IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 10, Issue 6, June Month 2021

DOI 10.17148/IJARCCE.2021.106129

A Distributed Deep Learning System for Web Attack Detection on Edge Devices

Adithya M[1], Anand N[2], Arun Kumar S K[3], Prajwal V G[4], Dr. Gunavathi H S[5]

BE, Department of CSE, BIT, Bangalore, India [1, 2, 3, 4]

Assistant Professor, Department of CSE, BIT, Bangalore, India [5]

Abstract: Today's world is a digital world, where decisions are taken on the internet and the internet forms a very integral and important part of the society and the economy. Naturally, the internet's security, essentially the security of the World Wide Web (www) is very important and groundbreaking. The internet is often vulnerable to attacks from possible hackers who try to compromise the system in order to illegally poach the resources of the system under question. These attacks are famously called web attacks and are a very common problem amongst the computer fraternity. Though there are several existing systems to counter the problem of attacks on the web, most of these systems have their own drawbacks, as in they do not provide classification on any other grounds except frequency, thus causing many web attacking http requests to fall out of the bracket. The objective of our project is to detect these web attacks from the http requests based on many parameters, and classify them as web attacks or not. We also plan to further classify the attacks as HTML, JavaScript or SQL attacks, thus providing a novelty. Thus, the system solves the problem of undetected web attacks through http requests and thus increases the security of the system manifold.

Keywords: Supply World Wide Web (www), web attacks, web attacking http requests, HTML, JavaScript or SQL attacks.

I.INTRODUCTION

The conventional ways to perform the various tasks have reduced with the help of internet technology. Web applications also have grown in a dramatic fashion and used by various fields across the globe. With the usage of cloud based systems it has become easy for maintenance of applications. Each of the web application on cloud or on a dedicated server will be identified by using unique URL. There is more probability of the system being under attack and then hacking of the server can take place. Among the different kind of attacks web server attacks are prominent with a 75% occupation. There are two different paradigms of gains from web attack with the first one being access to personal data for the end users and second one is injecting script which can hack the data when user clicks or performs some kind of downloads. Each web application by various companies has to implement protocols to secure the system. However, the number of such attacks can be brought down by identifying and classifying them by reading the attributes of the request and then executing the machine learning algorithm.

II. OBJECTIVES

i. To accomplish the classification of HTTP request into anomaly or non-Anomaly and then anomaly is classified into HTML, SQL and JavaScript classifier.

ii. To minimize the amount of security issues for a web application.

iii. To implement a software web application which will have screens to view the data sets, stop words, addition or removal of stop words, and create a view for clean data sets, tokenization, and frequency.

iv. Design and implementation of weight feature vector computation in which the clean data sets are converted into sequence of words followed by computation of frequency, and then IDFT and FV are found.

III. LITERATURE SURVEY

Rizal Munadi, et al. [1] did a survey which analyses the security of various websites in the province of Aceh. SQL Injection and XSS attacks allow attackers to gain sensitive information which may harm the existence of the site itself. Also based on the type of attack, preventive measures are specified that need to be understood and implemented to minimize the number of SQL Injection incidents in the future.

Ravi Kishore K, et al. [2] proposed a system concerned about malware attacks, which usually happen when a user visits a website knowingly or unknowingly. This paper gives a thorough implementation of an extension called 'JS Guard' in the user's web browser which prevents the downloading of the malicious file in the first place, thereby preventing the attack. The JS Guard monitors the incoming web page and determines whether it has any vulnerable tags. The only

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

Vol. 10, Issue 6, June Month 2021

DOI 10.17148/IJARCCE.2021.106129

drawback with this is, JS Guard installed web browser have taken 180 ms more time for loading a webpage when compared with the page load time of the web browser when JS Guard is not installed.

M. Y. Kim, et al. [3] Web sites can be categorized into data holding and non-data holding. For data holding web site the web attacks can execute malicious scripts which can get the entities information for unauthorized and irrelevant users. The classification of the attacks is done using the process of SVM. The dimensions of the tree system are increased in order to perform the classification. The amount of time taken for classification is high, which is the major drawback of this paper.

Z. Yuan et al. [4] the network users make use of small hand held devices in large numbers, hence possess more threat of malware attacks along with privacy concerns. The hidden apps are placed within the normal apps which are responsible for web attacks. The hidden apps responsible for malware attacks can be tracked based on HTTP requests send or received from these apps. This theory has not yet got approval for worldwide usage.

J. Saxe, et al. [5] Deep neural network can be performed with the help of detection of malwares. The value of detection rate will be done based on low false positive and volumes of hardware. The concept of classification score can be utilized for our project in order to build the approach to compute the accuracy. The detection rate is very low, accuracy is about 40% and also requires huge amount of RAM.

Rathod Mahesh Pandurang, et al. [6] This paper discusses the harm of SQL injection with which a cracker can modify the backend and this paper also discusses Cross Site Scripting attack which is used to capture user input and also distort a user web view. This paper then proposes a Mapping Model in which requests are mapped on generated queries and can be used productively to detect such kinds of attacks and prevention logic can be applied for attack removal. The Containerization method used here adds some delay in response time and also some server overhead. Until 150 requests per second the containerized approach performs similar to Vanilla approach. But beyond this, this system shows some degradation.

Tabish Rashid, et al. [7] The paper discusses about detecting malicious insiders inside an organization using Hidden Markov Models which distinguishes anomalies in the model user's normal behaviour. As the user's behaviour usually changes from time to time, an evolving 'concept Drift' Model is used which is helpful in preventing the detection of non-anomalies as anomalies. The model requires setting up of threshold value manually which significantly affects the results. Feature extraction is another cumbersome process as it mainly depends on the information being provided. Choosing a hyper parameter, training a person for the first 5 weeks are other limitations of this work.

IV. PROBLEM STATEMENT AND PROPOSED SOLUTION

"To detect weather the HTTP request is malicious or not, using request attributes and classify it as a SQL, HTML or JavaScript anomaly by using Machine learning". Our aim is to detect whether the HTTP request is malicious or not, where word vector along with neural network is used with the hidden layers along with TF-IDF methodology. The word vector block and TF-IDF are combined together in order to determine whether a given request is a web attack or it is not a web attack. There are different categories of clusters which are considered in order to perform the category checks for the web attacks and the category includes the SQL, HTML and JavaScript.

V. SYSTEM ARCHITECTURE

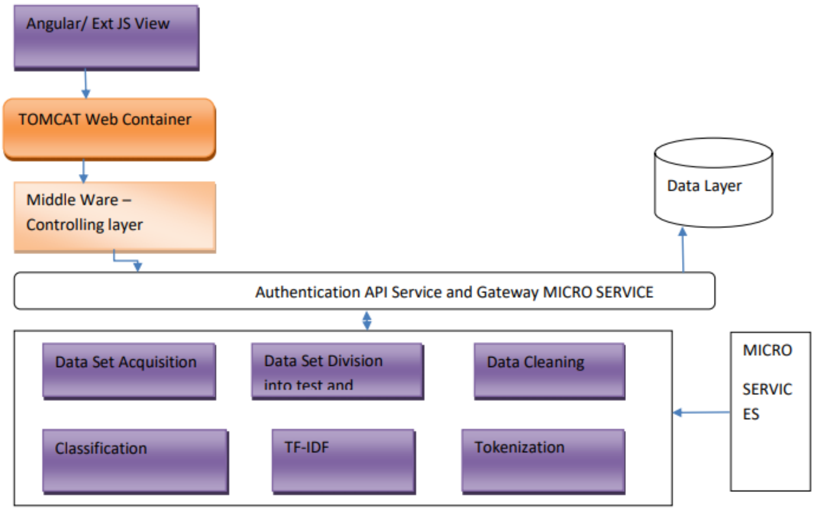


Figure 1 shows the system architecture of the proposed system and the following layers.

DOI 10.17148/IJARCCE.2021.106129

Figure 1 shows the system architecture. As shown in the figure there are various components which are involved in the working model for the project. These components are as follows:
Angular/Ext JS View For the development of the front end us done with the help of using angular and Ext JS framework along with java server pages.
TOMCAT Web Container: There are many servers available in the market which is responsible for handling the web requests. Most of the other servers are heavy weight and also are commercial in nature. Here we make use of open source and light weight tomcat server.
Middle Ware – Controlling layer: The request parameters from the front end and URL will be validated. If the request URL is valid then the form data will be validated and then, the request forwarding is done to the authentication layer. This also performs the basic validations like empty checks and regex validations. If any validation fails, then response is sent to the front end otherwise the request is forwarded to the authentication layer and respective services.
Data Layer: The data layer is responsible for storage of information related to registered users, admin, doctors. The data layer will also be able to store answers of users, appointment information as well as the classification information.
Authentication Layer: The users request is validated and after it is being checked whether the request contains valid application id and also has a valid session. User will be thrown out if the session is invalid.
Micro services: Each of the micro services are independent executing the business logic for the specific algorithm which can be data set collection operations, data cleaning operations, word stream operations, word stream count, inverse word stream count.

VI. SYSTEM DESIGN

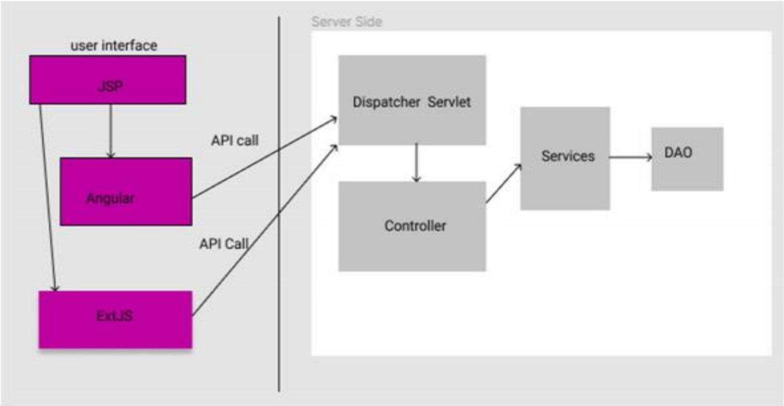


Figure 2 shows the UI of the project.

A UI is the key for the user to interact with the system. Figure 2 shows the interface architecture. As shown in the figure JSP page will refer angular as well as Ext JS frameworks to design the user interface and then API call will be made which will first go to dispatcher servlet which does high level validations, followed by controller which will have all the basic and advance validations, services will have all algorithms and business logic and DAO responsible for CRUD (Create Retrieve Update and Delete) operations. The algorithm responsible for storing undesired phrases in the program is shown in Figure 3. Creating Unwanted Words algorithm is in charge of coming up with an unwelcome word that isn't in the list of conventional unwanted words. If the unwanted term already exists, a validation error will be displayed; otherwise, the unwanted word will be produced.

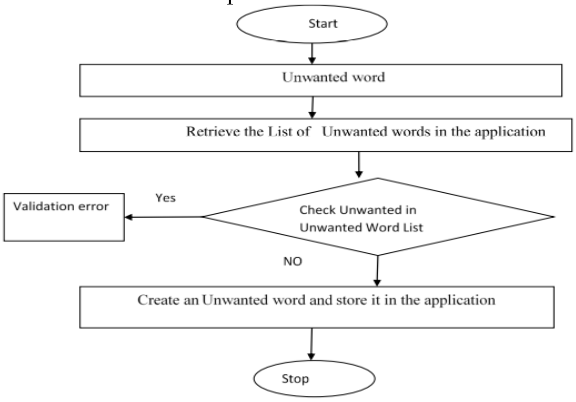

Start
Unwanted word
Retrieve the List of Unwanted words in the application
Validation error
Yes
Check Unwanted in Unwanted Word List
NO
Create an Unwanted word and store it in the application
Stop

Figure 3 shows the algorithm responsible for storing undesired phrases in the program.

DOI 10.17148/IJARCCE.2021.106129

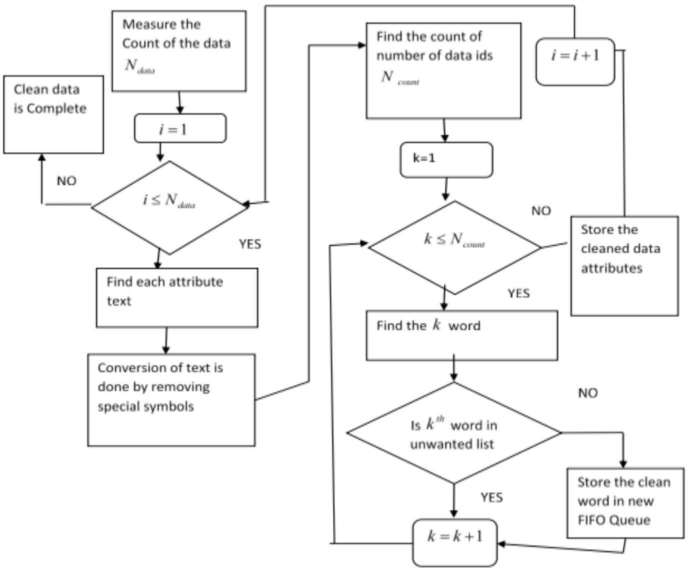

Measure the Count of the data N_{data}
Clean data is Complete
$i=1$
NO
$i \le N_{data}$
YES
Find each attribute text
Conversion of text is done by removing special symbols
Find the count of number of data ids N_{count}
$i=i+1$
k=1
NO
$k \le N_{count}$
YES
Store the cleaned data attributes
Find the k word
NO
Is k^{th} word in unwanted list
YES
Store the clean word in new FIFO Queue
$k=k+1$

Figure 4 depicts the quantity of data rows counted.

The major job is to employ clean data, which is obtained by cleaning each attribute by deleting any unneeded words and then removing any residual special symbols. Figure 4 depicts the quantity of data rows counted. From the first word to the last number of words, the process is repeated. After that, special symbols are deleted from each text attribute, and the number of counts for data ids is calculated. From the beginning of each word to the number of words in FIFO, each word is taken, checked, and shown to see whether there are any undesired words in the list. If the word is present, it is skipped; otherwise, it is added to the queue and incremented. This process continues till the task is accomplished.

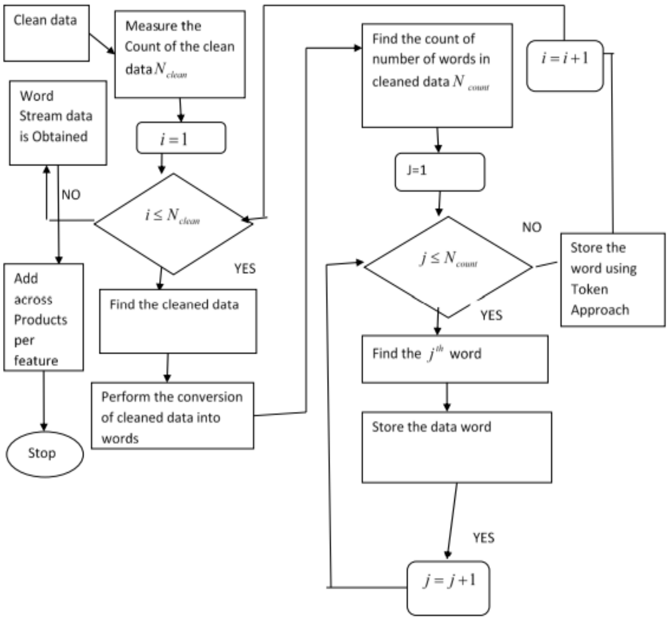

Clean data
Measure the Count of the clean data N_{clean}
Word Stream data is Obtained
$i=1$
NO
$i \le N_{clean}$
YES
Add across Products per feature
Find the cleaned data
Perform the conversion of cleaned data into words
Stop
Find the count of number of words in cleaned data N_{count}
$i=i+1$
J=1
NO
$j \le N_{count}$
YES
Store the word using Token Approach
Find the j^{th} word
Store the data word
YES
$j=j+1$

Figure 5 depicts word processing.

The conversion of cleaned text into a series of word stream values is handled by this streaming procedure. Figure 5 depicts word processing, in which cleaned text is obtained first, followed by conversion in a FIFO queue using a Delimiter. By that time, we've figured out how many words we've gotten. Now we classify each word, starting with the first and ending with the last, and save it with a unique representation id. Cleansed text is converted into a set of word stream values, which are then stored as a numeric value to represent count. The word stream processing is shown in Figure 6. First, a list of unique data ids is compiled, and the corresponding count is determined. Each unique id is found

DOI 10.17148/IJARCCE.2021.106129

starting with the first word and ending with the last word. Now we get the number of count of each unique word from the first data id list; this count of words repetition is done on specific data ids, and each word stream count is saved.

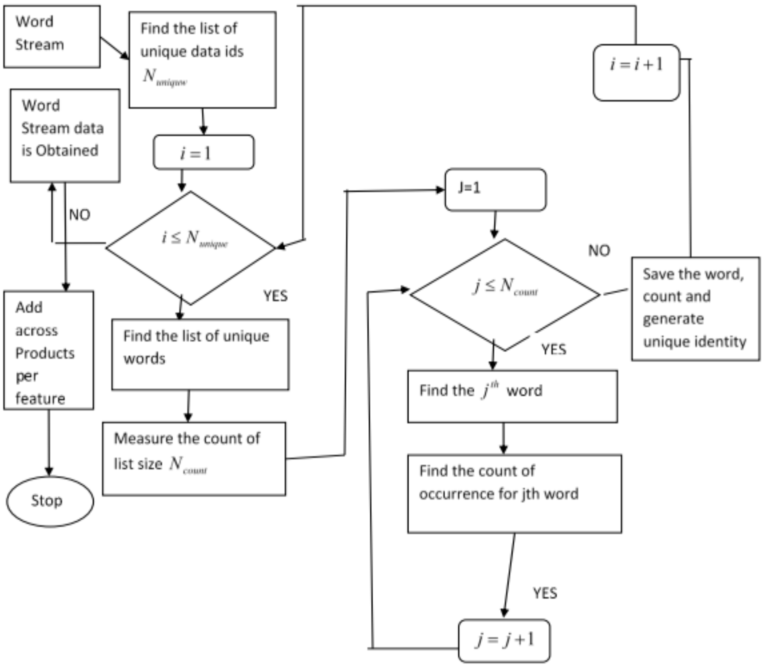

Word Stream
Find the list of unique data ids N_{unique}
$i = i + 1$
Word Stream data is Obtained
$i = 1$
J=1
NO
$i \le N_{unique}$
NO
$j \le N_{count}$
Save the word, count and generate unique identity
YES
Add across Products per feature
Find the list of unique words
YES
Find the j^{th} word
Measure the count of list size N_{count}
Find the count of occurrence for jth word
Stop
YES
$j = j + 1$

Figure 6 represents word stream count processing.

Algorithm: K-means Clustering:

- Purpose: Classification of HTTP request into anomaly or not.
- Input: The classified words related to various categories along with the total FV score for each category specific to dataset.
- Output: HTTP request is classified as anomaly and non-anomaly, and then based on the total FV score, anomaly is further classified into html, JavaScript and SQL.
- Different words related to various kinds of web attacks are obtained from the training vectors.
- Unique data ids are found by making use of TF-IDF matrix.
- The count of various web attack category words is obtained.
- The total feature vector for each of the data set based on the word and TF-IDF values is found.
- Compute the distance between the feature vector and the trained vectors.
- Find the minimum distance.
- The class label corresponding to the minimum distance is assigned a class respectively

Data Structure Design:

● JSON

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Python, and many others.

● JSON Web Tokens (JWT).

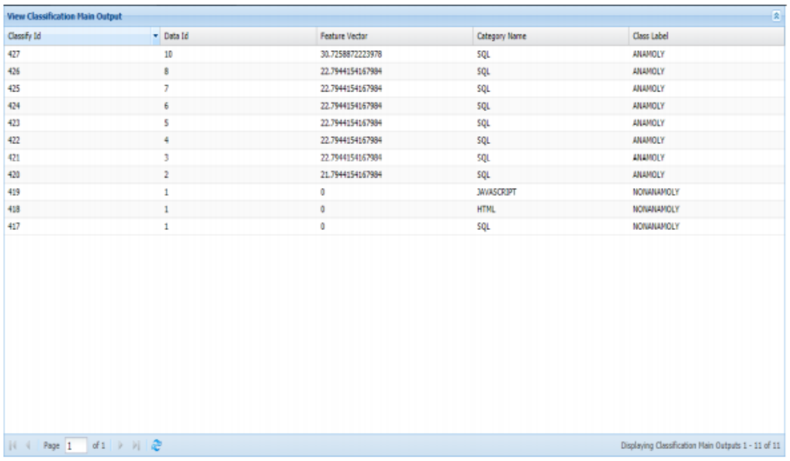
JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed. JWTs can be signed using a secret (with the HMAC algorithm) or a public/private key pair using RSA or ECDSA.

VII. RESULTS

The deep learning model using K-means clustering was used to build a web-attack detection system. Figure 7 shows the classification output of the project. Figure 8 shows the accuracy of the model after complete execution. To accomplish the classification of HTTP request into anomaly or non-Anomaly and then anomaly is classified into HTML, SQL and JavaScript classifier.

Vol. 10, Issue 6, June Month 2021

DOI 10.17148/IJARCCE.2021.106129



View Classification Main Output

Classify Id	Data Id	Feature Vector	Category Name	Class Label
427	10	30.7258872223978	SQL	ANAMOLY
426	8	22.7944154167984	SQL	ANAMOLY
425	7	22.7944154167984	SQL	ANAMOLY
424	6	22.7944154167984	SQL	ANAMOLY
423	5	22.7944154167984	SQL	ANAMOLY
422	4	22.7944154167984	SQL	ANAMOLY
421	3	22.7944154167984	SQL	ANAMOLY
420	2	21.7944154167984	SQL	ANAMOLY
419	1	0	JAVASCRIPT	NONANAMOLY
418	1	0	HTML	NONANAMOLY
417	1	0	SQL	NONANAMOLY

Displaying Classification Main Outputs 1 - 11 of 11

Figure 7 shows the classification output of the project.

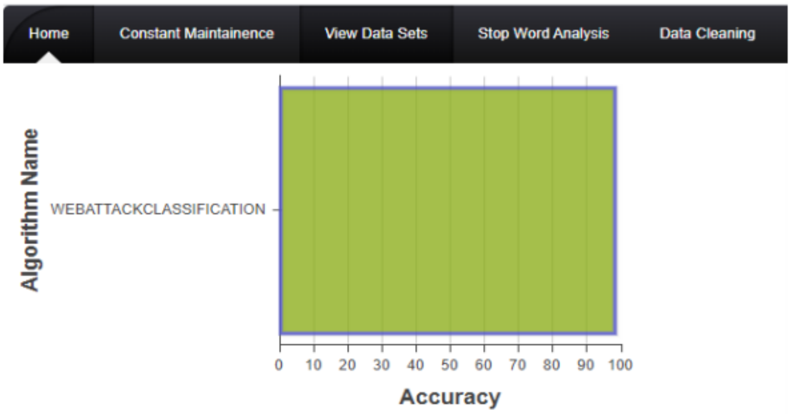


Figure 8 shows the accuracy measure of the model after execution.

VIII. Conclusions

Our system will take the data sets which contains web-attack and non-web-attack. The data sets undergo cleaning for removal of unwanted words and special symbols takes. After that the clean description is converted into a sequence of words. Then, redundancy is removed by doing a word stream count computation. After that weighted stream and number of datasets in which the word is present is calculated. Then, classification of data is done by using K-means algorithm. We have detected web attacks from the http requests based on many parameters, and classify them as web attacks or not. We also have classified the attacks as HTML, JavaScript or SQL attacks, thus providing a novelty. Thus, the system solves the problem of undetected web attacks through http requests and thus increases the security of the system.

IX. Future Scope

The main disadvantage of this project is that it was completed using a limited amount of datasets and no custom datasets were used. This project could be improved even further by including a subjective category for web attacks. Further tune the model for better performance.

References

[1] R. Munadi, T. S. Fajri, E. D. Meutia and E. Mustafa, "Analysis of SQL injection attack in web service (a case study of website in Aceh province)," 2013 3rd International Conference on Instrumentation, Communications, Information Technology and Biomedical Engineering (ICICIBME), Bandung, 2013, pp. 431-435, doi: 10.1109/ICICI-BME.2013.6698541.

[2] K. R. Kishore, M. Mallesh, G. Jyostna, P. R. L. Eswari and S. S. Sarma, "Browser JS Guard: Detects and defends against Malicious JavaScript injection based drive by download attacks," The Fifth International Conference on the Applications of Digital Information and Web Technologies (ICADIWT 2014), Bangalore, 2014, pp. 92-100, doi: 10.1109/ICADIWT.2014.6814705.

[3] M. Y. Kim, and D. H. Lee. (2014). Data-mining based SQL injection attack detection using internal query trees. Expert Systems with Applications. 41(11), pp: 5416-5430.

[4] Yuan, Zhenlong & Lu, Yongqiang & Wang, Zhaoguo & Xue, Yibo. (2014). Droid-Sec: Deep Learning in Android Malware Detection. ACM SIGCOMM Computer Communication Review. 10.1145/2619239.2631434.

DOI 10.17148/IJARCCE.2021.106129

[5] J. Saxe, and K. Berlin, "Deep neural network based malware detection using two dimensional binary program features," in International Conference on Malicious and Unwanted Software (MALWARE), IEEE, 2015.

[6] R. M. Pandurang and D. C. Karia, "A mapping-based podel for preventing Cross site scripting and sql injection attacks on web application and its impact analysis," 2015 1st International Conference on Next Generation Computing Technologies (NGCT), Dehradun, 2015, pp. 414-418, doi: 10.1109/NGCT.2015.7375152.

[7] T. Rashid, I. Agrafiotis, and J. RC Nurse, "A new take on detecting insider threats: exploring the use of hidden markov models," in Proceedings of the 8th ACM CCS International Workshop on Managing Insider Security Threats, ACM, 2016.