Machine Learning Algorithm for Web Vulnarabilities Detection

Murugesan Senthil Kumar, Saresh Kumar Ellamla

Department of Electronics and Communication Engineering

Sree Dattha Group of Institutions, Hyderabad, Telangana, India.

ABSTRACT

Web applications are the most common interface to security sensitive data and functionality available nowadays. They are routinely used to file tax incomes, access the results of medical screenings, perform financial transactions, and share opinions with our circle of friends, just to mention a few popular use cases. On the downside, this means that web applications are appealing targets to malicious users (attackers) who are determined to force economic losses, unduly access confidential data or create embarrassment to their victims. Securing web applications is well known to be hard.

There are several reasons for this, ranging from the heterogeneity and complexity of the web platform to the adoption of undisciplined scripting languages offering dubious security guarantees and not amenable for static analysis. In such a setting, black-box vulnerability detection methods are particularly popular. As opposed to white-box techniques which require access to the web application source code, black-box methods operate at the level of HTTP traffic, i.e., HTTP requests and responses. Though this limited perspective might miss important insights, it has the key advantage of offering a language-agnostic vulnerability detection approach, which abstracts from the complexity of scripting languages and offers a uniform interface to the widest possible range of web applications. This sounds appealing, yet previous work showed that such an analysis is far from trivial. One of the main challenges there is how to expose to automated tools a critical ingredient of effective vulnerability detection, i.e., an understanding of the web application semantics.

Keywords: Web vulnerabilities, web applications, machine learning.

1. INTRODUCTION

As technology evolves, cyber-criminals are also improving their attack methods, tools, and techniques to exploit organizations. Public web-services are common services that anyone can access, and many companies provide their services through open webpages. If a web-service fails or is compromised, it can cause a drop in corporate reputation or revenue. In general, security managers prevent intrusions from external attacks by registering all denied black-list policies for unused services in the firewall, but web-services in the Internet Demilitarized Zone (DMZ) cannot be blocked by firewalls because they are always open to public access. As such, identifying normal access and differentiating it from malicious attacks is an important task in cybersecurity. Many security incidents originated with web-attacks such as information disclosure, service failures, and malware infections.

Hypertext transfer protocol (HTTP) [1] is an application-level protocol for distributed, collaborative, and hypertext information systems. Today's HTTP is evolving where the information is transferred from web pages, and it is also used for exchanges or sending system commands to various devices, such as command-lines, update scripts, and mobile apps. Web-attacks often exploit vulnerabilities in applications in open web services rather than perform a host-level system penetration [2]. The attacker attempts to attack by sending exploitational code using a vulnerability in a specific domain or path file of the webserver. The webserver or device that is injected with the code can subsequently be compromised by the attacker [3].

2. LITERATURE SURVEY

Khalid et. al [4] proposed a method namely NMPREDICTOR to predict vulnerable files in website for vulnerability prediction as a classification problem by predicting legitimate or vulnerable code. In addition, it is an effort to employ the classification on different classifier of machine learning algorithms to judge elimination of vulnerable components. Numerous experiments have been conducted in our study to evaluate the performance of our proposed model. Through our proposed method, they have builds 6 classifiers on a training set of labeled files represented by their software metrics and text features. Additionally, they builds a Meta classifier, which combines the six underlying classifiers i.e. J48, Naive Bayes and Random forest. NMPREDICTOR is evaluated on datasets of three web applications, which offers 223 superior quality vulnerabilities found in PHPMyAdmin, Moodle and Drupal. Our proposed method shows a clearly has an advantage over results of existing studies in case of Drupal, PhpMyAdmin and Moodle.

Pan et. al [5] provides three contributions to the study of autonomic intrusion detection systems. First, they evaluate the feasibility of an unsupervised/semi-supervised approach for web attack detection based on the Robust Software Modeling Tool (RSMT), which autonomically monitors and characterizes the runtime behavior of web applications. Second, they describe how RSMT trains a stacked denoising autoencoder to encode and reconstruct the call graph for end-to-end deep learning, where a low-dimensional representation of the raw features with unlabeled request data is used to recognize anomalies by computing the reconstruction error of the request data. Third, they analyze the results of empirically testing RSMT on both synthetic datasets and production applications with intentional vulnerabilities. Our results show that the proposed approach can efficiently and accurately detect attacks, including SQL injection, cross-site scripting, and deserialization, with minimal domain knowledge and little labeled training data.

Kumar et. al [6] present EDIMA, a distributed modular solution which can be used towards the detection of IoT malware network activity in large-scale networks (e.g., ISP, enterprise networks) during the scanning/infecting phase rather than during an attack. EDIMA employs machine learning algorithms for edge devices' traffic classification, a packet traffic feature vector database, a policy module and an optional packet sub-sampling module. They evaluate the classification performance of EDIMA through testbed experiments and present the results obtained.

Vigneswaran et. al [7] utilized DNNs to predict the attacks on Network Intrusion Detection System (N-IDS). A DNN with 0.1 rate of learning is applied and is run for 1000 number of epochs and KDDCup-`99' dataset has been used for training and benchmarking the network. For comparison purposes, the training is done on the same dataset with several other classical machine learning algorithms and DNN of layers ranging from 1 to 5. The results were compared and concluded that a DNN of 3 layers has superior performance over all the other classical machine learning algorithms.

Mokbal et. al [8] proposed a robust artificial neural network-based multilayer perceptron (MLP) scheme integrated with the dynamic feature extractor for XSS attack detection. The detection scheme adopts a large real-world dataset, the dynamic features extraction mechanism, and MLP model, which successfully surpassed several tests on an employed unique dataset under careful experimentation, and achieved promising and state-of-the-art results with accuracy, detection probabilities, false positive rate, and AUC-ROC scores of 99.32%, 98.35 %, 0.3%, and 99.02%, respectively. Therefore, it has the potentials to be applied for XSS-based attack detection in either the client-side or the server-side.

Patil et. al [9] discusses three approaches for detecting phishing websites. First is by analyzing various features of URL, second is by checking legitimacy of website by knowing where the website is being hosted and who are managing it, the third approach uses visual appearance-based analysis for checking genuineness of website. They make use of Machine Learning techniques and algorithms for evaluation of these different features of URL and websites. In this paper, an overview about these approaches is presented.

Şahin et. al [10] proposes a novel software vulnerability prediction model based on using a deep learning method and SYMbiotic Genetic algorithm. They are first to apply Diploid Genetic algorithms with deep learning networks on software vulnerability prediction to the best of our knowledge. In this proposed method, a deep SYMbiotic-based genetic algorithm model (DNN-SYMbiotic GAs) is used by learning the phenotyping of dominant-features for software vulnerability prediction problems. The proposed method aimed at increasing the detection abilities of vulnerability patterns with vulnerable components in the software. Comprehensive experiments are conducted on several benchmark datasets; these datasets are taken from Drupal, Moodle, and PHPMyAdmin projects. The obtained results revealed that the proposed method (DNN-SYMbiotic GAs) enhanced vulnerability prediction, which reflects improving software quality prediction.

3. PROPOSED SYSTEM

Cross-Site Request Forgery (CSRF) is a well-known web attack that forces a user into submitting unwanted, attacker-controlled HTTP requests towards a vulnerable web application in which she is currently authenticated. The key concept of CSRF is that the malicious requests are routed to the web application through the user's browser, hence they might be indistinguishable from intended benign requests which were authorized by the user. The CSRF does not require the attacker to intercept or modify user's requests and responses: it suffices that the victim visits the attacker's website, from which the attack is launched. Thus, CSRF vulnerabilities are exploitable by any malicious website on the Web.

3.1 Advantages of Proposed System:

- The value of standard HTTP request headers such as Referrer and Origin, indicating the page originating the request.
- The presence of custom HTTP request headers like X-Requested-With, which cannot be set from a cross-site position.
- The presence of unpredictable anti-CSRF tokens, set by the server into sensitive forms.
- Algorithm: RandomForestClassifier

4. RESULTS AND DISCUSSION

Modules

- User
- Admin
- False Positives and False Negatives
- Machine Learning Classifier

Modules Description

User: The User can register the first. While registering he required a valid user email and mobile for further communications. Once the user register then admin can activate the customer. Once admin

activated the customer then user can login into our system. User can do the data preprocess. First required running website name. By using that website, the user can test the csrfs. By help of bolt tool the user can fetch related all csrfs and generated algorithm names. The result will be stored in json files. Later the user can get the results of Mitch dataset. The mitch dataset tested for POST method as well GET method to. The result will be displayed on the browser.

Admin: Admin can login with his credentials. Once he login he can activate the users. The activated user only login in our applications. The admin can set the training and testing data for the project of the Mitch Dataset. The user searches all urls related csrf token admin can view in his page. The admin can also check the POST method performed data from the dataset and GET method related data also.

False Positives and False Negatives: Mitch produces a false positive when it returns a candidate CSRF that cannot be actually exploited. This is something relatively easy to detect by manual testing, though this process is tedious and time-consuming. In general, it is not possible to reliably identify when Mitch produces a false negative, because this would require to know all the CSRF vulnerabilities on the tested websites. To estimate this important aspect, we keep track of all the sensitive requests returned by the ML classifier embedded into Mitch and we focus our manual testing on those cases. This is a reasonable choice to make the analysis tractable, because we first showed that the classifier performs well using standard validity measures.

Machine Learning Classifier: The ML classifier used by Mitch was trained from a dataset of around 6000 HTTP requests from existing websites, collected and labeled by two human experts. The feature space X of the classifier has 49 dimensions, each one capturing a specific property of HTTP requests. Those can be organized into following categories.

following set of numerical features:

numOfParams: the total number of parameters;

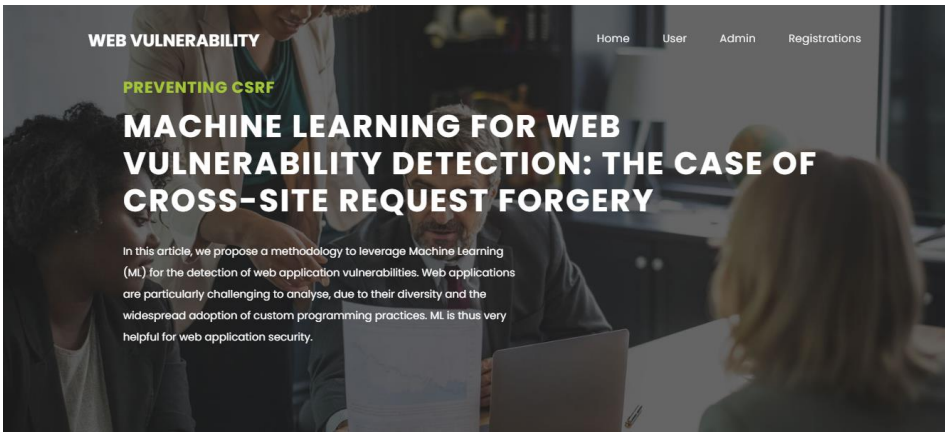
numOfBools: the number of request parameters bound to a boolean value;

numOfIds: the number of request parameters bound to an identifier, i.e., a hexadecimal string, whose usage was empirically observed to be common in our dataset;

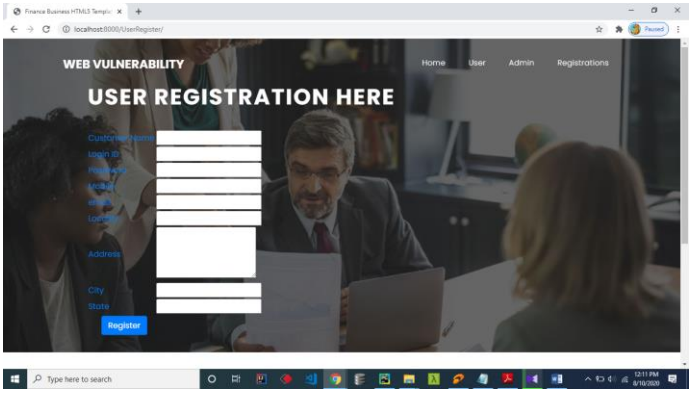
numOfBlobs: the number of request parameters bound to a blob, i.e., any string which is not an identifier;

reqLen: the total number of characters in the request, including parameter names and values.

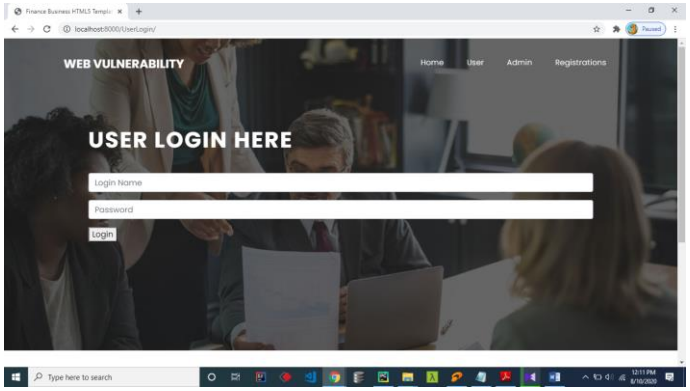
Home page:



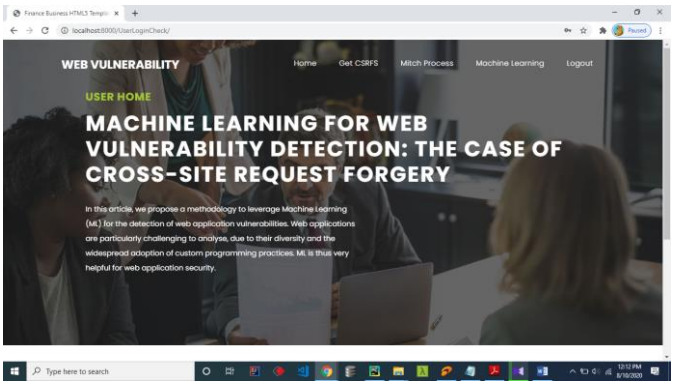
User Registration Form



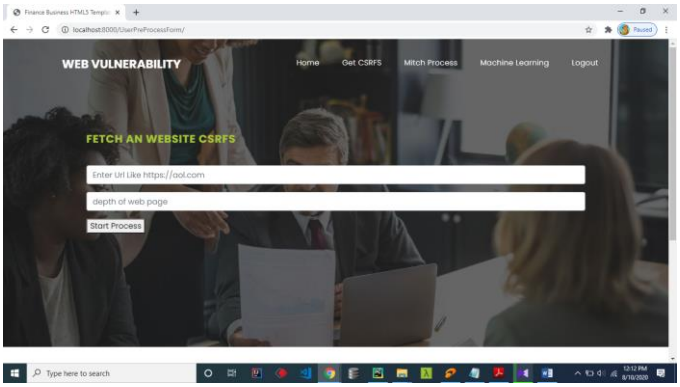
User Login Form:



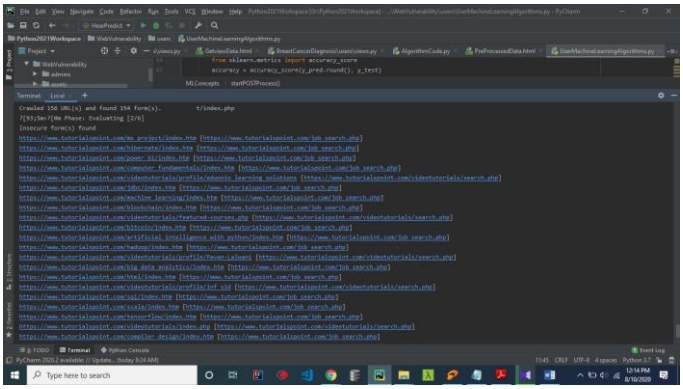
User Home:



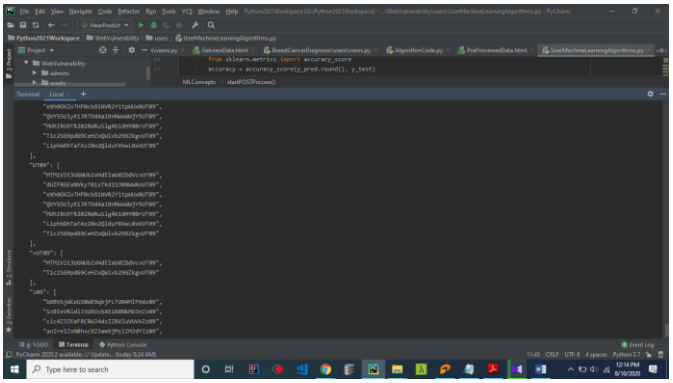
Getting website csrfs:



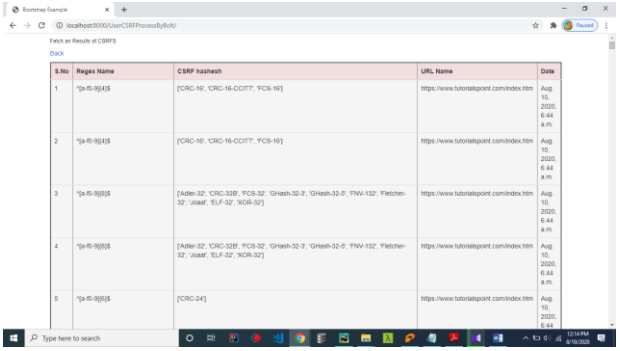
Scanning urls:



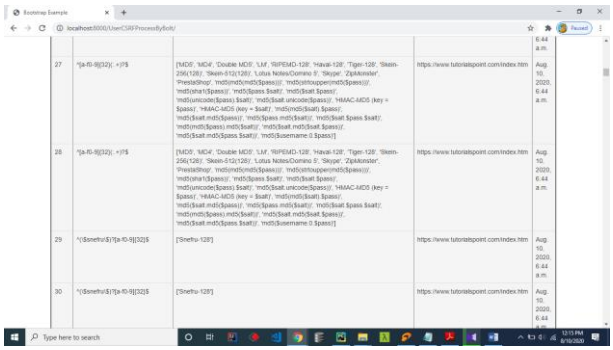
CSRF token:



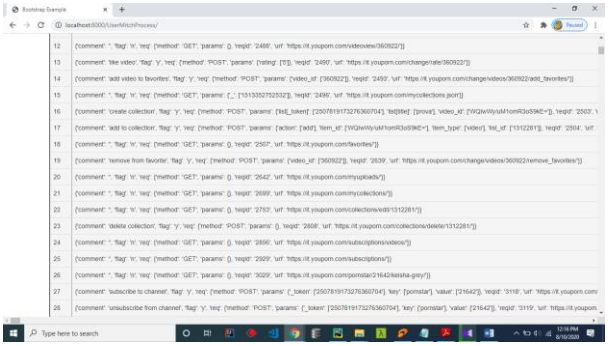
Given website csrf results



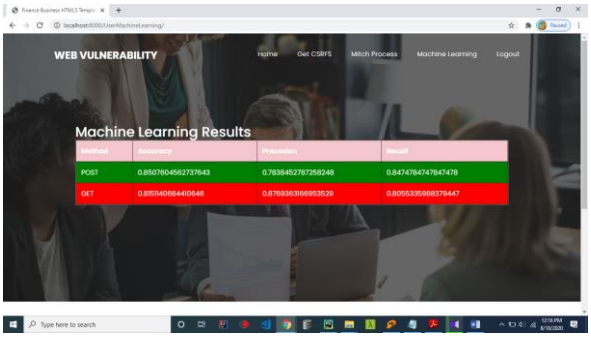
MD5 Token



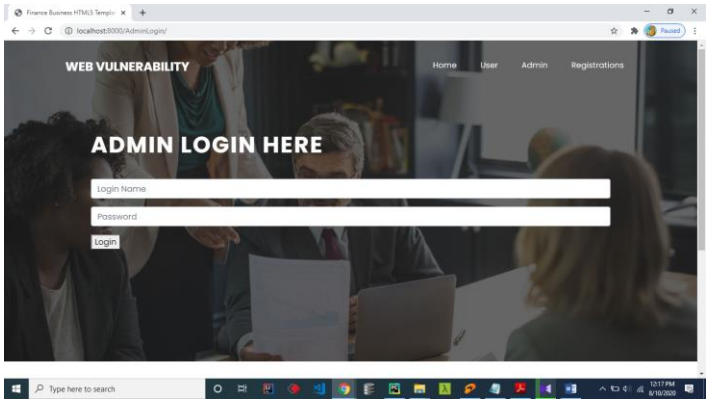
Mitch Detected sites:



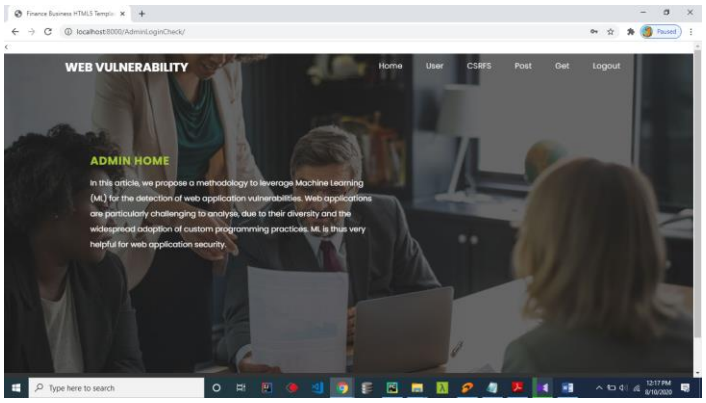
Machine Learning Results:



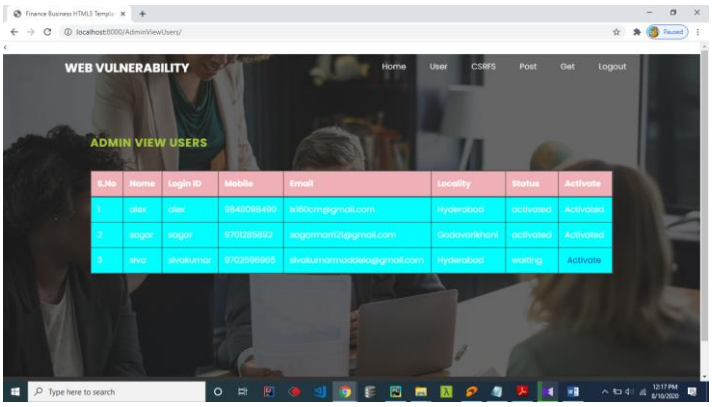
Admin Login:



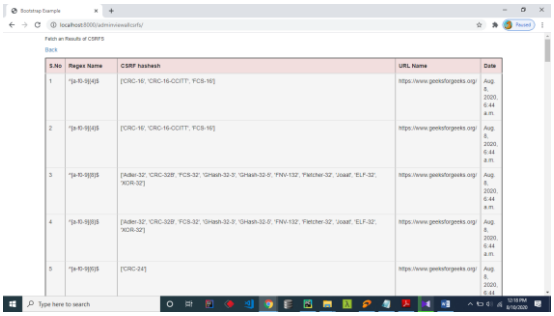
Admin Home Page:



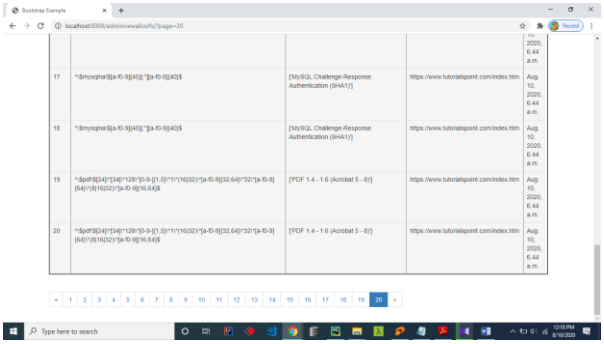
View Registered users:



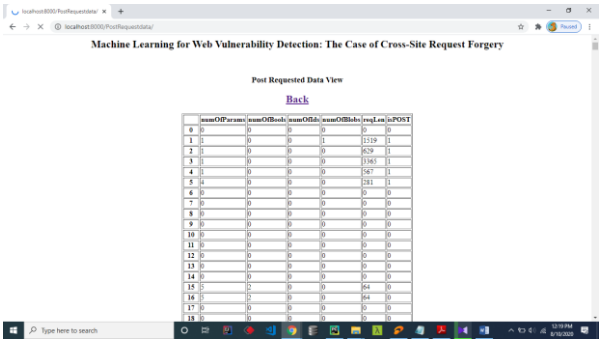
Admin View All CSRFS:



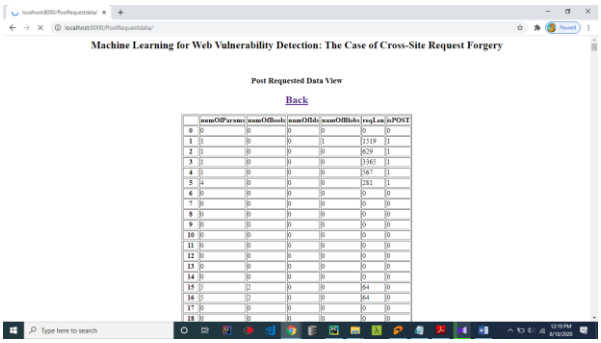
CSRFS:



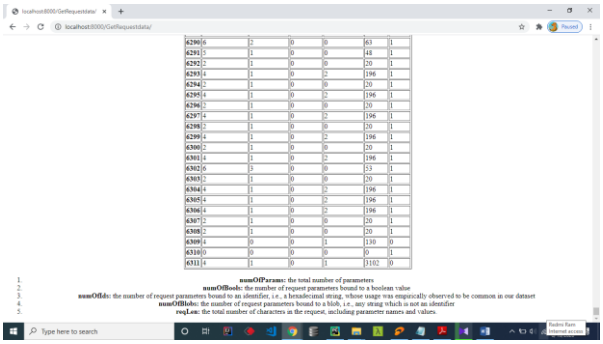
Post Data View:



Get Data:



Attribute Descriptions:



5. CONCLUSION

Web applications are particularly challenging to analyse, due to their diversity and the widespread adoption of custom programming practices. ML is thus very helpful in the web setting because it can take advantage of manually labeled data to expose the human understanding of the web application semantics to automated analysis tools. We validated this claim by designing Mitch, the first ML solution for the blackbox detection of CSRF vulnerabilities, and by experimentally assessing its effectiveness. We hope other researchers might take advantage of our methodology for the detection of other classes of web application vulnerabilities.

REFERENCES

[1] D. Atienza, Á. Herrero and E. Corchado, "Neural analysis of HTTP traffic for Web attack detection", Proc. Comput. Intell. Secur. Inf. Syst. Conf, pp. 201-212, 2015.

[2] P. Mishra, V. Varadharajan, U. Tupakula and E. S. Pilli, "A detailed investigation and analysis of using machine learning techniques for intrusion detection", IEEE Commun. Surveys Tuts., vol. 21, no. 1, pp. 686-728, 1st Quart. 2019.

[3] Y. Liu, S. Liu and X. Zhao, "Intrusion detection algorithm based on convolutional neural network", Beijing Ligong Daxue Xuebao/Trans. Beijing Inst. Technol., vol. 37, no. 12, pp. 1271-1275, 2017.

[4] Khalid, M.N., Farooq, H., Iqbal, M., Alam, M.T., Rasheed, K. (2019). Predicting Web Vulnerabilities in Web Applications Based on Machine Learning. In: Bajwa, I., Kamareddine, F., Costa, A. (eds) Intelligent Technologies and Applications. INTAP 2018. Communications in Computer and Information Science, vol 932. Springer, Singapore. https://doi.org/10.1007/978-981-13-6052-7_41

[5] Pan, Y., Sun, F., Teng, Z. et al. Detecting web attacks with end-to-end deep learning. J Internet Serv Appl 10, 16 (2019). https://doi.org/10.1186/s13174-019-0115-x

[6] A. Kumar and T. J. Lim, "EDIMA: Early Detection of IoT Malware Network Activity Using Machine Learning Techniques," 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), 2019, pp. 289-294, doi: 10.1109/WF-IoT.2019.8767194.

[7] R. K. Vigneswaran, R. Vinayakumar, K. P. Soman and P. Poornachandran, "Evaluating Shallow and Deep Neural Networks for Network Intrusion Detection Systems in Cyber Security," 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2018, pp. 1-6, doi: 10.1109/ICCCNT.2018.8494096.

[8] F. M. M. Mokbal, W. Dan, A. Imran, L. Jiuchuan, F. Akhtar and W. Xiaoxi, "MLPXSS: An Integrated XSS-Based Attack Detection Scheme in Web Applications Using Multilayer Perceptron Technique," in IEEE Access, vol. 7, pp. 100567-100580, 2019, doi: 10.1109/ACCESS.2019.2927417.

[9] V. Patil, P. Thakkar, C. Shah, T. Bhat and S. P. Godse, "Detection and Prevention of Phishing Websites Using Machine Learning Approach," 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), 2018, pp. 1-5, doi: 10.1109/ICCUBEA.2018.8697412.

[10] Şahin, C.B., Dinler, Ö.B. & Abualigah, L. Prediction of software vulnerability based deep symbiotic genetic algorithms: Phenotyping of dominant-features. Appl Intell 51, 8271–8287 (2021). https://doi.org/10.1007/s10489-021-02324-3