Detection and Prevention of Cross-site Scripting Attack with Combined Approaches

Hsing-Chung Chen[1,2,*], (Senior Member, IEEE), Aristophane Nshimiyimana[1], Cahya Damarjati[1,3], Pi-Hsien Chang[1,4,*],
[1] Department of Computer Science & Information Engineering, Asia University No.500, Liufeng Road, Wufeng District, Taichung City, Taiwan
[2] Dept. of Medical Research, China Medical University Hospital, China Medical University, Taiwan
[3] Dept. of Information Technology, Universitas Muhammadiyah Yogyakarta, Yogyakarta, Indonesia
[4] Information Management Center, Taichung City Government
[*]Corresponding authors: Hsing-Chung Chen (e-mail: cdma2000@asia.edu.tw, shin8409@ms6.hinet.net), Pi-Hsien Chang (e-mail: ps491@taichung.gov.tw)

2021 International Conference on Electronics, Information, and Communication (ICEIC) | 978-1-7281-9161-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICEIC51217.2021.9369796

***Abstract*— Cross-site scripting (XSS) attack is a kind of code injection that allows an attacker to inject malicious scripts code into a trusted web application. When a user tries to request the injected web page, he is not aware that the malicious script code might be affecting his computer. Nowadays, attackers are targeting the web applications that holding a sensitive data (e.g., bank transaction, e-mails, healthcare, and e-banking) to steal users' information and gain full access to the data which make the web applications to be more vulnerable. In this research, we applied three approaches to find a solution to this most challenging attacks issues. In the first approach, we implemented Random Forest (RF), Logistic Regression (LR), k-Nearest Neighbors (k-NN), and Support Vector Machine (SVM) algorithms to discover and classify XSS attack. In the second approach, we implemented the Content Security Policy (CSP) approach to detect XSS attacks in real-time. In the last approach, we propose a new approach that combines the Web Application Firewall (WAF), Intrusion Detection System (IDS), and Intrusion Prevention System (IPS) to detect and prevent XSS attack in real-time. Our experiment results demonstrated the high performance of AI algorithms. The CSP approach shows the results for the detection system report in real-time. In the third approach, we got more expected system results that make our third model system a more powerful tool to address this research problem than the other two approaches.**

Keywords—Cross-site Scripting Attack, Web Application Firewall; Intrusion Detection System; Intrusion Prevention System; machine learning

I. INTRODUCTION

Nowadays, as the number of web application users increases, it makes them more vulnerable to hackers because they exchange sensitive data more frequently. Therefore, for this reason, the improvements in security and privacy of the web applications are needed because the hacker targets vulnerable web applications by injecting malicious scripts code into a trusted web application or website to steal or destroy target information. Some researchers have been already done researches to address this issue using different approaches but until nowadays we still facing the same kind of issues. We learned from the previous researches and started from their knowledge to understand the issue to build new methods or approaches to solving this kind of issue.

Cross-site scripting (XSS) attack is a kind of code injection and lets attackers inject malicious scripts code into a trusted web application. An XSS attack happens when attackers are effectively injected malicious script code and send it to various web users in the form of a browser-side script. An attacker could increase restricted admittance to access delicate information such as web content, cookies, and so on through the web end users. This method is called cross-site scripting (XSS) and could contaminate a Web site or any user's browser accessing the information on it. It has become one of the leading threats to today's applications [1, 15-19]. This type of attack happens when a web-based application is infected with a malicious code such as JavaScript that could be requested from a user's browser [2, 15, 18, 19]. This could be complicated in the process of detecting and preventing XSS attacks because of the different way browsers interpret the code [3]. Thus, an XSS attack could be a serious threat to any organization doing business on the Web. It has become one of the biggest problems for developers of web applications. Fixing these vulnerabilities in a large system could be challenging and may not be accomplished easily. To address this attack, new methods are needed to deal with a huge amount of data from different web users. There are three major types of Cross-site scripting attacks namely [1]:

- Reflected XSS is where the injected malicious script code comes from the current HTTP request,
- Stored XSS is where the injected malicious script code comes from the web site's database,
- DOM-based XSS is known as client-side code, where the vulnerability is injected on client-side code rather than server-side code.

Our research combined different algorithms to find the best solution to detect and prevent the Cross-Site Scripting attack in real-time. To achieve our research target, we implemented RF, LR, k-NN, and SVM (linear and poly kernels) as artificial intelligence algorithms to discover, classify and identify XSS malicious and benign script code in the dataset. Moreover, we deployed the CSP approach to detect and prevent XSS attack through a web browser by embedding CSP tools, and we implemented a third approach namely Web Application Intrusion Detection And Prevention System Firewall (WAIDPFS) that combine IDS, IPS [4, 13] and WAF into our system model as a shield of web server containing web application (s) to detect and defend by preventing XSS attack as shown in section 2 and 3.

II. System Design

The first contribution and aim of this research are to discovering Cross-Site Scripting attacks using artificial intelligence to achieve our target, we used the XSS dataset shown in Figure 8 to 9 containing benign code and malicious script code. Our research proposed four algorithms: RF, LR, k-NN, and SVM (linear and poly kernels) for classification to achieve the research target. The second contribution of this research is to deploy our new approach namely WAIDPFS to detect and prevent XSS attack in real-time. Lastly, we implemented a Content Security Policy approach to detect XSS attack by receiving the report and improve HTML code by adding CSP HTTP headers to prevent the next attacks.

A. Datasets

The Cross-Site Script dataset used in our research study was obtained from the GitHub website [5] and contains 24096 rows and 66 columns of scripts, with 14096 labeled as benign and 10000 labeled as malicious. We splatted our dataset into two sets, the first set is the training dataset with 80% of our original dataset to extract features and fit the model, the second set is the testing dataset as new incoming entry data to test our model with 20% of the dataset.

B. Classifiers

In this research, we used four algorithms to build a model for classification to discovering malicious script code and calculate their performance from secondary datasets [5] to prevent attacks in the future as shown in Fig. 1. RF, SVM[6], k-NN [7], and LR [8, 9] algorithms are used [10]. SVM is used with two variations (linear kernel and polynomial kernel). k-NN was adjusted by setting parameter *k* and RF was adjusted by setting the number of trees.

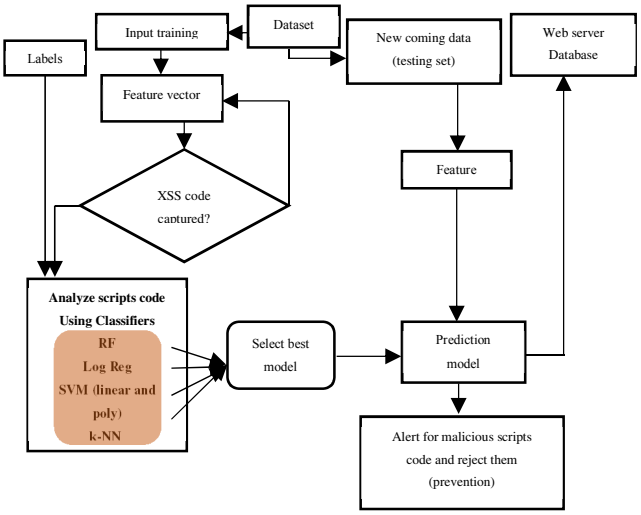

Fig. 1. Discovering the XSS attack system

C. Implementing the CSP algorithm

CSP is suggested by W3C on September 13, 2016. CSP is the best tool that web developers could utilize to restrict their applications in many ways to mitigating the harmful of code injection vulnerabilities such as XSS attack and minimizing the privilege with where their applications will be run [11]. CSP is proposed as the first tool of defense against code injection vulnerabilities. A web developer using CSP to build a trusted web application with Safelist allowed resources and their origins. In other words, scripts codes that are wanted to execute the web application are allowed from specific origins path. Otherwise, CSP blocks them as shown in Fig. 2.

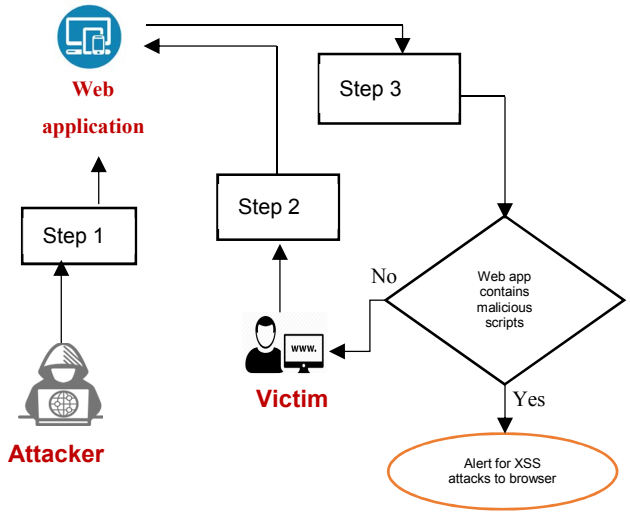

Fig. 2. CSP detection and prevention structure system

CSP detection and prevention structure system explanations:

Step 1: The attacker injects malicious scripts code into web applications to exploit client-side,

Step 2: User request web page to the webserver that attacker already injected scripts code,

Step 3: When the server response to the users, the page loaded on the user browser and show the report on the browser,

Step 4: if CSP detects malicious code Alert for XSS attacks to the browser.

CSP could report policy violations to the server to track instances of misapplication or other attacks using Content-Security-Policy-Report-Only "Content-Security-Policy-Report-Only: policy" as an HTTP header to specify our policy. The violation reporting mechanism in this document has been designed to reduce the risk that a malicious website could use violation reports to investigate the performance of other servers and set to the report-URI directive. Consider the malicious web site, https://onlineapplication.com/login.html, allowing as a source of online application, which uses the following policies by disallowing everything except stylesheet from https://cdn.onlineapplication.com with CSP HTTP header "Content-Security-Policy: default-src 'none'; style-src cdn.onlineapplication.com; report-URI /_/csp-reports". If the stylesheet is allowed to be executed only from https://cdn.onlineapplication.com, the website would attempt to connect the web link from its origin URL (http://onlineapplication.com). A web browser is capable of enforcing CSP to send the following violations report as a POST request to http://onlineaplication.com/_/csp-reports, when a web page is visited as shown in Fig. 3. If the malicious site attempts to load https://onlineapplication.com/login.html as an online application and the web server redirects to an untrusted URL, CSP will block the HTTP request.

```
{
  "csp-report":
  {
     "document-uri": "http://onlineapplication.com/login.html","referrer": "",
     "blocked-uri": "http://onlineaplication.com/css/style1.css",
     "violated-directive": "style-src cdn.onlineapplication.com",
     "original-policy": "default-src 'none'; style-src cdn.onlineaplication.com;
        report-uri /_/csp-reports"
   }
}
```

Fig. 3. CSP violation report

If violation reports contained the full blocked URL, the violation report might contain sensitive information that is in the redirected URL, such as session identifiers, IP address as identities. For this objective, the web browser includes only the URL of the original request, not the redirect target. Referrer to the following CSP algorithm for XSS attack as follow:

```
Request violate policy report algorithm
Begin
    Input:
      initialize request
      initialize Policy
     output
       Let violates = does not violate
       for Directive in Policy:
            If Results== blocked
                 Let violates =Directive
       Return violates
End
```

D. *Implement IDS/IPS/WA*

we implemented a new approach namely Web Application Intrusion Detection Prevention Firewall System [4] combined with AI algorithms to detect and prevent various attacks including the XSS attack as our research target to find the best solution to this issue. This combination of IDS/IPS/WAF [12] with the integration of AI algorithms, is the best solution for this issue as shown in Fig. 4, it will take or require more knowledge and time for the attackers to reach the webserver because every system block needs to analyze HTTP request traffics, if detected an attack directly send an alert to the system management. If an attacker tries to bypass any of one system block, will need to be checked again to another system bock, which means that the attacker will require another knowledge or new technique also to bypass. Our research contribution is to provide a new technique or approach to fighting against this kind of challenge, system detects HTTP request traffic containing any kind of malicious script code injection and prevents them by blocking or denying the request before reaching the webserver. As the result shown that the Web Application Intrusion Detection Prevention Firewall System (WAIDPFS) is the best technique to use in real-time to solve this kind of attack's problem.

The following steps were used to generate the final response to the user if the HTTP request was accepted or denied as shown in Fig. 4,

Step 1: User or Attacker requests a web page from a web server by sending an HTTP request via the internet,

Step 2: Before reach to the server, IDS analyze where the HTTP request comes from and if the requested web page doesn't contain any malicious scripts code with AI algorithms to classify them. If IDS detect some attacks such as injection of malicious scripts code, automatically block the HTTP request and reject it and send an alert of intrusion attack to system management, else allow the HTTP request to go through. And if an attacker used bypass techniques to avoid being detected by IDS and pass through it. IPS will be waiting to check the HTTP request again,

Step 3: IPS does the same process like IDS by sending an alert of intrusion to defend the webserver system. And if an attacker used bypass techniques to avoid being detected by IPS and pass through it. WAF will be waiting to verify the HTTP request again.

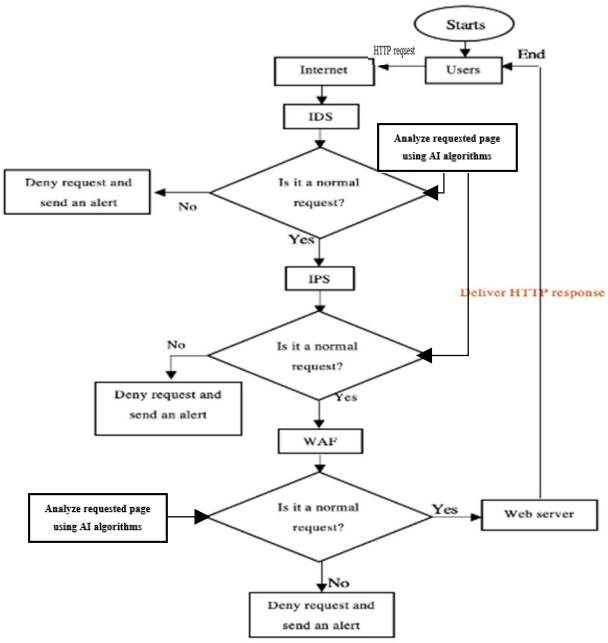

Fig. 4. WAIDPFS system flow

Step 4: WAF analyzes again the HTTP request with AI algorithms to classify them to find if there's no trace of injection of malicious script code and allow HTTP request to go to the webserver, else block the HTTP request by protecting web server and send an alert of intrusion.

Step 5: Once an HTTP request (with POST or GET) reaches at web server means the request is secure and the webserver response user HTTP request by sending back an HTTP response as shown in Fig. 4.

III. EXPERIMENT RESULTS AND DISCUSSIONS

In this section, we show the results of our experiments with the proposed approaches.

A. *Testing the AI algorithm*

Our research experiment was focused on the performance of RF, LR, k-NN, and SVM (linear and poly kernels) [14] algorithms using the XSS dataset and its features to build and

select the best model to discover XSS attack in the XSS dataset to prevent attacks in the real-time. We used the cross-validation (cv) technique to avoid overfitting and split data into five folds to calculate the performance of each algorithm to find the best algorithm(s) to accomplish the research target. Our model has been trained via four folds and tested them to the last fold. k-NN was adjusted by setting parameter *k* to 15, RF was adjusted by setting the number of trees to 100. To evaluate our model were constructed by classifiers using the entire training set. Then the testing set that holds new malicious script code and benign scripts was used for testing the classifiers' performance.

TABLE I. COMPARISON BETWEEN FOUR ALGORITHMS' PERFORMANCE

Algorithms / Compared Items	***RF***	***SVM(L)***	***SVM(P)***	***LR***	***k-NN***
Accuracy	99.91%	99.85%	99.62%	99.87%	99.91%
Precision	99.80%	99.75%	99.90%	99.80%	99.85%
Recall	100%	99.90%	99.20	99.90%	99.95%

The five folds calculation shown high performance for the classifiers, with prospective results for SVM (linear) provided better performance results than SVM (polynomial), k-NN, RF, and LR. As shown in Table 1, the classifiers model discriminated malicious script with benign scripts with high accuracy, high precision, and high recall, where RF performed slightly better compare SVM (linear, polynomial), k-NN, and LR, with high accuracy of 99.91%, precision 99.80%, and recall 100% with an evaluation of testing data. The experiment results show that classifiers built the model that could be worthy of discovering a Cross-Site Scripting attack in real-time.

B. Testing the CSP algorithm

In this research experiment, we tested CSP tools embedded into a web browser and then loaded DVWA as an open-source website from our server for injecting malicious scripts code to detect XSS attack and prevent them by adding CSP HTTP headers into HTML code. However, we deployed also CSP into WordPress with HTTP headers plugin to prevent attacks by sending an alert report automatically through email.

C. Testing IDS/IPS/WAF

In this research experiment, we tested the IDS/IPS/WAF system [4] combined with AI algorithms to detect and prevent various attacks including the XSS attack as our research target to find the best solution to this issue of attacks. We tested our approach by checking our firewall system capability using wafw00f to test if our web application is behind IDS/IPS/WAF. Moreover, we injected malicious script code to check the detection and prevention capability. The results are demonstrated that our web application is under protection by IDS/IPS/WAF system because the system detects and prevents attack automatically without any human intervention which is make it to be an artificial firewall system.

IV. CONCLUSION

This research experiment's results concluded that RF performed slightly better compare SVM (linear, polynomial), k-NN, and LR. The experiment results of this research have been demonstrated the best outcome which could help to figure out how to predict using artificial intelligence algorithms. With our new proposed approach WAIDPFS combined with AI algorithms for detecting and preventing Cross-Site Scripting demonstrated with high performance in real-time detection and prevention by defending automatically web server as shown in Fig. 4 while we injected malicious scripting code. With CSP approach, we got currently reports and also we deployed HTTP headers plugin into our HTML code to prevent next attack and plugin HTTP headers also on WordPress server to help IDS/IPS/WAF combined with AI algorithms to detect attackers easily and prevent them by blocking their HTTP request or IP address in the future XSS attack in the real-world to reinforcing the security and privacy of web applications.

ACKNOWLEDGMENT

This work was sponsored by the Ministry of Science and Technology (MOST), Taiwan, under Grant No. MOST 107-2221-E-468 -015. This work was also supported in part by Asia University, Taiwan, and China Medical University Hospital, China Medical University, Taiwan, under Grant ASIA-108-CMUH-05 and ASIA-107-CMUH-05. This work was also supported in part by Asia University, Taiwan, UMY, Indonesian, under Grant 107-ASIA-UMY-02.

REFERENCES

[1] K. Selvamani, A. Duraisamy, and A. Kannan, "Protection of Web Applications from Cross-Site Scripting Attacks in Browser Side," vol. 7, no. 3, pp. 229–236, 2010, [Online]. Available: http://arxiv.org/abs/1004.1769.

[2] T. Venkat, N. Rao, V. Tejaswini, and K. Preethi, "Defending Against Web Vulnerabilities and Cross-Site Scripting," J. Glob. Res. Comput. Sci., vol. 3, no. 5, pp. 61–64, 2012.

[3] S. Saha, "Consideration Points Detecting Cross-Site Scripting," vol. 4, no. 1, 2009, [Online]. Available: http://arxiv.org/abs/0908.4188.

[4] Y. C. Kao, J. C. Liu, Y. H. Wang, Y. H. Chu, S. C. Tsai, and Y. B. Lin, "Automatic blocking mechanism for information security with SDN," J. Internet Serv. Inf. Secur., vol. 9, no. 1, pp. 60–73, 2019, doi: 10.22667/JISIS.2019.02.28.060.

[5] F. A. M. and J. M. Howe, "Detecting Cross-Site Scripting Attacks Using Machine Learning," vol. 723, no. January. pp. 229–239, 200–210, 2018, doi: 10.1007/978-3-319-74690-6.

[6] T. M. Mitchell, Machine learning. McGraw-Hill, 1997.

[7] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," Am. Stat., vol. 46, no. 3, pp. 175–185, 1992, doi: 10.1080/00031305.1992.10475879.

[8] L. J. Helsloot, G. Tillem, and Z. Erkin, "Badass: Preserving privacy in behavioural advertising with applied secret sharing," J. Wirel. Mob. Networks, Ubiquitous Comput. Dependable Appl., vol. 10, no. 1, pp. 23–41, 2019, doi: 10.22667/JOWUA.2019.03.31.023.

[9] J. Tolles and W. J. Meurer, "Logistic regression: Relating patient characteristics to outcomes," JAMA - J. Am. Med. Assoc., vol. 316, no. 5, pp. 533–534, 2016, doi: 10.1001/jama.2016.7653.

[10] A. Kim, J. Oh, J. Ryu, J. Lee, K. Kwon, and K. Lee, "SoK: A systematic review of insider threat detection," J. Wirel. Mob. Networks, Ubiquitous Comput. Dependable Appl., vol. 10, no. 4, pp. 46–67, 2019, doi: 10.22667/JOWUA.2019.12.31.046.

[11] M. Johns, "Script-templates for the content security policy," J. Inf. Secur. Appl., vol. 19, no. 3, pp. 209–223, 2014, doi: 10.1016/j.jisa.2014.03.007.

[12] K. Nagendran, S. Balaji, B. A. Raj, P. Chanthrika, and R. G. Amirthaa, "Web Application Firewall Evasion Techniques," 2020 6th Int. Conf. Adv. Comput. Commun. Syst. ICACCS 2020, pp. 194–199, 2020, doi: 10.1109/ICACCS48705.2020.9074217.

[13] C. Day, "Intrusion prevention and detection systems," Manag. Inf. Secur. Second Ed., pp. 119–142, 2013, doi: 10.1016/B978-0-12-416688-2.00005-2.

[14] S. Han, J. Jung, H. Kim, S. J. Cho, and K. Suh, "Efficient android malware detection using API rank and machine learning," J. Internet Serv. Inf. Secur., vol. 9, no. 1, pp. 48–59, 2019, doi: 10.22667/JISIS.2019.02.28.048.

[15] H.-C. Chen, S.-S. Kuo, "Active Detecting DDoS Attack Approach Based on Entropy Measurement for the Next Generation Instant Messaging App on Smartphones," Intelligent Automation and Soft Computing (Autosoft Journal), Vol. 25, No. 1, pp. 217-228, 2019.

[16] H.-C. Chen., I. You, C.-E. Weng, C.-H. Cheng, and Y.-F. Huang, "A security gateway application for End-to-End M2M communications," Computer Standards

& Interfaces, Vol. 44, pp. 85-93, Feb. 2016.
[17] H.-C. Chen, C.-H. Mao, S.-S. Tseng, "An Approach for Detecting a Flooding Attack Based on Entropy Measurement of Multiple E-Mail Protocols," Journal of Applied Science and Engineering, Vol. 18, No. 1, pp. 79-88, 2015.
[18] H.-C. Chen., Z.-Y. Chen, S.-S. Tseng, K.-M. Chang, F. Chang and M.-H. Jiang, "The Hot Security Topics Analysis for the Announcements in ICANN Website by Using Web Crawler and Association Rule Technology," Part of the Advances in Intelligent Systems and Computing book series (AISC, volume 994), The 13th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS-2019), First Online: 19 June 2019.
[19] H.-C. Chen. And S.-S. Kuo, "DoS Attack Pattern Mining Based on Association Rule Approach for Web Server," The 12th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS-2018), July 4th – July 6th , 2018, Kunibiki Messe, Matsue, Japan.