tokenizing. The application peruses the test dataset and groups it utilizes the features separated from the learning process in the classification process.

The VDAS interface shows the number of web application vulnerabilities that has been detected and the total number of web vulnerabilities that has been detected. The STOP button will stop the VDAS scanning process and service. Meanwhile, the REFRESH button will refresh the LIST interface pages and the EXPORT button will export the scan results in the CSV file. Users can save that file for future reference and use.

- **VDAS implementation**

This project was developed on Windows desktop-based platform and implemented on Google Chrome. VDAS was developed using the python programming language, Docker and OracleVM VirtualBox. Docker is an open platform software that enables developers and system administrators to create distributed applications. VDAS uses Docker to run the services to users. Moreover, OracleVM VirtualBox was used to do port forwarding as Docker's local machine was run in OracleVM VirtualBox.

- **VDAS installation**

Before the VDAS can be used in a safe browsing, the extension has to be loaded into the chrome://extensions. The VDAS will be loaded into the browser. Once users or web developers browse the websites, the VDAS will automatically scan the websites to check for the existence of website vulnerabilities. While checking for vulnerabilities in that website, the VDAS will classify the type of web vulnerabilities, according to the Data Mining Algorithm and Learning Phase. The results of the scanning will be acknowledged to users and web developers through a CSV file.

3.4System testing

System testing is the most vital phase of this study. There are two main activities conducted in this phase; system testing that consists of functional testing and accuracy testing, as well as results analysis and discussion.

4.Results

This section discusses the results and findings gained from the development of the browser extension for web application vulnerability detection. There were two types of testing used; functionality testing and accuracy testing. Functionality testing focused on the functions of all the buttons and processes in the extension, while accuracy testing focused on the comparison of the VDAS output with the existing tools for web application vulnerability detection called Vega.

4.1Functionality testing

The functionality of the VDAS was tested for three times (T1, T2, T3) based on the menu provided in the developed prototype to make sure the function works correctly. *Table 1* presents the functionality results. The VDAS allows users to export the results obtained into Comma Separated Values (CSV) file which later can be used for further analysis.

Table 1 Functionality test result

Component	T1	T2	T3
START – allows the VDAS to scan the web	Pass	Pass	Pass
LIST - shows scan results	Pass	Pass	Pass
STOP - stops the VDAS' scanning process and service	Pass	Pass	Pass
REFRESH button - refreshes the LIST interface page	Pass	Pass	Pass
EXPORT button - exports the scan results into CSV file	Pass	Pass	Pass

4.2Accuracy testing

The accuracy testing was conducted on both vulnerable and secure websites using the respective URL. In order to validate that the results obtained by the VDAS are correct, the results were then compared with the results obtained from the testing of the same URL on Vega. *Table 2* shows the comparison between the VDAS and Vega.

Table 2 Comparison of Vega' and VDAS' detection result

Type of Vulnerability	Website URL	Vega	VDAS	Accuracy of detection
SQLi	oregon-airsoft.com	Yes	Yes	100
XSS	http://topupchat.com/old/profile.php?id=11290	Yes	Yes	100
LFI	http://localhost/computer_club_system/	No	No	100
RCE	http://localhost/bWAPP/bWAPP	No	No	100

5.Discussion

The developed browser extension has been able to detect four defined web application vulnerabilities. All functions have been executed successfully on the Google Chrome. Based on *Table 2*, the results obtained by both tools are the same. All testing conducted on each URL and using both tools was executed three times to ensure the accuracy of the detection.

Based on *Figure 3*, the VDAS browser extension is easy to be used by the user. Once the user has typed an URL, then the user has to click on the extension and press the 'START' button. Once the button pressed, the VDAS will immediately scan the website.

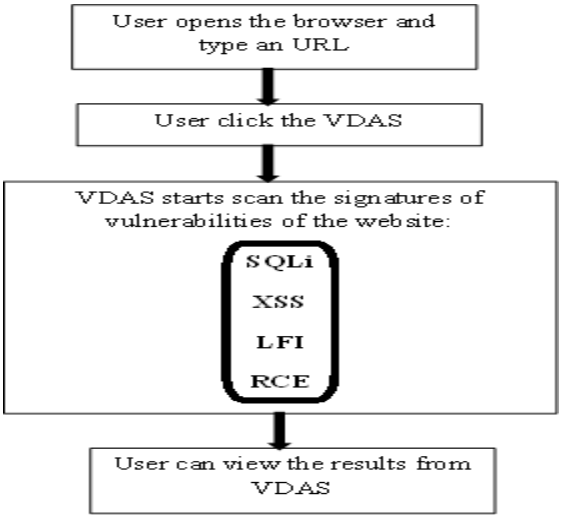

Figure 3 System architecture

The VDAS will search for the signatures of the defined web application vulnerabilities. The VDAS is a browser extension that is capable to detect websites with vulnerabilities that have a high percentage and probability of being exploited and attacked. The VDAS can detect Structured Query Language injection (SQLi), Cross Site Scripting (XSS), Local File Inclusion (LCI) and Remote Code Execution (RCE) vulnerabilities. After the scanning process completed, the result will be able to be viewed by the user. All the detected vulnerable websites will be stored in the list of scan results to allow users and web developers to view and stop from browsing those sites in the future.

5.1Limitation

This study is limited to four types of web application vulnerabilities; SQLi, XSS, LFI and RCE only. In addition, all testing of this study has been conducted using Google Chrome.

6.Conclusion and future work

In this research, the main objective was to develop a browser extension for web application vulnerability detection, avoidance and secure browsing. Thus, this research attempted to detect the web application vulnerabilities by using data mining technique. The extension was designed and developed to detect Cross Site Scripting (XSS), Structured Query Language injection (SQLi), Local File Inclusion (LCI), and Remote Command Execution (RCE). In conclusion, with the development of VDAS, the researchers hope that Internet users can have a secure web browsing. The browser extension was tested in Google Chrome for functionality and accuracy. Both testings', including Vega, provided a significant impact. For future work, this browser extension can be improvised to run successfully in other browsers as well. In addition, the extension can be further developed by improving the detection algorithm to detect other web application vulnerabilities.

Acknowledgment

Sincere appreciation goes to Universiti Teknologi MARA Cawangan Melaka for the support given to this research endeavour, TEJA: Internal Grant (GDT2020-20).

Conflicts of interest

The authors have no conflicts of interest to declare.

References

[1] Makino Y, Klyuev V. Evaluation of web vulnerability scanners. In 8th international conference on intelligent data acquisition and advanced computing systems: technology and applications 2015 (pp. 399-402). IEEE.

[2] www.owasp.org/index.php/Top_10_2017-Top_10. Accessed 04 January 2020.

[3] https://owasp.org/www-project-top-ten/. Accessed 04 January 2021.

[4] Awoleye OM, Ojuloge B, Ilori MO. Web application vulnerability assessment and policy direction towards a secure smart government. Government Information Quarterly. 2014; 31:S118-25.

[5] Huang C, Liu J, Fang Y, Zuo Z. A study on Web security incidents in China by analyzing vulnerability disclosure platforms. Computers & Security. 2016; 58:47-62.

[6] Begum A, Hassan MM, Bhuiyan T, Sharif MH. RFI and SQLi based local file inclusion vulnerabilities in web applications of Bangladesh. In international workshop on computational intelligence 2016 (pp. 21-5). IEEE.

[7] Smeets YR. Improving the adoption of dynamic web security vulnerability scanners. Radboud University, NL. 2015.

[8] Surian RU, Abd Rahman NA, Nathan Y. Nscanner: vulnerabilities detection tool for web application. In journal of physics: conference series 2020 (pp.1-9). IOP Publishing.

[9] Touseef P, Alam KA, Jamil A, Tauseef H, Ajmal S, Asif R, et al. Analysis of automated web application security vulnerabilities testing. In proceedings of the international conference on future networks and distributed systems 2019 (pp. 1-8).

[10] Jan S, Panichella A, Arcuri A, Briand L. Search-based multi-vulnerability testing of XML injections in web applications. Empirical Software Engineering. 2019; 24(6):3696-729.

[11] Bairwa S, Mewara B, Gajrani J. Vulnerability scanners-a proactive approach to assess web application security. International Journal on Computational Sciences & Applications. 2014.

[12] Ahanger TA. Port scan-a security concern. International Journal of Engineering and Innovative Technology. 2014; 3(10):241.

[13] Erturk E, Rajan A. Web vulnerability scanners: a case study. arXiv preprint arXiv:1706.08017. 2017.

[14] Rohrmann RR. Large scale anonymous port scanning. University of Arizona. 2017.

[15] https://resources.infosecinstitute.com/topic/the-art-of-network-vulnerability-assessment/. Accessed 04 January 2021.

[16] Saleh AZ, Rozali NA, Buja AG, Jalil KA, Ali FH, Rahman TF. A method for web application vulnerabilities detection by using Boyer-Moore string matching algorithm. Procedia Computer Science. 2015; 72:112-21.

[17] Buja G, Abd Jalil KB, Ali FB, Rahman TF. Detection model for SQL injection attack: An approach for preventing a web application from the SQL injection attack. In symposium on computer applications and industrial electronics 2014 (pp. 60-4). IEEE.

[18] Rahman TF, Buja AG, Abd K, Ali FM. SQL injection attack scanner using Boyer-Moore string matching algorithm. JCP. 2017; 12(2):183-9.

[19] Gol D, Shah N. Detection of web appication vulnerability based on RUP model. In national conference on recent advances in electronics & computer engineering 2015 (pp. 96-100). IEEE.

[20] Vithanage NM, Jeyamohan N. WebGuardia-An integrated penetration testing system to detect web application vulnerabilities. In international conference on wireless communications, signal processing and networking 2016 (pp. 221-7). IEEE.

[21] Zech P, Felderer M, Breu R. Knowledge-based security testing of web applications by logic programming. International Journal on Software Tools for Technology Transfer. 2019; 21(2):221-46.

[22] Naeem H. Detection of malicious activities in internet of things environment based on binary visualization and machine intelligence. Wireless Personal Communications. 2019; 108(4):2609-29.

[23] Naeem H, Guo B, Naeem MR, Ullah F, Aldabbas H, Javed MS. Identification of malicious code variants based on image visualization. Computers & Electrical Engineering. 2019; 76:225-37.

[24] Mantra IG, Hartawan MS, Saragih H, Abd Rahman A. Web vulnerability assessment and maturity model analysis on Indonesia higher education. Procedia Computer Science. 2019; 161:1165-72.

[25] Marashdih AW, Zaaba ZF, Suwais K, Mohd NA. Web application security: an investigation on static analysis with other algorithms to detect cross site scripting. Procedia Computer Science. 2019; 161:1173-81.

[26] Nurmyshev S, Kozhakhmet K, Atymtayeva L. Architecture of web based intellectual vulnerability scanners for OWASP web application auditing process. Int. Journal AETA, NSP. 2016; 5(3):51-5.

Alya Geogiana Buja is a Senior Lecturer at the Faculty of Computer and Mathematical Sciences in Universiti Teknologi MARA (UiTM) Cawangan Melaka. She is a PhD holder in the field of Information Security and graduated from Universiti Teknikal Malaysia Melaka (UTEM), MSc in Computer Science and BSc in Netcentric Computing from Universiti Teknologi MARA (UiTM). Her research interests are Networking and Information Security, Cryptanalysis and Cyber Security.

Email: geogiana@uitm.edu.my

Nurul Syahirah Khairuddin is a BSc holder in Data Communication and Networking from Universiti Teknologi MARA (UiTM) Cawangan Melaka Kampus Jasin. She has Diploma in Computer Science. Her research interests are Web Application Vulnerabilities and Network Security.
Email: syahirahkhairuddin1996@gmail.com

Noor Afni Deraman is currently a lecturer at Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA (UiTM) Melaka. She obtained her Bachelor's degree in Computer Science majoring in Software Engineering from Universiti Sains Malaysia (USM). She completed her Master's degree in Science (Computer Science) in 2006. Her research interest is in Data Science and Artificial Intelligence.
Email: noora465@uitm.edu.my

Khyrina Airin Fariza Binti Hj Abu Samah is a senior lecturer from Universiti Teknologi MARA (UiTM), Melaka Jasin Campus. She has 13 years of working experience in the Information Technology field in the semiconductor industry before joining UiTM. She has Diploma, Bachelor's Degree and Master's Degree in Computer Science and PhD in Information Technology. Her research interest in Artificial Intelligent, Operational Research, Algorithm Analysis, Clustering and Optimization, Evacuation Algorithm, Internet of Things (IoT) and Sentiment Analysis.
Email: khyrina783@uitm.edu.my

Mohd Nor Hajar Hasrol Bin Jono was born in 1978 in Klang, Selangor. He obtained his PhD in 2016. At present, he is working as a senior lecturer at the Faculty of Computer Science and Mathematics at Universiti Teknologi MARA (UiTM) Melaka, Malaysia. Now in the field of administration, he currently holds the position of Deputy Rector of Student Affairs UiTM Melaka. Previously, he was the Head of Training Division, Head of Systems Division and also Fellow at the i-Learn Center under the Academic Affairs Division, UiTM Shah Alam.
Email: hasrol@uitm.edu.my

Nor Aimuni Md Rashid is a lecturer at the Faculty of Computer and Mathematical Sciences UiTM Melaka. Before joining UiTM, she worked with Pay-Point Sdn Bhd in the position of System Programmer fo 2 years. She is a MSc holder in Information Technology from Universiti Kebangsaan Malaysia (2015) and BSc in Netcentric Computing from UiTM Shah Alam, Malaysia (2010). Her research interests are in Multi-Agent System, Distributed Computing, Cryptography and IoT.
Email: aimuni5294@uitm.edu.my