A Survey of Tools and Techniques for Web Attack Detection

Hoang Xuan Dau, Ninh Thi Thu Trang, Nguyen Trong Hung

***Abstract*—Web attacks include types of attacks to websites and web applications to steal sensitive information, to possibly disrupt web-based service systems and even to take control of the web systems. In order to defend against web attacks, a number of tools and techniques have been developed and deployed in practice for monitoring, detecting and preventing web attacks to protect websites, web applications and web users. It is necessary to survey and evaluate existing tools and techniques for monitoring and detecting web attacks because this information can be used for the selection of suitable tools and techniques for monitoring and detecting web attacks for specific websites and web applications. In the first half, the paper surveys some typical tools and techniques for monitoring and detecting web attacks, which have been proposed and applied in practice. The paper's later half presents the experiment and efficiency evaluation of a web attack detection model based on machine learning. Experimental results show that the machine learning based model for web attack detection produces a high detection accuracy of 99.57% and the model has the potential for practical deployment.**

***Tóm tắt*—Tấn công web gồm các dạng tấn công vào các website và ứng dụng web nhằm đánh cắp các thông tin nhạy cảm, có thể gây ngưng trệ hệ thống dịch vụ, hoặc chiếm quyền kiểm soát hệ thống. Để phòng chống tấn công web, nhiều kỹ thuật và công cụ đã được nghiên cứu, phát triển và ứng dụng trong thực tế phục vụ giám sát, phát hiện và ngăn chặn dạng tấn công này nhằm bảo vệ các website, ứng dụng web và người dùng web. Việc khảo sát, đánh giá các công cụ và kỹ thuật giám sát, phát hiện tấn công web hiện có là cơ sở cho lựa chọn công cụ, kỹ thuật phát hiện tấn công web phù hợp cho các hệ thống website, ứng dụng web cụ thể. Trong phần đầu, bài báo này khảo sát một số công cụ và kỹ thuật giám sát, phát hiện tấn công web tiêu biểu đã được nghiên cứu, phát triển và ứng dụng trên thực tế. Phần sau của bài báo trình bày nội dung thử nghiệm, đánh giá hiệu quả của một mô hình phát hiện tấn công web dựa trên học máy. Các kết quả thử nghiệm cho thấy, mô hình phát hiện tấn công web dựa trên học máy cho độ chính xác đạt tới 99.57%, có tiềm năng triển khai hiệu quả trên thực tế.**

Keywords— Web attack; Web attack detection; Web attack detection based on signatures; Web attack detection based on machine learning.

Từ khóa— Tấn công web; Phát hiện tấn công web; Phát hiện tấn công web dựa trên dấu hiệu; Phát hiện tấn công web dựa trên học máy.

I. INTRODUCTION

Along with the rapid development of the Internet and the global Web, websites and web applications (referred to as *web applications* from now on) have also grown very strongly and have become one of the most popular and essential applications on the Internet [1][2]. Web applications are used in almost all areas of social life such as business, commerce, finance - banking, manufacturing, sports, entertainment and social communication. Because of their popularity and importance, websites and web applications are also subject to many types of dangerous and sophisticated attacks and intrusions, aiming to steal sensitive information and disrupt operations, or take control of the web system. Types of popular attacks on websites and web applications (referred to as *web attacks* from now on) can be listed as SQL injection (SQLi), Cross-Site Scripting (XSS), Command injection (CMDi), path traversal, defacements and DoS/DDoS [1][2]. Our attacks can cause serious consequences to websites, web applications, as well as web users. They allow the attacker to (i) bypass the authentication system of the web application, (ii) carry out unauthorized modifications to web databases and content of web pages, (iii) extract data from web databases, (iv) steal sensitive information from webservers and user web browsers and (v) take control of web and database servers [3]. Figure 1 shows an

example of a SQLi attack to a website, in which the Attacker inserts the malicious SQL code into the search keyword to delete a table of the website's Database.

Figure 1. Example of a SQLi attack

Due to the danger of web attacks, a number of tools and techniques have been developed and applied in practice to monitor, detect and prevent this kind of attack to protect websites, web applications and web users. In general, there are three approaches to web attack defense, including (a) validating input data, (b) reducing the number of attackable surfaces, and (c) adopting a defense in depth strategy [1][2]. In particular, approach (a) requires a thorough examination of all input data and only valid data is passed to the next processing stage to reduce types of malicious code embedded in the input data. According to approach (b), the web application is divided into components and appropriate access control mechanisms are applied to restrict user access to each component in order to reduce the number of attackable locations. In approach (3), many layer defenses are used to form complementary layers to detect and prevent web attacks in order to better protect websites, web applications and web users [3]. Each of these three approaches mentioned above has many tools and techniques that have been researched and developed. Therefore, the survey and evaluation of existing web attack detection tools and techniques is necessary and is the basis for the selection of appropriate tools and techniques for monitoring and detecting web attacks for the practical deployment in specific websites and web applications.

This paper carries out two main tasks which are also two contributions as follows:

1. Investigate typical web attack detection and monitoring tools and techniques that have been developed and applied. This is a necessary task and the survey results can be used as a basis for selecting suitable web attack detection and monitoring tools and techniques for deployment in specific websites and web applications; and

2. Test and evaluate the effectiveness of web attack detection techniques based on traditional supervised machine learning algorithms to find the best-supervised machine learning algorithm for the web attack detection model. This is an important premise for building a web attack detection system for practical applications.

The rest of the paper includes the following sections: Section II describes typical types of security vulnerabilities and web attacks, Section III examines tools and techniques for monitoring and detecting web attacks, Section IV describes and tests a machine learning-based web attack detection model and Section V is the conclusion of the paper.

II. COMMON WEB VULNERABILITIES AND ATTACKS

According to OWASP, the reason that attackers can perform various types of web attacks is that they exploit the types of security holes that exist in websites and web applications [1]. OWASP periodically publishes a list of security vulnerabilities and the most recent list of web vulnerabilities is released in 2017. Table I lists the OWASP Top 10 Web Security Vulnerabilities in 2017 [1].

TABLE I. OWASP TOP 10 2017

Vulnerability	Description
A1-Injection	The code injection error allows the insertion and execution of various types of malicious code, such as SQL, OS and LDAP.
A2-Broken Authentication	Weak application authentication or session authentication failure that allows the theft of password, encryption keys, or impersonation/user session hijacking
A3-Sensitive Data Exposure	This error allows the theft of sensitive data because they are not properly protected, or by appropriate measures.
A4-XML External Entities (XXE)	Error handling XML files allows execution of malicious code embedded in externally referenced XML files.
A5-Broken Access Control	Weak access control error allows unauthorized access to functionality, or data, like accessing other users' accounts, viewing sensitive files...
A6-Security Misconfig-uration	Improper security configuration error, such as insecure default configuration, incomplete or special configuration...

Vulnerability	Description
A7-Cross-Site Scripting (XSS)	XSS code injection error allows HTML or JavaScript code to be inserted to steal sensitive data from web users' browsers.
A8-Insecure Deserialization	Unsafe deserialization can lead to remote code execution, replay attacks, or privilege escalation.
A9-Using Components with Known Vulnerabilities	Components, libraries containing known vulnerabilities used in applications and running with application privileges can be easily exploited to attack the system.
A10-Insufficient Logging & Monitoring	Inadequate logging and monitoring, coupled with poor incident response allow further system attacks, while maintaining persistence, pivoting to more systems, and tampering, extracting, or destroying data.

In addition to the list of web security vulnerabilities, OWASP also provides a list of typical attacks on websites and web applications as described in Table II.

TABLE II. TOP 5 COMMON WEB ATTACKS

Types of Attacks	Description
SQLi	SQL injection attack, in which SQL exploit code is inserted into user data submitted to webserver and finally executed on the website's database server.
XSS	XSS injection attack, where an XSS exploit (usually JavaScript) embedded in a web page runs within the browser with user access privilege, can access sensitive user information stored in the browser.
Defacements	The attack changes the content and thereby changes the appearance of the web page. This type of attack causes damage to the website owner.
Path traversal	The attack exploits path traversal, or normalization, errors in webservers and web applications.
Session hijacking	The attack exploits web session security flaws, in which an attacker can access a user's session and perform the same operations as the user himself.

III. TOOLS AND TECHNIQUES FOR MONITORING AND DETECTING WEB ATTACKS

A. TOOLS AND SOLUTIONS FOR MONITORING AND DETECTING WEB ATTACKS

This section examines some commercial and open-source solutions and tools for monitoring and detecting typical attacks on websites and web applications. Many solutions and tools are used in practice, such as VNCS Web Monitoring [4], Nagios Web Application Monitoring Software [5], Site24x7 Website Defacement Monitoring [6], ModSecurity [7], Snort IDS [8]. This section examines three widely used solutions and tools, including VNCS Web Monitoring [4], Nagios Web Application Monitoring Software [5] and ModSecurity [7].

1. VNCS Web Monitoring

VNCS Web monitoring [4] is a solution that allows monitoring of many websites simultaneously based on collecting, processing and analyzing access logs using Splunk platform developed by Vietnam Cybersecurity Technology Joint Stock Company (VNCS).

VNCS Web monitoring collects weblogs from servers to be monitored, then transfers them to the central system for processing and analysis. This system allows for centralized log management, has automatic analysis and real-time alerting, supports manual log analysis to find problems, supports monitoring and alerting the operating status of the website, supports the detection of attacks to change content, change the interface, detection of SQLi, XSS and malicious code on websites.

The limitation of VNCS Web monitoring is that the problem of transporting large volumes of logs from the monitored servers to the processing center requires a stable connection with high throughput. In addition, this is a commercial solution, so the installation and operating costs are relatively high.

2. Nagios Web Application Monitoring Software

Nagios Web Application Monitoring Software [5] is a toolkit that allows monitoring websites, web applications, web transactions and web services, including features such as availability monitoring, URL monitoring, HTTP status monitoring, content monitoring, website hijacking detection, and SSL certificate monitoring. Nagios provides many tools to help configure objects to be monitored easily and quickly, such as Website Monitoring Wizard, Website URL Monitoring Wizard, Website Transaction Monitoring Wizard, HTTP Monitoring Plugins.

The main drawback of Nagios is that it is a commercial solution, so installation and operating costs are high.

3. ModSecurity

ModSecurity [7] is a type of web application firewall (WAF) that is used to filter user queries sent to webservers, as shown in Figure 4. ModSecurity uses a set of predefined rules to filter HTTP requests. If the request is determined to be valid, it is sent to the webserver, otherwise, the request is blocked and a warning is logged. ModSecurity can be installed on the server to protect multiple websites, preventing most types of web attacks, such as SQLi and XSS.

The advantage of ModSecurity over commercial WAF solutions (such as Barracuda Networks WAF, Imperva SecureSphere, and Radware AppWall) is that it is open, free, lightweight, and deeply integrated into webservers, like the Apache webserver. The limitation of ModSecurity is its compatibility with webservers. ModSecurity may work fine with the Apache webserver, but may have problems with the Microsoft IIS webserver.

4. Comments

In general, solutions and tools for monitoring and detecting web attacks are relatively diverse, including solutions and tools for monitoring, detecting web attacks, anomalous access behaviors, as well as web application firewalls, or intrusion detection systems. However, most of the solutions and tools mentioned above perform signature-based, or rule-based monitoring and detection. Therefore, they are only capable of detecting known attacks. In addition, some commercial solutions and tools have high installation and operating costs. Some solutions, like ModSecurity, have compatibility issues with different webserver platforms.

B. TECHNIQUES FOR DETECTING WEB ATTACKS

This section examines a number of studies on monitoring techniques to detect typical attacks on websites and web applications. The surveyed studies fell into two groups: (i) the detection group based on the signatures, patterns, or rules and (ii) the detection group based on the anomaly.

1. Detection based on signatures and rules

Detecting attacks in general and detecting web attacks in particular based on signatures, or rules, is an attack detection method based on matching a set of signatures of known attacks with collected monitoring data. An attack is detected when at least one signature matching is successful. The signature-based attack detection technique has the advantage of being able to quickly and accurately detect known attacks. However, this technique's disadvantage is not able to detect new types of attacks or attacks that exploit zero-day vulnerabilities because their signatures have not been updated on the database. In addition, building and updating the signature database are often done manually, which is labor-intensive.

Many researches and proposals for signature-based web attack detection techniques have been published in specialized journals and conferences over last decade [9]. This section introduces three typical proposals for rule-based web attack detection, including OWASP ModSecurity Core Rule Set (CRS) [10], SQL-IDS [11] and XSS-GUARD [12].

CRS [10] is a set of rules developed by the OWASP project to detect many types of web attacks listed in the OWASP Top 10 with low false alarm rates. CRS can be used with ModSecurity – the module included with the Mozilla Apache webserver. The advantage of CRS is that it is supported and regularly updated by OWASP and the global web security community. However, because the CRS includes a fairly large number of rules, it is relatively cumbersome and can have compatibility problems when integrated into other web application firewalls, or used with other webservers, such as the Microsoft IIS webserver.

SQL-ID [11] is a specification-based SQLi attack detection system. SQL-ID first constructs a rule set that specifies the structure of valid SQL commands generated by the web application and sends it to the database server for execution. Then, SQL-ID performs monitoring, pre-processing and classification of the incoming SQL queries based on the built specification rule set. Only SQL statements classified as "valid"

are sent to the database server for execution, and invalid SQL statements are blocked and logged. Experiments show that SQL-ID achieves 0% false alarms and can be used to protect multiple websites simultaneously because the system is deployed as an intermediary between the webserver and the database server. However, the manual building of the rule set specifying the structure of valid SQL statements must be done separately for each website and consumes a lot of time, especially for large-scale websites and web applications.

XSS-GUARD [12] is a framework that allows monitoring and preventing XSS attacks by creating a "shadow page" and comparing the shadow page with the real page before sending the real page to the client. The shadow page is automatically generated side-by-side with the real page from the webserver response, but uses clean input data (no scripting) of the same length as the real data. Tests show that XSS-GUARD is capable of blocking many forms of XSS attacks listed by OWASP. Moreover, this method does not require a large rule set and frequent updates. However, XSS-GUARD significantly increases the load on the webserver because it has to continuously generate and compare shadow pages with real pages on each request from the web user.

2. Detection based on anomaly

Anomaly-based attack detection is based on the assumption that attack behaviors are often closely related to anomalous behaviors. The process of building and deploying an anomaly-based attack detection system consists of two phases: (1) training and (2) detection. During the training phase, the object's profile in normal working mode is built. During the detection phase, the current behavior of the system is monitored and a warning is fired if there is a sharp difference between the current behavior and the behavior recorded in the object's profile. The advantage of anomaly-based attack detection is that it has the potential to detect new types of attacks without requiring prior knowledge about them. However, this method has a relatively high false alarm rate compared to the signature-based detection method. In addition, this technique is also resource intensive for object profiling and current behavior analysis.

As can be seen, the most important task to be done in most anomaly-based attack detection proposals is building a profile of the object to be monitored in normal working mode. There are many methods for profile construction, including statistics, finite state machines, and machine learning, of which statistics and machine learning are the most widely used methods [13]. Therefore, this section focuses on investigating some proposed web attack detection based on statistics and machine learning, including Bearte et al. [14], Torrano-Gimenez et al. [15], Liang et al. [16], Pan et al. [17] and Hoang [3].

Betarte et al. [14] propose to use 1-class classification models and n-gram-based classification models to improve the detection ability and accuracy of website attack detection for ModSecurity [7]. Experiments performed included (1) detection using only OWASP CRS [10], (2) detection using only 1-class classification model, or n-gram-based classification model, and (3) combined detection using OWASP CRS and machine learning models. Experimental results on CSIC2010 [18], ECML/PKDD2007 and self-generated datasets show that the machine learning and association models have much higher accuracy than OWASP CRS.

Torrano-Gimenez et al. [15] propose to build an anomaly-based detection system. The system is installed as a proxy server or a web application firewall that stands between the client and the webserver. During the training phase, an XML file describing the set of valid requests for a website is automatically generated from the training data by a statistical method. Then in the detection phase, the XML file is used to categorize the requests sent to the webserver. If the request is determined to be normal, it is forwarded to the webserver for processing. Otherwise, requests that are determined to be anomalous will be blocked. Experiments show that the proposed system gives high accuracy and low error rate when the amount of training data is large enough (from 10,000 requests).

Liang et al. [16] propose to use the RNN deep learning network to build web attack detection models. Experiments on the CSIC 2010 dataset [18] show that the proposed model achieves an overall accuracy of over 98%. Moreover, this method can eliminate the manual and time-consuming work including selecting and extracting features for training and detection.

Using a fairly similar approach, Pan et al. [17] suggest using the Robust Software Modeling Tool to monitor and extract the execution information of an application. Then use the collected information to train the stacked denoising autoencoder deep learning network to build the detection model. Experiments show that the proposed method can detect many types of web attacks with an average F1 measure of over 90%.

Hoang [3] proposes a machine learning-based detection model of web attacks using weblogs, in which a decision tree machine learning algorithm is used to build a detection model from the training data extracted from weblog. The experimental results on the HTTP Param Dataset dataset [19] and real weblog data set show that the proposed model is capable of effectively detecting 4 common web attacks including SQLi, XSS, CMDi and path traversal with an overall accuracy of 98.56%. In addition to the advantage of high accuracy, the proposed model also requires lower computational resources than the recommendations based on deep learning networks [16][17].

3. Comments

From the above surveys, some comments about web attack detection proposals can be drawn as follows:

- Proposals based on signatures and rules are capable of quickly and accurately detecting known types of web attacks. However, building and updating the set of signatures and rules manually consume a lot of time [10][11];
- Anomaly-based proposals, especially those based on machine learning give high detection accuracy and the ability to detect new attacks. Furthermore, building profiles or detection models can be done automatically from the training data. Nevertheless, anomaly-based proposals often have a relatively high false alarm rate compared to signature-based detection. In addition, they also require a lot of computational resources for detection model building [14][15][16][17];
- Some proposals are only capable of detecting one type of web attack. In addition, some suggest using comparison or monitoring mechanisms that can severely degrade server performance [12][17].

IV. EXPERIMENTS ON DETECTING WEB ATTACKS BASED ON MACHINE LEARNING

In this section, we evaluate the common web attack detection model proposed in [3] by examining the detection performance of the model using various supervised machine learning algorithms. On that basis, the machine learning algorithm that gives the best detection performance is selected to develop a web attack detection system for practical deployment.

A. INTRODUCTION TO THE DETECTION MODEL

The web attack detection model [3] is implemented in 2 phases: (1) the training phase and (2) the detection phase. The training phase as shown in Figure 2 includes the following steps:

- Collect training data, including normal URIs (Uniform Resource Indicators) and attack URIs;
- Training data is preprocessed to select and extract features. After preprocessing, each URI is converted to a vector and the training dataset is converted to a training matrix of $M \times (N+1)$ elements, where M is the total number of URIs in the training set. training and N is the characteristic number. The last column of the training matrix stores the label of the URI.
- Training data in the form of a training matrix is included in the training phase to

build a classifier, or a model used for the detection phase.

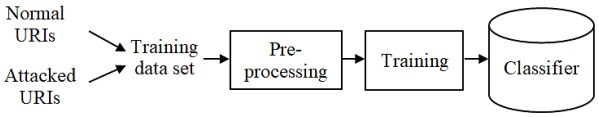

Figure 2. Web attack detection model: Training phase

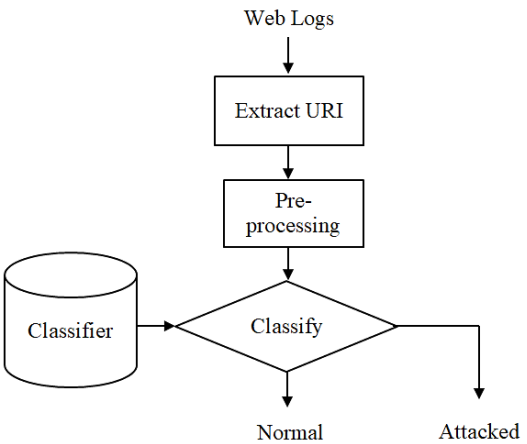

Figure 3. Web attack detection model: Detection phase

The detection phase as depicted in Figure 3 includes the following steps:

- URIs extracted from weblog data are used as input for discovery;
- Each URI is preprocessed as it is done with the training URI. The result of the preprocessing is the vector of the URI to use for the next step;
- The URI's vector is classified using the Classifier built during the training phase. The result of this step is the status of the URI: Normal or Attacked.

B. Experimental dataset and pre-processing

1. *Experimental dataset*

The dataset used for the test is the HTTP Param Dataset [19] consisting of 31,067 URIs of web requests, including the URI length and label. There are 2 types of URI labels, Norm (Normal) and Amon (Attacked). The Anom label includes four types of attacks: SQLi, XSS, CMDi, and path traversal. The dataset is distributed by label as follows: 19,304 normal strings, 10,852 SQLi strings, 532 XSS strings, 89 CMDi strings and 290 path traversal strings. All the records of the dataset are used for training and testing the detection model using different machine learning algorithms, with 80% of the records randomly selected for training and 20% remaining for testing.

2. *Data pre-processing*

The preprocessing performs the separation and vectorization of the features of URIs. This stage includes 2 tasks:

- Separation of the URI features using the n-gram technique. The n-gram technique was chosen because of its simplicity and fast execution. We choose 3-gram to separate the URI features;
- Vectorize the URI features using the IF-IDF (Term Frequency-Inverse Document Frequency) method. For each 3-gram, the *tf-idf* value is calculated as follows:

$$\mathrm{tf}(t,d) = \frac{\mathrm{f}(t,d)}{\max\{\mathrm{f}(w,d) : w \in d\}} \quad (1)$$

$$\mathrm{idf}(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|} \quad (2)$$

$$tf\text{-}idf(t, d, D) = tf(t, d) \times idf(t, D) \quad (3)$$

where, $tf(t, d)$ is the frequency of 3-gram t in URI d; $f(t,d)$ is the number of times 3-gram t occurs in the URI d; $\max\{f(w,d): w \in d\}$ is the maximum number of occurrences of any 3-gram in the URI d; D is the set of all URIs and N is the total number of URIs. Because the number of URI features (3-gram number) is quite large, the PCA (Principal Component Analysis) method is used to reduce the number of features to 256 - experimentally selected.

C. Model training and testing

The training phase uses supervised machine learning algorithms supported by the SkLearn library in Python to build and test the detection model. The supervised machine learning algorithms used include Naïve Bayes, support vector machines (SVMs), decision trees, and

random forests. These are traditional machine learning algorithms that are widely applied in many fields, including information security [20][21]. With the SVM algorithm, Linear and RBF kernel functions are used [20]. For each machine learning algorithm, the training and testing are run 10 times and the final result is the average of the results of the runs. For each run, 80% of the randomly drawn records from the dataset are used for model training and the 20% remaining are used for model testing.

The model testing metrics used include PPV (accuracy), TPR (positive true rate, or sensitivity), FPR (positive error rate), FNR (negative error rate), F1 (measure F1) and ACC (overall accuracy). These metrics are calculated using the following formulas:

$$PPV = \frac{TP}{TP + FP} \quad (4)$$

$$TPR \frac{TP}{TP + FN} \quad (5)$$

$$FPR = \frac{FP}{FP + TN} \quad (6)$$

$$FNR = \frac{FN}{FN + TP} \quad (7)$$

$$F1 = \frac{2TP}{2TP + FP + FN} \quad (8)$$

$$ACC = \frac{TP + TN}{TP + FP + FN + TN} \quad (9)$$

where, the parameters TP, TN, FP and FN are the components of the confusion matrix given in Table III.

TABLE III. THE CONFUSION MATRIX

		Actual Class	
		Attacked	***Normal***
Predicted Class	***Attacked***	TP	FP
	Normal	FN	TN

The testing results of the web attack detection model with different machine learning algorithms using the 10-fold cross-validation method are shown in Table IV.

D. Discussion

The experimental results in Table IV show that the random forest algorithm gives the best detection performance, while the Naïve Bayes algorithm gives the worst performance. The detection performance of the models based on the decision tree algorithms and the SVM is at the same level. The F1 measure of the detection models based on the machine learning algorithms of random forest, decision tree, Rbf-SVM, Linear-SVM, and Naive Bayes is 99.57%, 98.76%, 98.33, 98.10%, and 75.10%, respectively. Although the random forest algorithm requires higher computational resources than the decision tree algorithm, the random forest-based model has significantly higher detection performance than the decision tree-based model, especially in reducing false alarm rates. Specifically, the measures of FPR and FNR of the random forest-based model are 0.0775% and 0.7253%, respectively, while these measures of the decision tree-based model are 0.5185% and 1.6122%, respectively.

TABLE IV. THE PERFORMANCE OF THE TESTED MODEL BASED SOME MACHINE LEARNING ALGORITHMS USING THE 10-FOLD CROSS-VALIDATION

Used Algorithms	**PPV (%)**	**TPR (%)**	**FPR (%)**	**FNR (%)**	**ACC (%)**	**F1 (%)**
Naive Bayes	62,42	94,26	33,74	5,7402	76,70	75,10
Linear-SVM	97,64	98,56	1,4508	1,4444	98,55	98,10
Rbf-SVM	98,68	97,97	0,8065	2,0253	98,73	98,33
Decision Tree	99,14	98,39	0,5185	1,6122	99,07	98,76
Random Forest	99,87	99,27	0,0775	0,7253	99,68	99,57

TABLE V. COMPARISON OF OVERALL DETECTION ACCURACY (ACC) OF THE TESTED MODEL WITH THE SURVEYED MODELS

Tested model	**Hoang**	**Liang et al.**	**Pan et al.**
99,68%	98,56%	98,42%	91,40%

Table V provides the results of comparing the overall detection accuracy (ACC) of the tested model with the surveyed models. It can be seen that the tested web attack detection model gives superior overall detection accuracy compared to the existing detection models. Specifically, the overall detection accuracy of the tested model, Hoang [3], the deep learning-based models of Liang et al. [16] and Pan et al. [17] is 99.68%, 98.56%, 98.42% and 91.40%, respectively.

V. CONCLUSION

This paper has surveyed tools, solutions and techniques for monitoring and detecting web attacks. The survey and evaluation results are the

basis for the selection of tools, solutions, or techniques to monitor and detect web attacks in accordance with the specific web application systems in practice. The results of testing and evaluating detection models built on different machine learning algorithms show that the random forest machine learning algorithm gives the highest detection accuracy and the lowest false alarm rate. Moreover, the tested model in the paper also achieved superior detection performance compared to the investigated machine learning-based detection models.

Future expansion directions of the paper include (1) expanding the model that allows to detect more types of web attacks and (2) developing a web attack detection system based on the web attack detection model using random forest algorithm for practical deployment.

REFERENCES

[1] OWASP, Open Web Application Security Project, http://www.owasp.org, accessed 1.2021.

[2] Hoàng Xuân Dậu, An toàn ứng dụng web và cơ sở dữ liệu, Học viện Công nghệ Bưu Chính Viễn Thông, 2017.

[3] Hoang, X.D. Detecting Common Web Attacks Based on Machine Learning Using Weblog. K.-U. Sattler et al. (Eds.): ICERA 2020, LNNS 178, pp. 311–318, 2021.

[4] VNCS – Giải pháp giám sát website tập trung, http://vncs.vn/portfolio/giai-phap-giam-sat-websites-tap-trung, accessed 1.2021.

[5] Nagios Web Application Monitoring Software, https://www.nagios.com/solutions/web-application-monitoring/, accessed 1.2021.

[6] Site24x7, Website Defacement Monitoring, https://www.site24x7.com/monitor-webpagedefacement.html, accessed 1.2021.

[7] Mod Security, https://www.modsecurity.org, accessed 1.2021.

[8] Snort IDS, http://www.snort.org, accessed 1.2021.

[9] Abhishek Kumar Baranwal, Approaches to detect SQL injection and XSS in web applications, EECE 571B, Term Survey Paper, University of British Columbia, Canada, 2012.

[10] OWASP ModSecurity Core Rule Set, https://www.owasp.org/index.php/Category:OWASP_ModSecurity_Core_Rule_Set_Project, accessed 1.2021.

[11] Kemalis, K. and T. Tzouramanis. SQL-IDS: A Specification-based Approach for SQLinjection Detection. SAC'08. Fortaleza, Ceará, Brazil, ACM (2008), pp. 2153-2158.

[12] P. Bisht, and V.N. Venkatakrishnan, "XSS-GUARD: Precise dynamic prevention of Cross-Site Scripting Attacks," In Proceeding of 5th Conference on Detection of Intrusions and Malware & Vulnerability Assessment, LNCS 5137, 2008, pp. 23-43.

[13] Doyen Sahoo, Chenghao Liu, and Steven C.H. Hoi, Malicious URL Detection using Machine Learning: A Survey, https://arxiv.org/abs/1701.07179, Mar 2017.

[14] Gustavo Betarte, Eduardo Giménez, Rodrigo Martínez, and Álvaro Pardo, Machine learning-assisted virtual patching of web applications, https://arxiv.org/abs/1803.05529, Mar 2018.

[15] Carmen Torrano-Gimenez, Alejandro Pérez-Villegas and Gonzalo Alvarez, An Anomaly-Based Approach for Intrusion Detection in Web Traffic, published by The Allen Institute for Artificial Intelligence, 2009.

[16] Jingxi Liang, Wen Zhao and Wei Ye. "Anomaly-Based Web Attack Detection: A Deep Learning Approach". ICNCC 2017, Kunming, China, December 8-10, 2017.

[17] Yao Pan, Fangzhou Sun, Zhongwei Teng, Jules White, Douglas C. Schmidt, Jacob Staples and Lee Krause. "Detecting web attacks with end-to-end deep learning". Journal of Internet Services and Applications (2019) 10:16, SpringerOpen.

[18] HTTP DATASET CSIC 2010, https://www.isi.csic.es/dataset/, accessed 1.2021.

[19] HTTP Param Dataset, https://github.com/Morzeux/ HttpParamsDataset, accessed 1.2021.

[20] A. Smola and S.V.N. Vishwanathan, "Introduction to Machine Learning," Cambridge University, 2008.

ABOUT THE AUTHORS

Hoang Xuan Dau

Workplace: Posts and Telecommunications Institute of Technology

Email: dauhx@ptit.edu.vn

Education: Received bachelor degree in 1994, master degree in 2000 and PhD of computer science in 2006.

Recent research direction: attack and intrusion detection, malware detection, system and software security, web security, machine learning-based applications for information security.

Ninh Thi Thu Trang

Workplace: Posts and Telecommunications Institute of Technology

Email: trangninh.nt3@gmail.com

Education: Received bachelor degree in 2016 and master of information systems in 2018.

Recent research direction: Web security, network monitoring and security.

Nguyen Trong Hung

Workplace: Academy of People's Security

Email: tronghungt31@gmail.com

Education: Received bachelor degree in 2013 and master of information security in 2018.

Recent research direction: attack and intrusion detection, malware detection, and web security.