VISVESVARAYA TECHNOLOGICAL UNIVERSITY
"Jnana Sangama", Belagavi-590018

PROJECT REPORT
ON

DETECTION OF WEB ATTACKS BASED ON LOGS USING DEEP LEARNING

Submitted in partial fulfillment of the requirements of the degree of

BACHELOR OF ENGINEERING
in
INFORMATION SCIENCE & ENGINEERING
For the academic year
2020-2021

Submitted By

Architha J
(1GA17IS001)

B.U. Kavya
(1GA17IS005)

Sri Harsha A
(1GA17IS038)

Varsha Venkatesh
(1GA17IS045)

Under the guidance of
Mrs.Deepthi V S
Assistant Professor, Dept. of ISE
GAT, Bengaluru

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

GLOBAL ACADEMY OF TECHNOLOGY
Ideal Homes, Rajarajeshwari Nagar, Bengaluru-560098
(Affiliated to VTU, Belagavi and Approved by AICTE, New Delhi, NAAC Accredited with 'A' Grade)

GLOBAL ACADEMY OF TECHNOLOGY

Ideal Homes, Rajarajeshwari Nagar, Bengaluru-560098.

(Affiliated to VTU, Belgavi and Approved by AICTE, New Delhi, NAAC Accredited with 'A' Grade)

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

CERTIFICATE

Certified that the Project work entitled **'Detection of Web Attacks based on Logs using Deep Learning '** is a work carried out by **Architha J 1GA17IS001, B.U.Kavya 1GA17IS005, Sri Harsha A 1GA17IS038, Varsha Venkatesh 1GA17IS045,** bonafide students of **Global Academy of Technology, Bengaluru** in partial fulfillment for the award of the degree of Bachelor of Engineering in **Information Science & Engineering** of the **Visvesvaraya Technological University, Belagavi** during the year **2020-2021**. It is certified that all corrections/suggestions indicated for the Internal Assessment have been incorporated in the report deposited in the departmental library. The project Phase-II report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

Signature of the Guide	Signature of the HOD	Signature of the Principal
Mrs. Deepthi V S	**Dr. Kiran Y C**	**Dr. N. Ranapratap Reddy**
Assistant Professor,	Prof. & Head, Dept. of ISE,	Principal,
Dept. of ISE,		
GAT, Bengaluru.	GAT, Bengaluru.	GAT, Bengaluru.

Name of the Examiners	**Signature with date**
1.---	---
2.---	---

ABSTRACT

Internet is fast growing now a days, as more business activities are being automated and an increasing number of computers are being used to store sensitive information, the need for secure computer systems becomes more apparent. Today’s interconnected world makes everyone more susceptible to cyber-attacks. A cyber-attack is a malicious and deliberate attempt by an individual or organization to breach the information system of another individual or organization.

Web attacks are the ones which occur on a website or web applications. Some of the web-based attacks are: XSS, Injection attacks, DNS spoofing, Session hijacking, SQLi, Phishing, brute force, DOS and many more. Among these attacks, SQL injection and Cross-site scripting are frequently reported attacks. In these times, the data is highly vulnerable, and the attackers continuously find new ways to illegally retrieve them, as a result, the data should be protected from such attacks.

The proposed system detects SQLi and XSS attacks using the Deep Learning approach. A website vulnerable to SQLi and XSS attacks are created using PHP in the front end which is connected to the database. The deep learning algorithms are constructed using python. The query string entered by the user and its corresponding log data is given as input to the Intrusion detection system (deep learning algorithm), which detects the given input to be malicious or benign. Based on the result, the user is either allowed to access the website or blocked.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our effort with success.

We are grateful to our institution, **Global Academy of Technology**, with its ideals and inspirations for having provided us with the facilities, which has made this project a success.

We earnestly thank **Dr. N RanaPratap Reddy, Principal, Global Academy of Technology** for facilitating academic excellence in the college and providing us with the congenial environment to work in, that helped us in completing this project.

We wish to extend our profound thanks to **Dr. Kiran Y C, Prof. & Head, Department of Information Science & Engineering, GAT,** for giving us the consent to carry out this project.

We would like to express our sincere thanks to our internal guide **Mrs. Deepthi V S, Assistant Professor, Department of Information Science & Engineering, GAT,** for his/her able guidance and valuable advice at every stage, which helped us in the successful completion of the project.

We owe our sincere thanks to our project coordinator, **Mr. Dheeraj D, Assistant Professor, Department of Information Science & Engineering, GAT**, for his immense help during the project and also for his valuable suggestions on the project report preparations.

We would like to thank all the teaching and non-teaching staff for their valuable advice and support. We would like to express our sincere thanks to our parents and friends for their support.

Architha J	**B.U.Kavya**
(1GA17IS001)	**(1GA17IS005)**
Sri Harsha A	**Varsha Venkatesh**
(1GA17IS038)	**(1GA17IS045)**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Overview

A cyber-attack is any type of offensive action that targets computer information systems, infrastructures, computer networks or personal computer devices, using various methods to steal, alter or destroy data or information systems. It also includes criminal acts like hacktivist protests, harassment and extortion, money laundering, and more.

Web attacks: It is the attack in which some data will be injected into a web application to manipulate the application and fetch the required information. There are client-side vulnerabilities and server-side vulnerabilities which lead to a web application attack.

Most Common Types of Web Attacks:

- Cross-site scripting (XSS). This involves an attacker uploading a piece of malicious script code onto a website that can then be used to steal data. Although this strategy is relatively unsophisticated, it remains quite common and can-do significant damage.
- SQL Injection (SQLI). This happens when a hacker submits destructive code into an input form. If a system fails to clean this information, it can be submitted into the database where it can change, delete or reveal data to the attacker.
- Path traversal. Also resulting from improper protection of data that has been inputted, these web server attacks involve injecting patterns into the webserver hierarchy that allow bad actors to obtain user credentials, databases, configuration files and other information stored on hard drives.
- Local File Inclusion. This relatively uncommon attack technique involves forcing the web application to execute a file located elsewhere on the system.
- Distributed Denial of Service (DDoS) attacks. Such destructive events happen when an attacker bombards the server with requests. In many cases, hackers use a network of compromised computers or bots to mount this offensive. Such actions paralyze the server and prevent legitimate visitors from gaining access to the services.
- Server side includes (SSI): Server side includes injection vulnerability allows an attacker to exploit a web application by injecting scripts or SSI directives in HTML pages. SSI injection attack is done by manipulation of SSI used in the application

through user inputs.

- OS command injection (also known as shell injection) is a web security vulnerability that allows an attacker to execute arbitrary operating system (OS) commands on the server that is running an application, and typically fully compromise the application and all its data.

Figure 1.1 shows the most common web attacks. As observed, XSS and SQLi tops the list.

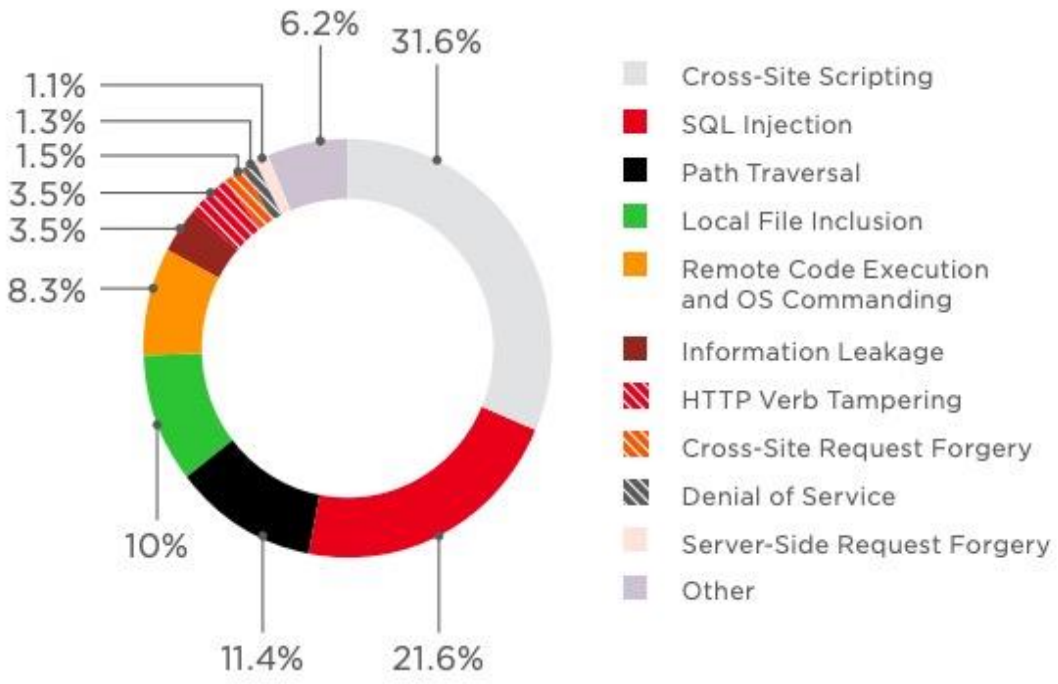


Figure 1.1: Most Common Web Attacks

A Denial-of-Service (DoS) attack is an attack meant to shut down a machine or network, making it inaccessible to its intended users. DoS attacks accomplish this by flooding the target with traffic, or sending it information that triggers a crash. In both instances, the DoS attack deprives legitimate users (i.e. employees, members, or account holders) of the service or resource they expected. There are two general methods of DoS attacks: flooding services or crashing services. Flood attacks occur when the system receives too much traffic for the server to buffer, causing them to slow down and eventually stop.

A wide array of tools and techniques are used to launch DoS-attacks. The simplest DoS attack relies primarily on brute force, flooding the target with an overwhelming flux of packets, oversaturating its connection bandwidth or depleting the target's system resources. Bandwidth-saturating floods rely on the attacker's ability to generate the overwhelming flux of packets. A common way of achieving this today is via distributed denial-of-service, employing a botnet.

A DDoS provide the attacker multiple advantages:

- He can leverage the greater volume of machine to execute a seriously disruptive attack
- The location of the attack is difficult to detect due to the random distribution of attacking systems (often worldwide)
- It is more difficult to shut down multiple machines than one
- The true attacking party is very difficult to identify, as they are disguised behind many (mostly compromised) systems

Cyber Security plays an important role in the field of information technology. Cyber security is the practice of defending computers, servers, mobile devices, electronic systems, networks, and data from malicious attacks. Securing the information has become one of the biggest challenges in the present day. Today Internet is the fastest growing infrastructure in everyday life. In today’s technical environment many latest technologies are changing the face of the mankind. But due to these emerging technologies we are unable to safeguard the private information in a very effective way and hence these days cybercrimes are increasing day by day. Web applications do raise a number of security concerns from improper coding. Serious vulnerabilities allow criminals to gain direct and public access to databases in order to steal sensitive data – this is known as a web application attack.

A neural network is a series of algorithms that endeavors to recognize underlying relationships in a set of data through a process that mimics the way the human brain operates. They are used in a variety of applications in financial services, from forecasting and marketing research to fraud detection and risk assessment.

Types of Neural Networks:

- **Perceptron:** It is the smallest unit of neural network that does certain computations to detect features or business intelligence in the input data. It accepts weighted inputs, and apply the activation function to obtain the output as the final result.

- **Feed Forward Neural Networks:** The simplest form of neural networks where input data travels in one direction only, passing through artificial neural nodes and exiting through output nodes. Where hidden layers may or may not be present, input and output layers are present there. Based on this, they can be further classified as a single-layered or multi-layered feed-forward neural network.
- **Multilayer Perceptron**: An entry point towards complex neural nets where input data travels through various layers of artificial neurons. Every single node is connected to all neurons in the next layer which makes it a fully connected neural network.
- **Convolutional Neural Network:** Convolution neural network contains a three-dimensional arrangement of neurons, instead of the standard two-dimensional array. Input features are taken in batch-wise like a filter. The network understands the images in parts and can compute these operations multiple times to complete the full image processing.
- **Radial Basis Function Neural Networks:** Radial Basis Function Network consists of an input vector followed by a layer of RBF neurons and an output layer with one node per category. Classification is performed by measuring the input's similarity to data points from the training set where each neuron stores a prototype.
- **Recurrent Neural Networks:** Designed to save the output of a layer, Recurrent Neural Network is fed back to the input to help in predicting the outcome of the layer. LSTM networks are a type of RNN that uses special units in addition to standard units. LSTM units include a 'memory cell' that can maintain information in memory for long periods of time.

1.2 Objectives

Cyber Security is an essential component of any company or enterprise across the world, hence the scope of Cyber Security is enormous. Cyber Security is the technology, process, and practice, designed to protect devices, programs, and data from damages, attacks, and other unauthorized access. Many authorized institutions, like the military, government agencies, financial institutions, Banking Sector, etc. have confidential information that is stored on computers and transmitted to networks. With growing cyber-attacks, it has become necessary to protect this sensitive data and personal information.

Here stopping identity theft isn’t the only goal, but protecting data integrity is also important. As cybercriminals are becoming more advanced, there is a need to understand their change in target, how are that affecting organizations, and their methods used in targeting.

1.3 Motivation

Cybercrime remains a growing challenge in terms of security and privacy practices. Today’s interconnected world makes everyone more susceptible to cyber-attacks. The threat of attacks on web applications to extract data or to distribute malicious code persists. Few security issues of web applications are XSS attacks, SQL injection, Local file inclusion, Path traversal and many more. Some of these attacks are designed to prevent competitors from participating in major events, while others target the complete shutdown of online businesses for months. Effects of these attacks include: Authentication bypass, information disclosure, compromised data integrity, user’s session hijacking, and access cookies.

CHAPTER 2

LITERATURE SURVEY

2.1 SQL Injection Detection Using Machine Learning

In [1], the author Sonali Mishra proposed a method that detects SQL injection attacks using a machine learning based heuristic algorithm. The pre-processed spreadsheet file which contained injected and non-injected SQL statements was imported into the proposed feature extractor. The feature extractor analyses the statements and generates a feature array for each. A MATLAB program was developed to investigate each statement and analyse it to produce an array of features. The features that were selected are: presence of comment character, number of semicolons, the number of commands per statement, presence of always true conditions, presence of abnormal commands, and presence of special key words. The dataset consists of 616 SQL statements. The MATLAB classification learners were used to train and test the dataset with 23 different machine learning algorithms. Based on the accuracy of the results the best five classifiers to develop the proposed SQL injection detection system were selected. The proposed system was extensively tested, and the results showed that both Ensemble Boosted and Bagged Trees classifiers provided the highest classification accuracy (93.8%). The system was also scalable to easily implement minor modifications. To enhance the efficiency of the system, more non-injected SQL statements need to be considered in the dataset and more features need to be studied and experimented in future.

2.2 Detection of SQL injection attacks: A machine learning approach

In [2], the authors of the paper Musaab Hasan, Zayed Balbahaith, and Mohammed Tarique, proposed the usage of machine learning algorithm called Gradient Boosting Classifier from ensemble machine learning approaches to classify and detect SQL Injection attacks. Naive Bayes classifier was used to classify between malicious and non-malicious SQL queries. To train the model they have used a training dataset that consists of both malicious and non-malicious SQL queries and also every query in the training data was labelled. The dataset was divided into two parts. The first part consists of plain text sentences of 4000 rows and the second part consists of SQL injected statements of 6000 rows that were created from a tool named Libinjection. Regex in python was used for tokenizing each entry in both the SQL Injection and plain-text datasets. After tokenization of the dataset, feature extraction was performed on the data and G-test scores; entropy was calculated for all token values in dataset. The prediction accuracy of Naïve Bayes classifier was 92.8% and gradient boosting classifier was 97.4%. The results from both experiments show that Gradient Boosting approach does perform better in terms of prediction accuracy. The model identified all three types of SQL Injection attacks. Gradient boosting has several trade-offs: gradient boosting is computationally expensive than simple classifiers in terms of memory and computation. They are more susceptible to overfitting and tend to take longer time in learning phases. The machine learning model can also be advanced further with better feature extraction.

2.3 Log based Dynamic Intrusion Detection of Web Applications

In [3] the author Harsh Bhagwani proposed a method that uses logs for intrusion detection of web applications. This aims to address the classification of malicious HTTP requests received by a web application and identification of the smallest attack requests. The dataset consists of a total of 50116 samples, out of which 35006 were valid requests and 15110 were attack requests. Dataset has been created using Torpeda framework. Data processing involves URL Decoding of the requests, Lowercasing the requests (except CRLF where we are uppercasing the requests) and Replacing some characters. These features are labelled as 0/1 showing whether a particular keyword or punctuation is present or not in the HTTP request and these features have been validated with all the classifiers using k-fold cross validation with k=10. The 6 classifiers were used to detect all the attacks. Random Forest model provides accuracy of 99.34% was considered best classifier to detect XSS. For SQLi, Random Forest classifier provides highest accuracy of 97.91%. Gradient boosting and Random Forest classifiers provide best accuracy of 99.28% while detecting Path traversal attack. Lightweight Directory Access Protocol (LDAP) Injection was detected better with SVM classifier with accuracy of 99.89%.

XML Path (XPath) Injection was detected by Random Forest with high accuracy of 99.89%. Gradient boosting was able to detect OS command injection with the accuracy of 98.13%. Server Side Includes (SSI) Injection was detected best by Random Forest Algorithm with accuracy of 99.89%. Carriage Return Line Feed (CRLF) was detected by all the classifiers with accuracy of 100%. Any Anomaly Detection was detected by Random forest with accuracy of 96.92%. The models which were used have shown good results in the discovery of attacks. The proposed work gives a good average detection accuracy of 99.02%, and use of subsequent attack models has made the system more robust.

2.4 SQL Injection Detection for Web Applications Based on Elastic-Pooling CNN

In [4] the authors Xin Xie, Chunhui Ren, Yusheng Fu , Jie Xu , and Jinhong Guo proposed a method to detect SQL injection based on Elastic-Pooling CNN (EPCNN).Elastic-Pooling CNN uses text-CNN and spp-net can output fixed size of one-dimensional vectors one-dimensional vectors when non-fixed size matrices input to them. These methods can realize text classification of different lengths and image recognition of different sizes, but the output is a one-dimensional vector.In Data pre-processing Word2vec method is used to vectorize each original query string and the single character is used as the minimum training unit Elastic-Pooling CNN for SQL injection detection mainly uses regular matching. Regular technology has high recognition accuracy and speed. Elastic-Pooling CNN methods cannot identify new attacks. The accuracy obtained using Elastic-Pooling CNN is 99%.

2.5 Detection of SQL Injection Attacks by Removing the Parameter Values of SQL Query

In [5], the authors Rajashree A. Katole, Dr. Swati S. Sherekar, Dr. Vilas M. Thakare, proposed a method that detected SQL injection attacks by using a combined static and dynamic analysis. The attribute values of a fixed parameter value are the static query. Dynamic queries are the attribute values with dynamic pretender values. These two queries are compared and if they are the same then it is a normal query, else it is a malicious query. It could detect all types of SQL injection. A total of 120 questions were issued for each page of which 45 were malicious and 75 were normal. All the 75 normal queries were labelled normal and 40 out of 45 malicious queries were detected., hence raising the detection rate to 96%. The work on efficient methods for detecting the SQL injection attack and methods to prevent it. A less time must be consumed to detect the SQL injection for more new and robust methods are to be developed. Impact on business must be understood to reduce the risk of SQLi attack.

2.6 Research on SQL injection detection technology based on SVM

In [6], Zhuang Chen, Min Guo, Lin zhou, proposed an approach that detects SQL injection based on SVM. The dataset consists of black and white samples. 1000 samples of SQL injected in GitHub and exploitdb. The texts in the dataset are broken into word segmentation and one hour encoded. Word to Vec is a way to express text features as N-dimensional vectors. Words such select, Union, null, where are marked and others are marked as none. Support vector machine algorithm is used to classify the SQLi which provides an accuracy of 95%. However, there are some problems faced in practical application. In order to identify malicious behaviour from a large number of labels and combinations of the functions and high-quality dataset. Secondly, word 2vec saves the text vector into memory. The memory required for model training depends on the size of the data

2.7 Detecting Cross-Site Scripting Attacks using Machine Learning

In [7] the authors Fawaz A. Mereani and Jacob M. Howe, proposed machine learning approach to detect stored XSS with high accuracy and precision. The paper collects both malicious and benign scripts in order to balance dataset. For the training set, malicious scripts were obtained from developer sites, a selection from XSSed, the largest online archive of XSS vulnerable websites and additional scripts were collected by crawling sites known to be untrustworthy. Structural Features: They contain complete set of non-alphanumeric characters that can occur in JavaScript. Behavioural Features: These are a selection of the commands and functions that can be used in JavaScript. The feature data is used as input for supervised learning algorithms. In this work, support vector machines (SVM), k-nearest neighbour (k-NN), and Random Forests. Two variations on SVMs are used, with a linear kernal and with a polynomial kernal. The training dataset was divided at random into five folds, with training on four of the five folds. SVM (Linear Kernel) Evaluation provided an average accuracy of 94.74%. SVM (Polynomial Kernel) Evaluation provides average accuracy of 97.06%. 97.12% average accuracy was obtained using k-NN Classifier Evaluation. Random Forest Classifier Evaluation provides average accuracy of 97.22%. These classifiers can be added as a security layer either in a browser or (as intended) on a server. Future work is to investigate these aspects, as well as to use the same features with a Neural Network classifier.

2.8 Detecting Web Attacks Using Multi-Stage Log Analysis

In [8] the authors Melody Moh, Santhosh Pininti, Sindhusha Doddapaneni, and Teng-Sheng Moh, proposed a method to detect web attacks using multi-stage log analysis.Most existing solutions for detecting these attacks use log analysis, and employ either pattern matching or machine learning methods. One commonly used pattern matching methods is ELK (Elasticsearch, Logstash, and Kibana). Pattern matching methods can be effective, dynamic they however cannot detect new kinds of attacks. Supervised machine learning methods can detect new attacks. This work proposes a multi-stage log analysis architecture, which combines both pattern matching and supervised machine learning methods. It uses logs generated by the application during attacks to effectively detect attacks and to help preventing future attacks. the two-stage system has combined the advantages of both systems, and has substantially improved the detection accuracy. The multi-stage log analysis concept would be highly applicable to many intrusion detection applications. Single-Stage Architecture: Using Kibana, the pattern matching method results in 85.3% accuracy in detecting the SQL injections. Using Machine Learning method results in 80.04% accuracy. Multi-stage Architecture: Pattern Matching followed by Machine Learning this combination has achieved 94.7 % accuracy. Machine Learning followed by Pattern Matching this combination has achieved 95.4% accuracy.

2.9 Cross Site Scripting (XSS) Attack Detection Using Intrusion Detection System

In [9], the authors Kunal Gupta, Rajni Ranjan Singh, Manish Dixit, proposed a system that detects Cross-Site Scripting (known as XSS) attack using Intrusion Detection system (IDS). Here attack signature was utilized to detect XSS attack. To test the usefulness and effectiveness of proposed work a proof-of-concept prototype was implemented using SNORT IDS. The process includes capturing of all the packets that are in transition on the network and selection of relevant packets that are filtered. Here selective packet capturing is done. Next, signature-based detection was applied on the selective packets in this case only TCP packets were used. After detection, the alerts were generated, and metadata logs were created. A Cisco tool SNORT IDS was installed on the Network to inspect the network traffic data and then packets were filtered according to rules provided on the tool Snort. The components of snort architecture included: The sniffer, The pre-processor, The detection engine and the output. When the console was run, all attacks were detected effectively then it created alerts and logs successfully. After successful implementation of rule, XSS attack was able to be detected through Alert File containing details of the attack and attack was also logged in the form of dumps as metadata in same directory in snort. Experiments were done in real network environment. Few false-positive were generated, but it caught all the scripting attack successfully. The speed of the detection process can be increased using other keywords with Perl Compatible Regular Expression (PCRE) in future.

2.10 XSS Classifier: An Efficient XSS Attack Detection Approach Based on Machine Learning Classifier on SNS.

In [10], the authors Shailendra Rathore, Pradip Kumar Sharma, Jong Hyuk Park, proposed a system that detects XSS attacks using machine learning classifier on SNS. Social Networking servicing applications such as Twitter, Facebook have certain features on the webpages which make these applications more susceptible to the XSS attack. The proposed approach has four major steps: feature identification, collection of webpages, feature extraction and training dataset construction, and machine learning classification. The paper lists three main features: URL feature, HTML tag feature, SNS features. In the second step collecting webpages, A database is created by collecting malicious and benign webpages from internet resources such as XSSed, Alexa, Elgg. Each webpage is manually labeled as XSS or non-XSS based on extracted features and the training dataset is 1000 webpages in which 400 malicious and 600 benign webpages were constructed. Decorate and ADTrees were the two machine learning classifiers used to classify the instance. ADTree provides an accuracy of 97% and Decorate provides an accuracy of 96%. The two major contributions in the area of SNS. Firstly, it provides an analysis on recent characteristics of SNS. Secondly, the paper purposed a machine learning approach to detect XSS attacks. In the future, the enhancement in the proposed future set is required. Secondly, it can enhance its application to advanced machine learning algorithms such as deep learning and extreme machine learning.

CHAPTER 3

HIGH LEVEL DESIGN

3.1 Existing System

There are several existing systems to prevent SQL injection and XSS attack which faced few drawbacks. For example, many systems perform pattern matching or provide the query to machine learning algorithms or both to detect SQLi queries. In such cases, queries provided to the system must be preprocessed in order to increase the accuracy provided by the algorithm. Some systems use heuristic algorithm to prevent the SQL injection attack to overcome the drawback of pattern matching, but these algorithms cannot detect the non-injected SQL statements are they are mainly concerned with sqli queries. The ensemble machine learning algorithms such as gradient boosting and naïve bayes classifier approach were used in the system but Gradient Boosting approach is computationally expensive than simple classifiers in terms of memory and computation. More memory is needed to store multiple trees.

SQL injection queries can be detected by processing the text data in HTTP request packets, these data are converted to vectors by word2vec method and vectors can be sent to SVM machine learning algorithm. The practical disadvantage of this method is that word2vec is consumes memory as it is dependent on the size of training dataset. Queries can be detected without machine learning algorithm by comparing the static and dynamic queries although this method can detect many types of sql injection queries; they consume time to compare the query with all the static queries.

Existing XSS prevention system detect XSS script using snort rule, after successful implementation of rule, XSS attack was able to be detected through Alert File containing details of the attack and attack was also logged in the form of dumps. However, some false-positive were generated. Efficient machine learning algorithm such as decorate and ADTree are used, but enhancement of feature set was required.

3.2 Proposed System

A number of approaches have been proposed to detect XSS and SQLi attack vulnerabilities. Some of the existing approaches used machine learning algorithms such as Naïve bayes classifier, ensemble algorithms, decorates algorithms, AD Trees, elastic pooling CNN. There are several drawbacks of these methods and some of them are not compactable with the updated technologies. To overcome these drawbacks, the proposed system uses log based deep learning approach to detect XSS and SQLi attacks. A website vulnerable to SQLi and XSS attacks is created using PHP in the front end which is connected to the SQL database. The deep learning algorithms are constructed using python. Web log file is log file automatically created and maintained by a web server. This contains information about who was visiting the site, where they came from, and exactly what they were doing on the web site.

Objectives of proposed framework:

1. Detecting the presence of an attack on the website.
2. Classification of SQL injection attacks.
3. Classification of Cross-site scripting (XSS) attacks.
4. Prevention of SQL injection and Cross-site Scripting attacks.

Outcomes of proposed framework:

1. Brief understanding and effectuation of SQLi and XSS attacks.
2. Analysis of web log files to detect SQLi and XSS attacks.
3. Understanding and implementation of deep learning algorithms.
4. Preventing the risk of SQLi and XSS attacks.

3.3 System Requirements

These are the following requirements for the system to be able to have the implementation of the model. The main requirements are listed as below:

3.3.1 Software Requirement Specification

A software requirements specification (SRS) is a detailed description of a software system to be developed with its functional and non-functional requirements.

A software requirements specification (SRS) is a detailed description of a software system to be developed with its functional and non-functional requirements. The SRS is developed based the agreement between customer and contractors. It may include the use cases of how user is going to interact with software system. The software requirement specification document consistent of all necessary requirements required for project development. To develop the software system we should have clear understanding of Software system. To achieve this we need to continuous communication with customersto gather all requirements.

A good SRS defines the how Software System will interact with all internal modules, hardware, communication with other programs and human user interactions with widerange of real-life scenarios. Using the SRS document on QA lead, managers creates test plan. It is very important that testers must be cleared with every detail specified in this document in order to avoid faults in test cases and its expected results.

3.3.2 Functional Requirements

A functional requirement defines a function of a system or its component, where a function is described as a specification of behaviour between outputs and inputs. Functional requirements may involve calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish.

A typical functional requirement will contain a unique name and number, a brief summary, and a rationale. This information is used to help the reader understand why the requirement is needed, and to track the requirement through the development of the system.

3.3.3 Non-functional Requirements

A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviours. They are contrasted with functional requirements that define specific behaviour or functions.

3.3.4 Hardware Requirements

Processor	:	Intel Core i5
Hard Disk	:	50 GB Free Space.
Ram	:	8 GB(Minimum)
Display	:	Monitor(1024*768 pixels) or higher resolution monitor with 32 bit color settings.

3.3.5 Software requirements

Operating System	:	Kali Linux, Windows
Programming language: Back - End	:	Python
Programming language: Front - End	:	HTML, PHP, CSS, Bootstrap,JavaScript
Development Environment	:	Jupyter Notebook/ any other Python IDE
Application Server	:	XAMPP
Database	:	MYSQL

3.4 System Architecture

System architecture is the conceptual model that defines the structure, behaviour, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviours of the system. System architecture conveys the informational content of the elements consisting of a system, the relationships among those elements, and the rules governing those relationships.

The architectural components and set of relationships between these components that an architecture description may consist of hardware, software, documentation, facilities, manual procedures, or roles played by organizations or people. The module breakdown of blink to text communication and sentiment-based music system can be seen in Figure 3.1. A system architecture primarily concentrates on the internal interfaces among the system's components or subsystems, and on the interface(s) between the system and its external environment, especially the user.

Various organizations can define systems architecture in different ways, including:

- The fundamental organization of a system, embodied in its components, their relationships to each other and to the environment, and the principles governing its design and evolution.
- A representation of a system, including a mapping of functionality onto hardware and software components, a mapping of the software architecture onto the hardware architecture, and human interaction with these components.
- An allocated arrangement of physical elements which provides the design solution for a consumer product or life-cycle process intended to satisfy the requirements of the functional architecture and the requirements baseline.
- Architecture consists of the most important, pervasive, top-level, strategic inventions, decisions, and their associated rationales about the overall structure (i.e., essential elements and their relationships) and associated characteristics and behaviour.
- A description of the design and contents of a computer system. If documented, it may include information such as a detailed inventory of current hardware, software and networking capabilities; a description of long-range plans and priorities for future purchases, and a plan for upgrading and/or replacing dated equipment and software.

Figure 3.1 shows the architecture of the proposed system.

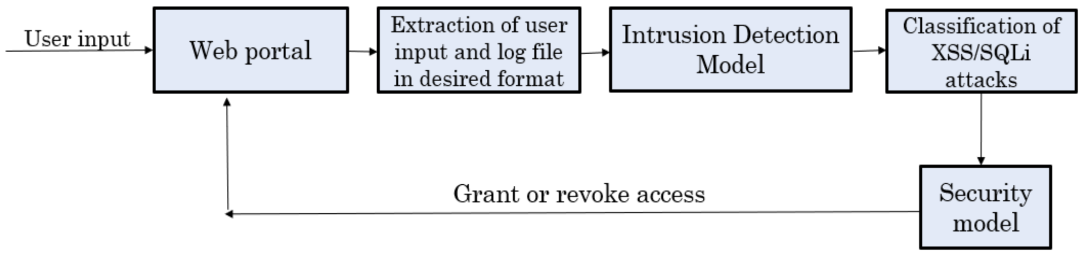

User input
Web portal
Extraction of user input and log file in desired format
Intrusion Detection Model
Classification of XSS/SQLi attacks
Security model
Grant or revoke access

Figure 3.1: System Architecture

Components:

- Web Portal
- User Input extraction and log data
- Intrusion Detection System
- Classification of SQLi/XSS attack
- Security Model

3.5 Data Flow Diagram

A data-flow diagram is a way of representing a flow of a data of a process or a system (usually an information system). The data-flow diagram also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow; there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.

Components:

1) Process:

The process (function, transformation) is part of a system that transforms inputs to outputs. The symbol of a process is a circle, an oval, a rectangle or a rectangle with rounded corners (according to the type of notation). The process is named in one word, a short sentence, or a phrase that is clearly to express its essence.

2) Data Flow:

Data flow (flow, dataflow) shows the transfer of information (sometimes also material) from one part of the system to another. The symbol of the flow is the arrow. The flow should have a name that determines what information (or what material) is being moved. Exceptions are flows where it is clear what information is transferred through the entities that are linked to these flows.

3) Warehouse:

The warehouse (data store, data store, file and database) is used to store data for later use. The symbol of the store is two horizontal lines; the other way of view is shown in the DFD Notation. The name of the warehouse is a plural noun (e.g. orders) - it derives from the input and output streams of the warehouse. The warehouse does not have to be just a data file, for example, a folder with documents, a filing cabinet, and optical discs.

4) Terminator:

The Terminator is an external entity that communicates with the system and stands outside of the system. It can be, for example, various organizations (e.g. a bank), groups of people (e.g. customers), authorities (e.g. a tax office) or a department (e.g. a human-resources department) of the same organization, which does not belong to the model system. The terminator may be another system with which the modelled system communicates.

Figure 3.2 describes the flow of data throughout the proposed system.

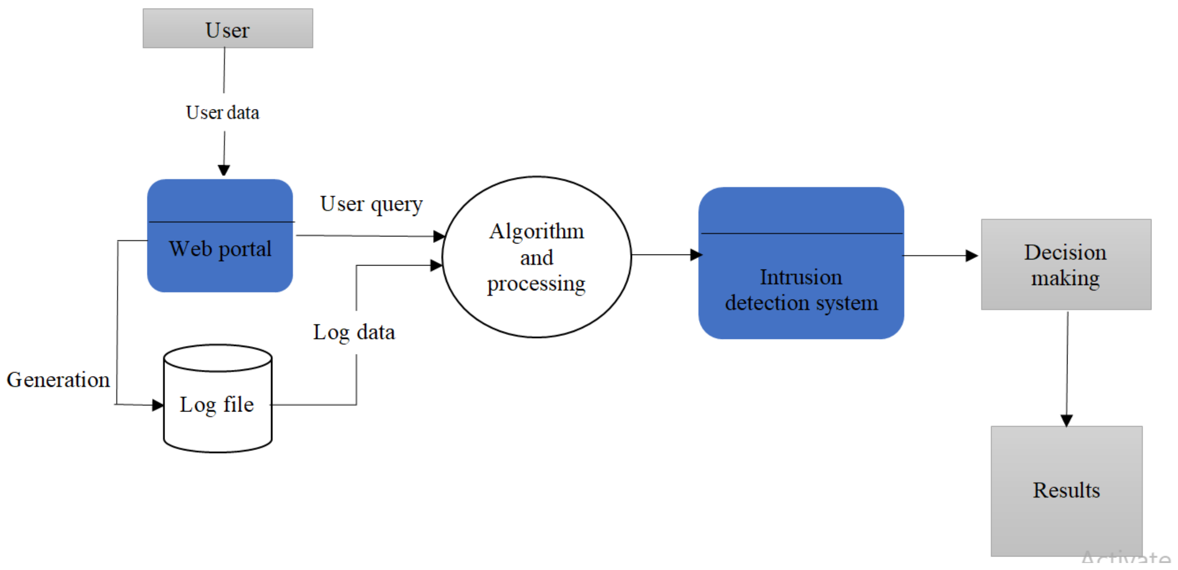


Figure 3.2: Dataflow Diagram

- In the above shown Figure 3.2, The user can enter data in different pages of the web portal such as login page, registration page, search page etc.
- Log data is generated using the user's data and other information of the user such as IP address, browser client, protocol used etc and this information is stored in a log file.
- The user data (query string) and its corresponding log data is given as input to the Deep learning algorithm (Intrusion detection system).
- The IDS classifies the user input into malicious input or benign input based on the

log data and user input.

- The IDS decides whether to allow the users request or to decline the users request based on the Classification made.
- If the request is valid or classified as benign, then it is processed and response is sent back to the user.
- If the request is invalid or classified as malicious, the request is declined.

CHAPTER 4

LOW LEVEL DESIGN

4.1 System Architecture Breakdown

Figure 4.1 is breakdown of the system architecture into query string and log module.

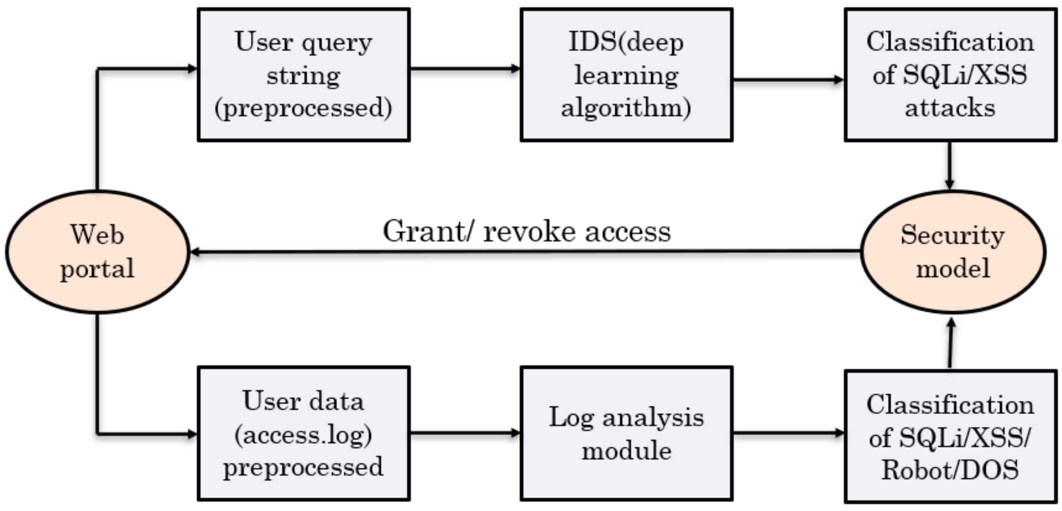

User query string (preprocessed)
IDS(deep learning algorithm)
Classification of SQLi/XSS attacks
Web portal
Grant/ revoke access
Security model
User data (access.log) preprocessed
Log analysis module
Classification of SQLi/XSS/ Robot/DOS

Figure 4.1: System Architecture Breakdown of each module

4.1.1 Module-1: Classification of query string

- Description: Analysis of query string and classifying it into sqli, xss or a normal input.
- Methodology:

➢ Sub module 1 - Detection of SQLi : A dataset was prepared using the kali-Linux tool "sqlmap". Sqlmap allows injecting malicious queries into our website, since some queries cannot be included by the tool these are included manually.

The dataset is preprocessed through CountVectorizer available in sklearn.feature_extraction.text. The CountVectorizer provides a simple way to both tokenize a collection of query statements and build a vocabulary of known words. This encoded vector is a sparse matrix that is provided as input to the multilayer neural network for training and testing.

This model is saved into a JSON file and weights are stored in the hdf5 file. The model is loaded to predict the preprocessed test instance into a binary classification. Based on the result provided to the front end, web portal can grant or deny access to the user.

➢ Sub module 2: Detection of XSS: A dataset was prepared using the kali-Linux tool "owasp". Since the dataset contains Chinese characters such as œ�, š, ^, Ÿ, it is necessary to remove them, this can be done by converting it into ASCII characters for the preprocessing of data.

These values are stored into numeric array are passed into convoluted neural network model for training and testing. For Further use, the model is saved into a JSON file and weights are stored in the hdf5 file and is loaded to predict the preprocessed test instance into a binary classification.

Another model, with 63 feature set was built to detect xss attacks. The features are selected manually and passed through a multi-layer perceptron. The model is saved and loaded when the script is called.

- ➢ Input: Query string
- ➢ Output: Binary classification(Malicious or not)

- Input: Query string
- Output : Binary classification(Malicious or not)

4.1.2 Module-2 : Classification using Log data

- Description: Analysis of log data from the website and classifying it into SQLi, XSSS, DOS or a normal input.
- Methodology:

  All the requests sent to a server will be stored in access.log file. Using Apache log parser, the log data is split into required format. The following fields: IP address, Date, Time, Method, Requested URL, Status code, number of Bytes received, Source URL and User agent are extracted from the log file and stored in pandas dataframe. Based on the values of different fields and patterns in log data, the log analysis is done to classify the type of request into DOS, Human, Robot, SQLi and XSS request. Based on certain features such as number of requests, source URL and other features, the requests can be classified into DDOS, robot, SQLi, XSS.

- Input: Log file data
- Output: Classification of log data into DOS, Robot, SQLi, XSS.

4.2 Data Flow Diagram Level-0

Figure 4.2 provides the basic overview of the entire system through level-0 data flow concept.

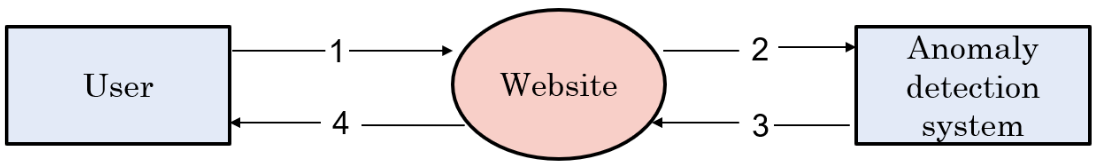


Figure 4.2: Data Flow diagram Level 0

1. User interacts with the website by providing query data into the text fields.
2. Users' query string along with the generated log file data is sent to the anomaly detection system.
3. Anomaly detection system detects for any attack. If there is an attack, it classifies into XSS/ SQLi and alerts the website.
4. Based on the results by anomaly detection system, the user is granted/ revoked further access to website.

4.3 Data Flow Diagram Level-1

Figure 4.3 depicts basic modules in the system and flow of data among various modules through level-1 data flow diagram.

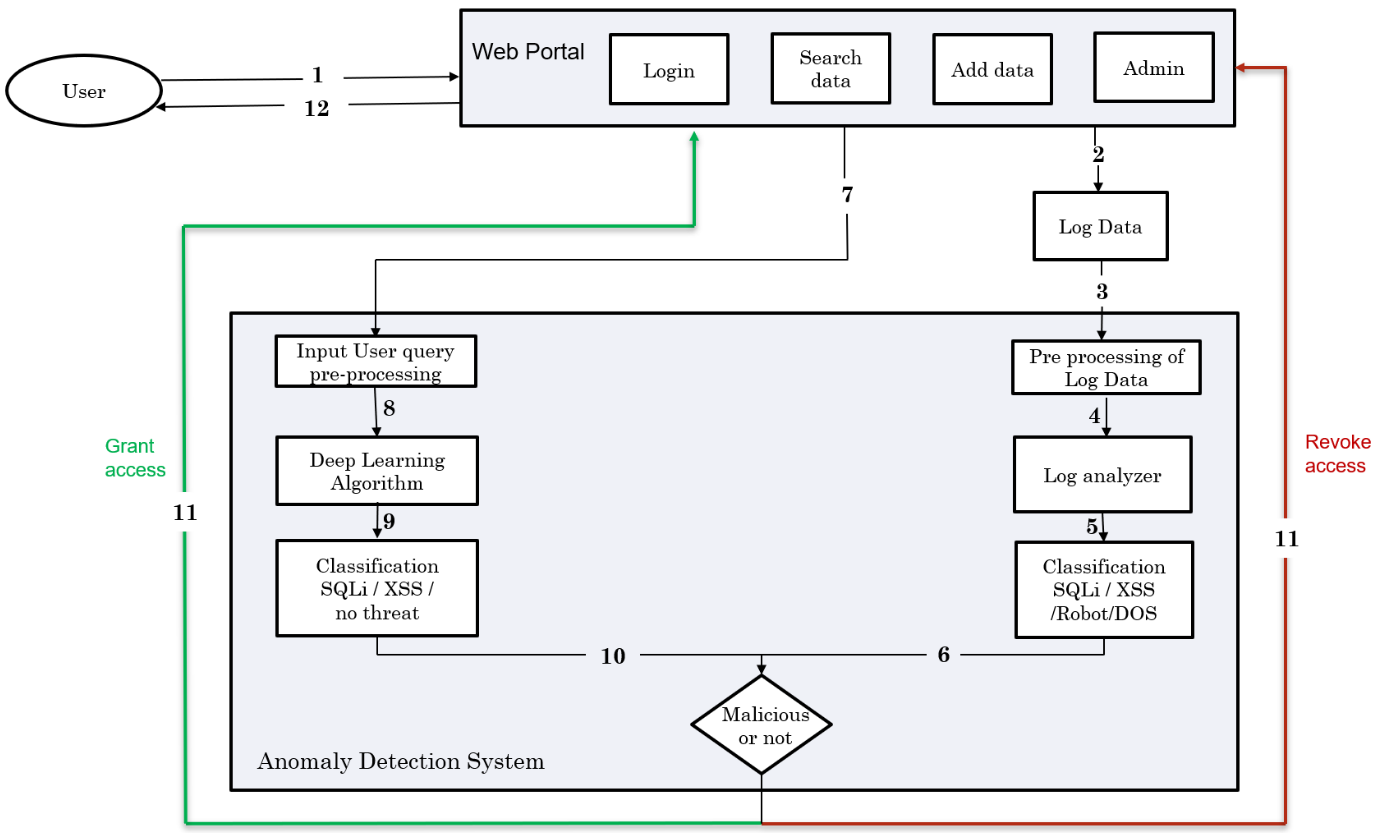


Figure 4.3: Data Flow Diagram Level 1

1. User provides query string to the social media-based web portal through login/search/post fields.
2. The Apache server generated a log data for the corresponding user requests.
3. The generated log data is parsed and sent to the pre-processing stage.
4. The log data is then converted into the required form (pre-processed) and sent to the log analysis module.
5. Based on different attack patterns the requests are classified into XSS/SQLi/Robot/DOS/Benign.
6. Based on the classification, the data is stored into different files for further analysis.
7. The data inserted in the website fields is given to the query string module for pre-processing.

8. The pre-processed data is converted into the required format and given to the deep learning algorithm module.
9. The algorithm classifies the query string as SQLi/XSS attack.
10. The classified results are provided to the security model for further analysis.

11. Based on the results from Query string module and Log module, the user is further granted / revoked with the access.

4.3 Sequence Diagram

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development.

Sequence Diagrams captures:

- the interaction that takes place in a collaboration that either realizes a use case or an operation (instance diagrams or generic diagrams)
- high-level interactions between user of the system and the system, between the system and other systems, or between subsystems (sometimes known as system sequence diagrams)

Purpose of Sequence Diagram

- Model high-level interaction between active objects in a system
- Model the interaction between object instances within a collaboration that realizes a use case
- Model the interaction between objects within a collaboration that realizes an operation
- Either model generic interactions (showing all possible paths through the interaction) or specific instances of a interaction (showing just one path through the interaction)

Figure 4.4 shows object interactions arranged in time sequence.

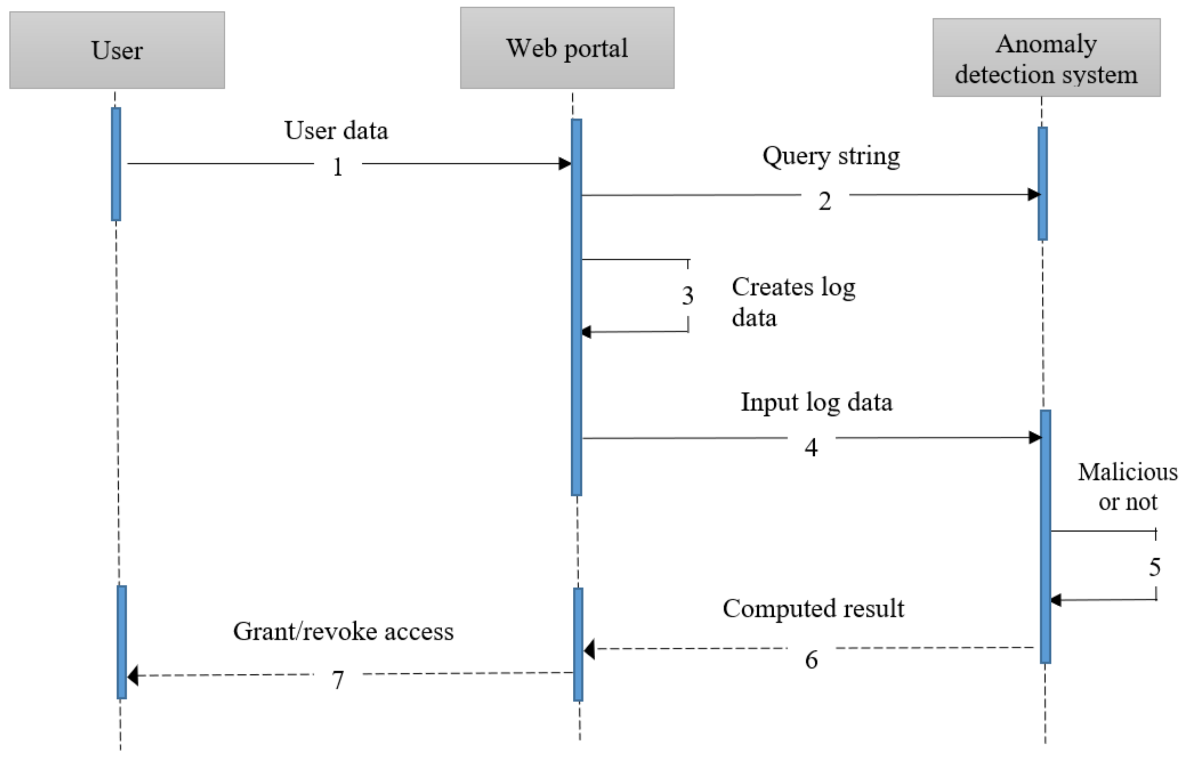


Figure 4.4: Sequence Diagram

1. The user interacts with the Web portal by providing the query string in the desired fields of the developed website.
2. The query string from the website is then sent to the anomaly detection system which houses the deep learning algorithms.
3. Apache server generates the log data for every user request and stores it in access.log file.
4. The log data is parsed and converted into the required format for pre-processing in the anomaly detection system module. Based on different patterns in the log data, log analysis is done.
5. Based on the results of query string module, the data is classified as XSS/SQLi. The log analysis module classifies the requests as XSS/SQLi/Robot/DOS/Benign.
6. The computed results are notified to the Web portal.
7. The user's access to the website is allowed or cancelled.

4.4 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses. The use case diagrams for the interface can be seen in Figure 4.5.

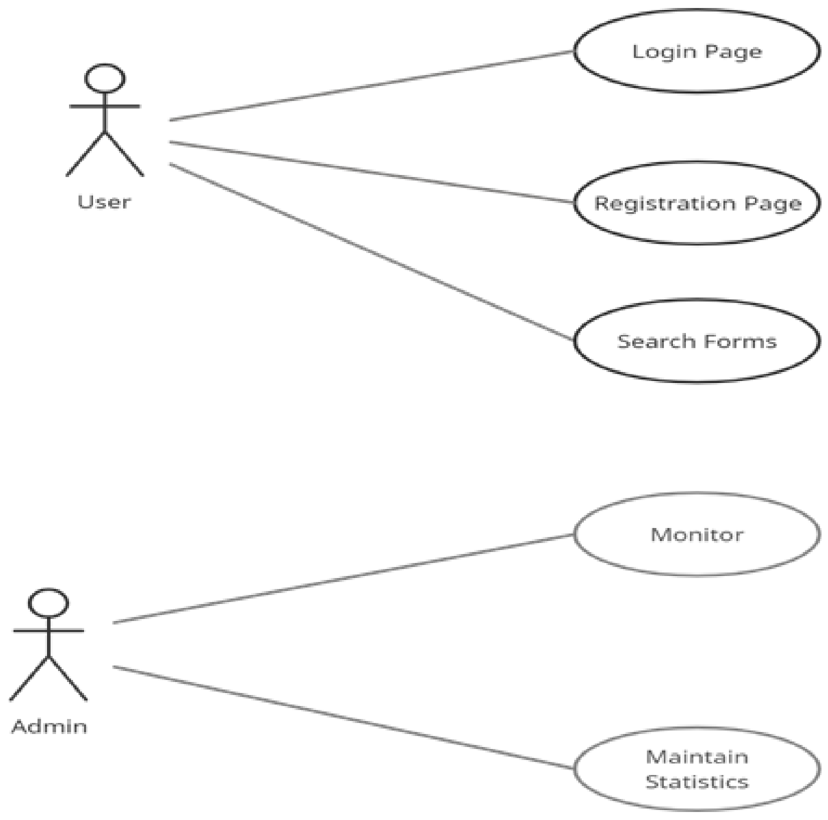


Figure 4.5: Use Case Diagram (Interface 1)

There are two actors namely user and the web portal.

User:

- Step 1: The user visits the webpage and enters data (benign or malicious scripts) in any of the text box in login page or registration page or ‘search details’ page.
- Step 2: The Apache server generates a log for each input given by user and for all the activities of the user and stores this in a log file. The log data and the query string are given as input to the Intrusion detection system to verify if the input

data is benign or malicious. Based on the output of the IDS the user request is sent to the database and the response is sent back to the user or else the user will be blocked.

- Step 3: On successful validation of the user, step 1 is repeated else blocked.

Admin:

- Admin logs' in to the admin portal with his credentials.
- The web portal validates the admin based on the credentials.
- The admin has access to perform log analysis which renders the user statistics on the webpage statistics (Details about number of malicious requests or benign requests).
- The web portal shows details which include the log file details of the corresponding user and either the request is benign or malicious.

CHAPTER 5

IMPLEMENTATION DETAILS

5.1 Domain and Architecture

1) Multilayer Neural Network

A multilayer perceptron (MLP) is a class of feedforward artificial neural network (ANN). The term MLP is used ambiguously, sometimes loosely to any feedforward ANN, sometimes strictly to refer to networks composed of multiple layers of perceptrons (with threshold activation).An MLP consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training. Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable.

Advantages of Multilayer Perceptron:

- Can be applied to complex non-linear problems.
- Works well with large input data.
- Provides quick predictions after training.
- The same accuracy ratio can be achieved even with smaller data.
- Can handle missing values, model complex relationships(like non-linear trends) and support multiple inputs.
- Capable of generalization, they classify an unknown pattern with other known patterns that share the same distinguishing features.
- Ability to derive meaning from complicated or imprecise data.
- It can be used to extract patterns and detect trends that are too complex to be notified by either humans or other computer techniques.
- Ability to separate the filtered data using more complex shapes.

2. Convolutional Neural Network

Convolutional Neural Network (CNN) is revolutionizing several application domains such as visual recognition systems, self-driving cars, medical discoveries, innovative eCommerce and learn to create innovative solutions around image and video analytics to solve complex machine learning and computer vision related problems and implement real-life CNN models. This book starts with an overview of deep neural networks with the example of image classification and walks you through building your first CNN for human face detector and also use concepts like transfer learning with CNN, and Auto-Encoders to build very powerful models, even when not much of supervised training data of labelled images is available.

Later we build upon the learning achieved to build advanced vision related algorithms for object detection, instance segmentation, generative adversarial networks, image captioning, attention mechanisms for vision, and recurrent models for vision. By the end you should be ready to implement advanced, effective and efficient CNN models at your professional project or personal initiatives by working on complex image and video datasets.

Advantages of CNN

- The usages of CNNs are motivated by the fact that they are able to learn relevant features from an image /video at different levels similar to a human brain.
- Another main feature of CNNs is weight sharing. Let’s take an example to explain this. Say you have a one layered CNN with 10 filters of size 5x5. Now you can simply calculate parameters of such a CNN, it would be 5*5*10 weights and 10 biases i.e 5* 5*10 + 10 = 260 parameters. Now let’s take a simple one layered NN with 250 neurons, here the number of weight parameters depending on the size of images is ‘250 x K’ where size of the image is P X M and K = (P *M). Additionally, you need ‘M’ biases. For the MNIST data as input to such a NN we will have (250*784+1 = 19601) parameters. Clearly, CNN is more efficient in terms of memory and complexity. Imagine NNs and CNNs with billions of neurons, then CNNs would be less complex and saves memory compared to the NN.
- In terms of performance, CNNs outperform NNs on conventional image recognition tasks and many other tasks. Look at the Inception model, Resnet50 and many others for instance.
- For a completely new problem CNNs are very good feature extractors. This means that you can extract useful attributes from an already trained CNN

with its trained weights by feeding your data on each level and tune the CNN a bit for the specific task. Eg: Add a classifier after the last layer with labels specific to the task. This is also called pre-training and CNNs are very efficient in such tasks compared to NNs. Another advantage of this pre-training is we avoid training of CNN and save memory, time. The only thing you have to train is the classifier at the end for your labels.

5.2 Requirements Planning

5.2.1 Tools

1) Anaconda Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text. Its uses include data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more. The best method of installing the Jupyter Notebooks is by the installation of the Anaconda package. A notebook kernel is a “computational engine” that executes the code contained in a Notebook document. Markdowns are a way to make Jupyter Notebooks more presentable and help to make the viewers or readers a more concise understanding of the code. Users can add headers, images, widgets, bullet points, ordered lists, hyperlinks, and so much more.

2) VS Code

The Visual Studio integrated development environment is a creative launching pad that you can use to edit, debug, and build code, and then publish an app. An integrated development environment (IDE) is a feature-rich program that can be used for many aspects of software development. Over and above the standard editor and debugger that most IDEs provide, Visual Studio includes compilers, code completion tools, graphical designers, and many more features to ease the software development process. Visual Studio offers a suite of tools that enable you to easily create cloud-enabled applications powered by Microsoft Azure which can be used to configure, build, debug, package, and deploy applications and services on Microsoft Azure directly from the IDE.

5.2.2 Modules and Libraries

- Log parser:

  Logparser provides a toolkit and benchmarks for automated log parsing, which is a crucial step towards structured log analytics. By applying logparser, users can automatically learn event templates from unstructured logs and convert raw log messages into a sequence of structured events. The logparser module is included in log.py file to segregate all the fields available in a log file and formats into pandas dataframe.

- Numpy:

  NumPy is a Python library used for working with arrays, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. Numpy is used to contain input matrix for Convoluted Neural Network.

- Pandas:

  Pandas is an open source Python package that is most widely used for data science/data analysis and machine learning tasks. Pandas DataFrame is two-dimensional size-mutable, potentially heterogeneous tabular data structure with labelled axes (rows and columns).In the real world, a Pandas DataFrame will be created by loading the datasets from existing storage, storage can be SQL Database, CSV file, and Excel file.

- Keras:

  Keras is a powerful and easy-to-use free open source Python library for developing and evaluating deep learning models. It wraps the efficient numerical computation libraries Theano and TensorFlow and allows you to define and train neural network models in just a few lines of code. It can be used to construct deep learning models.

- Tensorflow:

  TensorFlow is a Python library for fast numerical computing created and released by Google. It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of Tensor Flow. Tensor Flow is mainly used for: Classification, Perception, Understanding, Discovering, Prediction and Creation.

- Matplotlib:

  Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, Python, or GTK. It is used to plot the pie plot to provide statistics on user data.

- Count Vectorizer:

  CountVectorizer is a great tool provided by the scikit-learn library in Python. It is used to transform a given text into a vector on the basis of the frequency (count) of each word that occurs in the entire text. The value of each cell is nothing but the count of the word in that particular text sample.

- Pickle:

  Python pickle module is used for serializing and de-serializing a Python object structure. Any object in Python can be pickled so that it can be saved on disk. What pickle does is that it “serializes” the object first before writing it to file. Pickling is a way to convert a python object (list, dict, etc.) into a character stream. It is used to store the vocabulary created by the count vectorizer.

5.3 Programming Resources

5.3.1 Sample Code for Module-1

```
import sys

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.feature_extraction.text import CountVectorizer

import sklearn

from keras.models import model_from_json

import pickle
```

```
def load_csv():
    global sqlidf
    contents=[]
    with open("sqli.csv",'r',encoding = 'utf-16') as f:
        for line in f:
            word=line.split('\n')
            list2 = [x for x in word if x]
            list1 = list2[0].rsplit(',',maxsplit=1)
            sentence=list1[0][1:]
            label=list1[1][:-1]
            listx=[sentence,label]
            contents += [listx]
    contents=contents[1:]
    sqlidf = pd.DataFrame(contents,columns=['Sentence','Label'])
load_csv()


sqlidf['Sentence'] = sqlidf['Sentence'].astype(str)
sqlidf['Label']=sqlidf['Label'].astype(int)
X=sqlidf['Sentence']
y=sqlidf['Label']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)


# load json and create model
json_file = open('model.json', 'r')
```

```
loaded_model_json = json_file.read()

json_file.close()

loaded_model = model_from_json(loaded_model_json)

# load weights into new model

loaded_model.load_weights("model.h5")



def testing1(querystring):

    instance = X_test

    instance.iloc[0] = querystring[0]

    vocabulary_to_load = pickle.load(open("dictionary.pickle", 'rb'))

    loaded_vectorizer =
sklearn.feature_extraction.text.CountVectorizer(vocabulary=vocabulary_to_load)

    loaded_vectorizer._validate_vocabulary()

    instance_posts = loaded_vectorizer.transform(instance).toarray()

    pred = loaded_model.predict(instance_posts)

    if pred[0]>0.5:

        res=1

    else:

        res=0

    return res

query=sys.argv[1]

print(testing1([query]))
```

5.3.2 Sample Code for Module-2

```
#!/usr/bin/env python

# coding: utf-8


from tensorflow.keras import models

from apachelogs import LogParser

import pandas as pd

import sys

import numpy as np

parser = LogParser("%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-
Agent}i\"")


import pandas as pd

global f

f=pd.DataFrame(columns=['IP_Address','Date','Time','Method','Req_url','Status_code','By
tes_recieved','Source_url','User_Agent'])

data={}


def append_df_to_excel(df, excel_path):

    try:

        df_excel = pd.read_excel(excel_path)

        result = pd.concat([df_excel, df], ignore_index=True)

        result.to_excel(excel_path, index=False,engine='xlsxwriter')

    except:
        df.to_excel(excel_path,index=False,engine='xlsxwriter')


def df_to_excel_new(val, excel_path):
    if len(val) ==3:
        try:
            l = pd.read_excel(excel_path)
            abc = np.array(l.iloc[0])
            abc[0]+= val[0]
            abc[1]+= val[1]
            abc[2]+=val[2]
            l.iloc[0] = abc
            l.to_excel(excel_path,index=False,engine='xlsxwriter')
        except:
            l = pd.DataFrame(columns=['Total','Bengin','DDOS'])
            l.loc[0] = val
            l.to_excel(excel_path,index=False,engine='xlsxwriter')


    if len(val)==5:
        try:
            l = pd.read_excel(excel_path)
            abc = np.array(l.iloc[0])
            abc[0]+= val[0]
            abc[1]+= val[1]
            abc[2]+= val[2]
            abc[3]+= val[3]

            abc[4]+=val[4]

            l.iloc[0] = abc

            l.to_excel(excel_path,index=False,engine='xlsxwriter')

        except:

            l = pd.DataFrame(columns=['Total','Bengin','Robot','SQLi','XSS'])

            l.loc[0]=val

            l.to_excel(excel_path,index=False,engine='xlsxwriter')


try:

    file1 = open("MyFile.txt","r")

    i1=int(file1.readline())

    i1+=1

    file1.close()

except:

    i1=0




with open('D:/Xampp/apache/logs/access.log') as fp:  # doctest: +SKIP

    # count=i1

    data={}

    for i,entry in enumerate(parser.parse_lines(fp)):

        if (i>=i1):
            print(i)
            ob=entry.request_time
            date = ob.strftime("%d-%b-%Y")
            time=ob.strftime("%H:%M:%S")
            try:
                method=entry.request_line.split(" ")[0]
                url=entry.request_line.split(" ")[1]
            except:
                method=None
                url=entry.request_line
    #          code=entry.final_status
            data['IP_Address']=entry.remote_host
            data['Date']=str(date)
            data['Time']=str(time)
            data['Method']=method
            data['Req_url']=url
            data['Status_code']=entry.final_status
            data['Bytes_recieved']=entry.bytes_sent
            data['Source_url']=entry.headers_in['Referer']
            data['User_Agent']=entry.headers_in['User-Agent']
            f = f.append(data,ignore_index=True)
        else:
            continue
###print(entry.remote_host,date,time,entry.request_line,entry.final_status,entry.bytes_sen
t,entry.headers_in['Referer'],entry.headers_in['User-Agent'])
```

```
        #count+=1
####print(f)
if f.empty:
    sys.exit()



filew = open("MyFile.txt","w+")
filew.write(str(i))
filew.close()
# f.to_excel('data.xlsx',index=False ,engine='xlsxwriter')
append_df_to_excel(f,"data.xlsx")
# df = pd.DataFrame(pd.read_excel("data.xlsx"))
df=f
df=df.sort_values(by=['IP_Address','Date','Time'])
##print(df)
p=pd.DataFrame()
time=0
ip=0
ua=0
source=0
dos=[]
# df = pd.DataFrame(pd.read_excel("sorted.xlsx"))
df = df.fillna('-')
p=df[df['User_Agent']=='-']
```

```
####print(p)

nosource=pd.DataFrame()

nosource=p[p['Source_url']=='-']

nosource=p[p['Status_code']==400]

##print(nosource)

ddos=pd.DataFrame(columns=['IP_Address','Date','Time','Num_Req'])

name={}

count=0

if len(nosource)==0:

    print("No DDOS attack found")

else:

    pre=nosource.values[0]

    ##print(pre)

    num=0

    for val in nosource.values:

        num+=1



        if pre[0]==val[0] and pre[1]==val[1] and pre[2]==val[2]:

            count+=1



        else:

            name['IP_Address']=pre[0]

            name['Date']=pre[1]

            name['Time']=pre[2]
            name['Num_Req']=count

            ###print(pre[2],count)

            ddos=ddos.append(name,ignore_index=True)

            count=1

            pre=val

    name['IP_Address']=pre[0]

    name['Date']=pre[1]

    name['Time']=pre[2]

    name['Num_Req']=count

    ddos=ddos.append(name,ignore_index=True)

   ###print(num)

   ###print(ddos)


   ###print(len(ddos))

    append_df_to_excel(ddos, "ddos.xlsx")

    # ddos.to_excel('ddos.xlsx',index=False )

# append_df_to_excel(ddos, "ddos.xlsx")

inject=pd.DataFrame()

inject= df.drop_duplicates().merge(nosource.drop_duplicates(),
on=nosource.columns.to_list(),

                   how='left', indicator=True)

inject.loc[inject._merge=='left_only',inject.columns!='_merge']

df1=pd.DataFrame()

df1 = pd.merge(df,nosource, how='outer', indicator=True)
```

```
Output=df1.loc[df1._merge == 'left_only']

####print(df1)

##print(len(Output))

del Output['_merge']

# for i in Output.values:

#    ###print(i)


##print(Output)

# Output.to_excel('output.xlsx')

other=Output[Output.Req_url.str.contains('^/final_codes')]

##print(other)

df2=pd.DataFrame()

df2 = pd.merge(Output,other, how='outer', indicator=True)


tool=df2.loc[df2._merge == 'left_only']

####print(df1)

del tool['_merge']

# for i in Output.values:

#    ###print(i)

##print(tool)

#tool.to_excel('owsap.xlsx')

append_df_to_excel(tool, "owasp.xlsx")

attack=Output[Output.Req_url.str.contains('^/final_codes.*\.php.*')]

##print(attack)
```

```
##print(len(attack))

sql=attack[attack.User_Agent.str.contains('sql')]

##print(sql)

# sql.to_excel('sqlattack.xlsx',index=False)

append_df_to_excel(sql, "sqlattack.xlsx")

df3=pd.DataFrame()

df3 = pd.merge(attack,sql, how='outer', indicator=True)


rob1=df3.loc[df3._merge == 'left_only']

####print(df1)

##print(len(rob1))

del rob1['_merge']

# for i in Output.values:

#    ###print(i)

##print(rob1)

# tool.to_excel('owsap.xlsx')

blank=rob1[rob1['Source_url']=='-']

main=blank[blank.Req_url.str.contains('inject.php')]

dele= pd.merge(blank,main, how='outer', indicator=True)

blank=dele.loc[dele._merge == 'left_only']

# ###print(df1)

###print(len(rob1))

del blank['_merge']

a1= pd.merge(rob1,blank, how='outer', indicator=True)
```

```
actual=a1.loc[a1._merge == 'left_only']

# ###print(df1)

###print(len(rob1))

del actual['_merge']

# ###print(actual)

ex=actual

ex=ex.sort_values(by=['IP_Address','Date','Time'])

human = ex.drop_duplicates(subset = ["IP_Address","Date","Time"])

df4=pd.DataFrame()

df4 = pd.merge(ex,human, how='outer', indicator=True)


robo=df4.loc[df4._merge == 'left_only']

# ###print(df1)


del robo['_merge']


# ###print(human)

###print(robo)

con=[human,robo]

human=pd.concat(con)

human=human.sort_values(by=['IP_Address','Date','Time'])

sqltool=[sql,blank]

sqltool=pd.concat(sqltool)

sqltool=sqltool.sort_values(by=['IP_Address','Date','Time'])
```

```
##print(sqltool)

append_df_to_excel(sqltool, "robot.xlsx")

append_df_to_excel(human, "human.xlsx")

##print(human)


from urllib.parse import unquote

import urllib.parse

import logsql as f1

sqlattack1=human[human.Req_url.str.contains('sqlver.php')]


sqli= pd.DataFrame()

lst=[]

for  i in sqlattack1.values:

    file=i[4]

    try:

        first=file.split("=")[1]

        first=first.split("&")[0]

        first= urllib.parse.unquote(first)

        two=file.split("=")[2]

        two= unquote(two)

        a=f1.testing1([first])

        b=f1.testing1([two])


        if a ==1 or b ==1:
            lst+=[i]



    except:
        print("No Get Methods")


# In[27]:
sqlattack2=human[human.Req_url.str.contains('user_search_disp.php')]
##print(sqlattack2)
for i in sqlattack2.values:
    # print(i)
    file=i[4]
    data=file.split("=")[1]
    data= unquote(data)
    d = f1.testing1([data])
    if d == 1:
        lst+=[i]
sqli=sqli.append(lst)
append_df_to_excel(sqli,'Newsqli.xlsx')
# print(sqli)
from log1xss import test as f2
xsslst=[]
xss1=human[human.Req_url.str.contains('xssver.php')]
# print(xss1)
```

```
xss2= pd.DataFrame()
for i in xss1.values:
    file=i[4]
    xss=file.split("=")[1]
    xss= unquote(xss)
    # print(xss)
    c = f2(xss)
        if c ==1 :
        xsslst+=[i]
    # print(xss)
xss2=xss2.append(xsslst)
append_df_to_excel(xss2,'Newxss.xlsx')


# f.drop_duplicates()
tot=len(f)
print("total",tot)
dd=len(nosource)
print(dd)
ben =tot - dd
dos={}
dos['Total']=tot
dos['Bengin'] = tot - dd
dos['DDOS']=dd
print(dos)
```

```
lst1 =[tot,ben,dd]

print("list1",lst1)

df_to_excel_new(lst1,"adminddos.xlsx")


att={}

a_total=len(attack)

a_rob=len(sqltool)

a_sql=len(sqli)

a_xss=len(xss2)

ben1= a_total-a_rob-a_sql-a_xss

att['Total']=a_total

att['Bengin']= ben1

att['Robot']=a_rob

att['SQLi']=a_sql

att['XSS']=a_xss

lst=[a_total,ben1,a_rob,a_sql,a_xss]

print(a_total,ben1,a_rob,a_sql,a_xss)

print(att)


df_to_excel_new(lst,"admindata.xlsx")
```

CHAPTER 6

PROJECT GANTT CHART

The below Table 6.1 shows the Gantt Chart of the project Flow.

Task Name	Oct 2nd week	Oct 2nd week-4th week	Nov 1st week-4th week	Dec 1st week- 4th week	Jan 1st week-4th week	March1st week-4th week	April 1st week-4th week	May 1st week – June 4th week	July 3rd week
Identification of problem statement	🟧								
Approval of title and project Synopsys		🟧							
Literature survey			🟧	🟧					
Design					🟧				
Coding and Implementation						🟧	🟧	🟧	
Testing and debugging							🟧	🟧	
Reporting and documentation									🟧

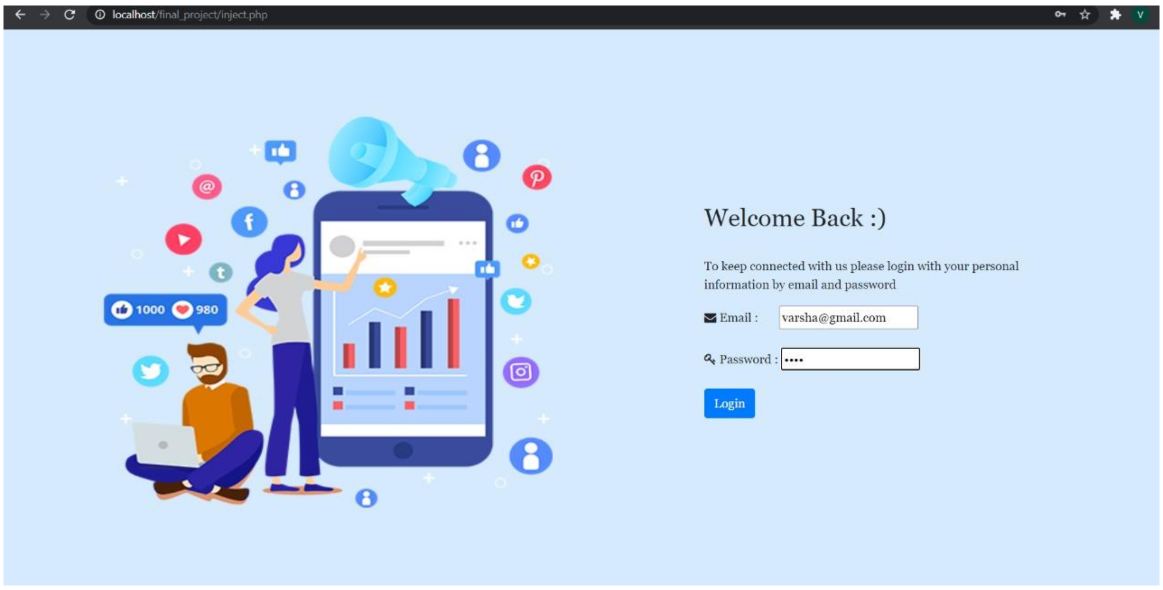
Table 6.1: Gantt chart

CHAPTER 7

EXPERIMENTAL RESULTS

7.1 Module-1 Snapshots

Figure 7.1 is the login page for the existing user



Welcome Back :)

To keep connected with us please login with your personal information by email and password

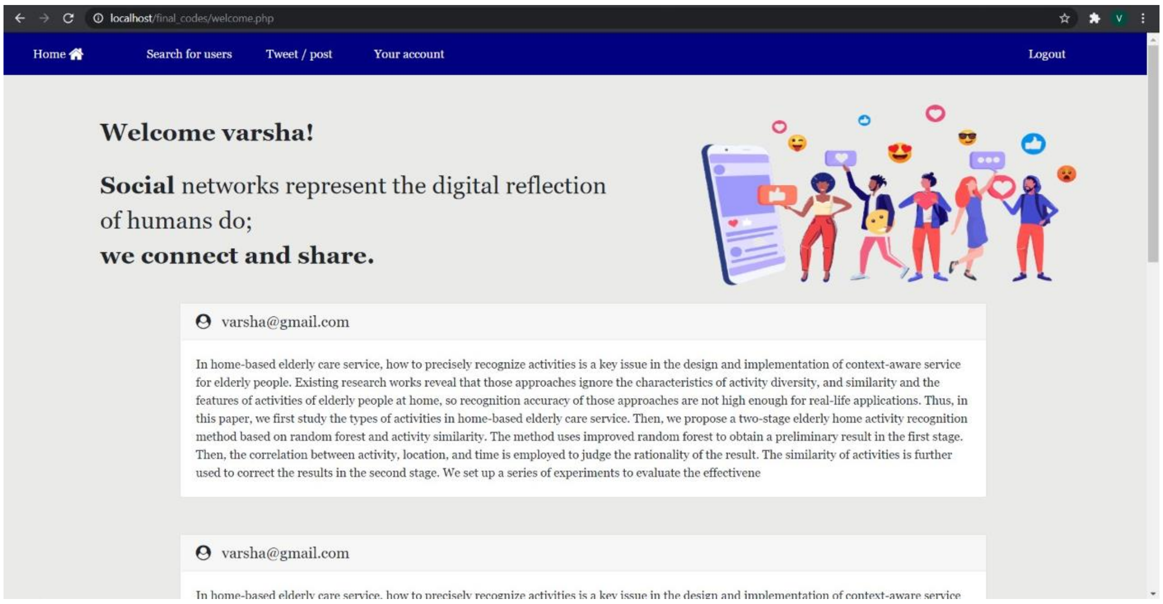
Email : varsha@gmail.com

Password : ••••

Login

Figure 7.1: Login page

Figure 7.2 is the welcome page for the successful login of the user



Welcome varsha!

Social networks represent the digital reflection of humans do;
we connect and share.

varsha@gmail.com

In home-based elderly care service, how to precisely recognize activities is a key issue in the design and implementation of context-aware service for elderly people. Existing research works reveal that those approaches ignore the characteristics of activity diversity, and similarity and the features of activities of elderly people at home, so recognition accuracy of those approaches are not high enough for real-life applications. Thus, in this paper, we first study the types of activities in home-based elderly care service. Then, we propose a two-stage elderly home activity recognition method based on random forest and activity similarity. The method uses improved random forest to obtain a preliminary result in the first stage. Then, the correlation between activity, location, and time is employed to judge the rationality of the result. The similarity of activities is further used to correct the results in the second stage. We set up a series of experiments to evaluate the effectivene

varsha@gmail.com

In home-based elderly care service, how to precisely recognize activities is a key issue in the design and implementation of context-aware service

Figure 7.2: Successful login

Figure 7.3 shows an attempt to SQLi attack in the search field of the web page

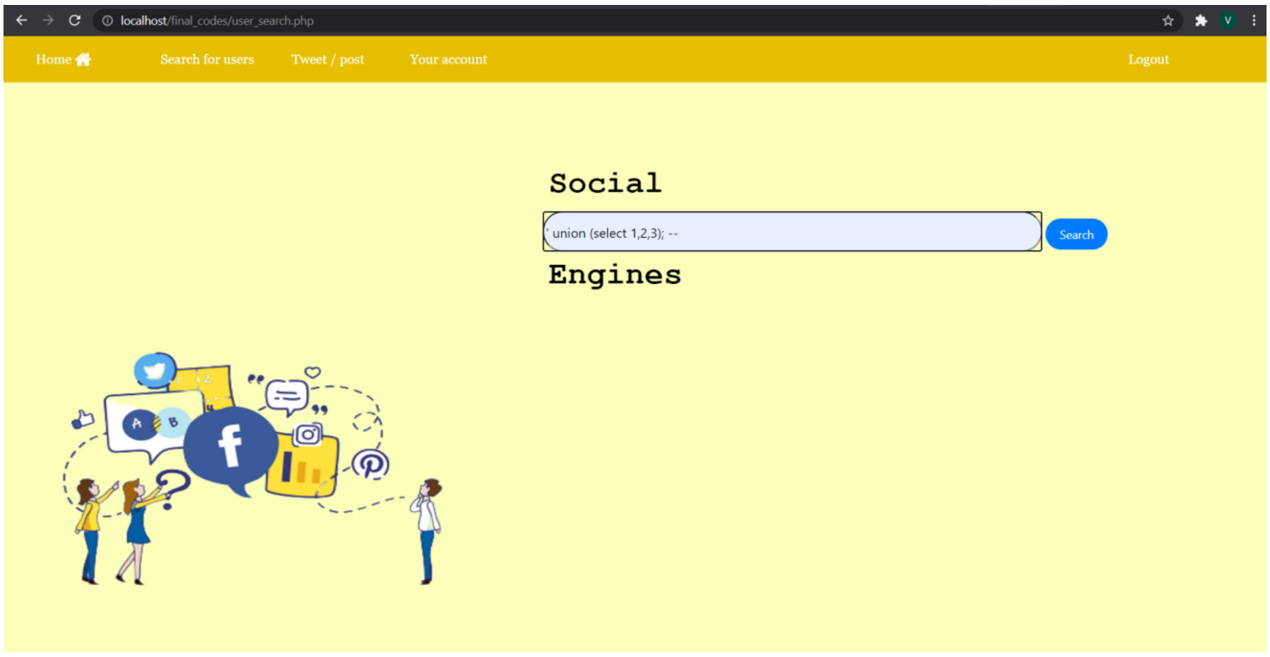
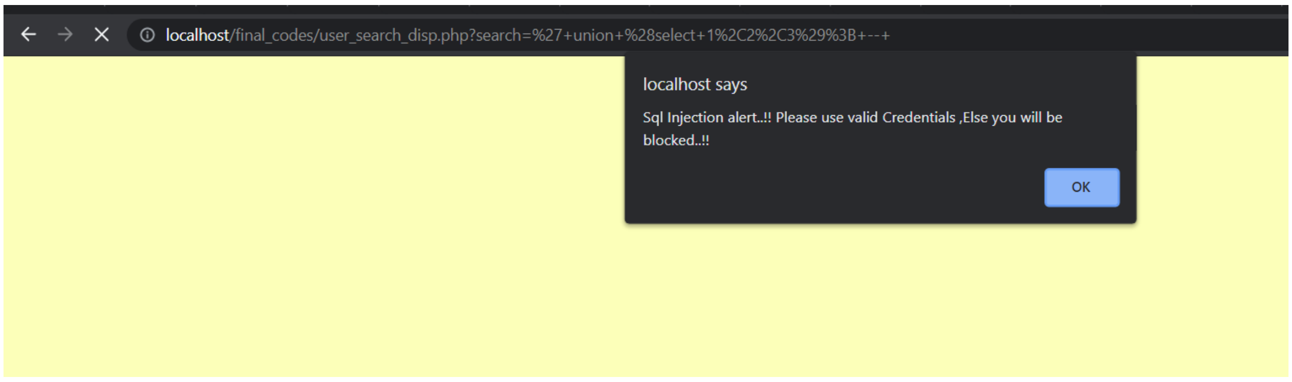


Figure 7.3: Search engine page

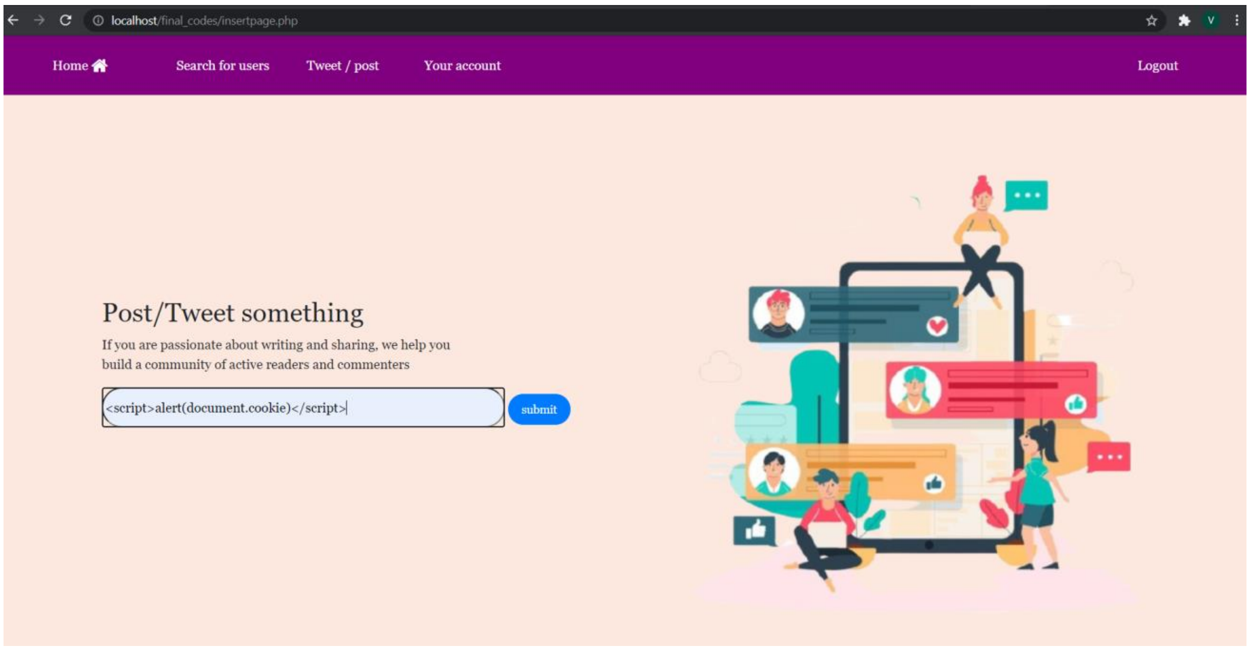
Figure 7.4 is the alert notification for the SQLi attack attempt


localhost/final_codes/user_search_disp.php?search=%27+union+%28select+1%2C2%2C3%29%3B+--+
localhost says
OK

Sql Injection alert..!! Please use valid Credentials ,Else you will be blocked..!!

Figure 7.4: Alert of SQLi attack

Figure.7.5 shows an attempt to XSS attack in the post/tweet field of the web page



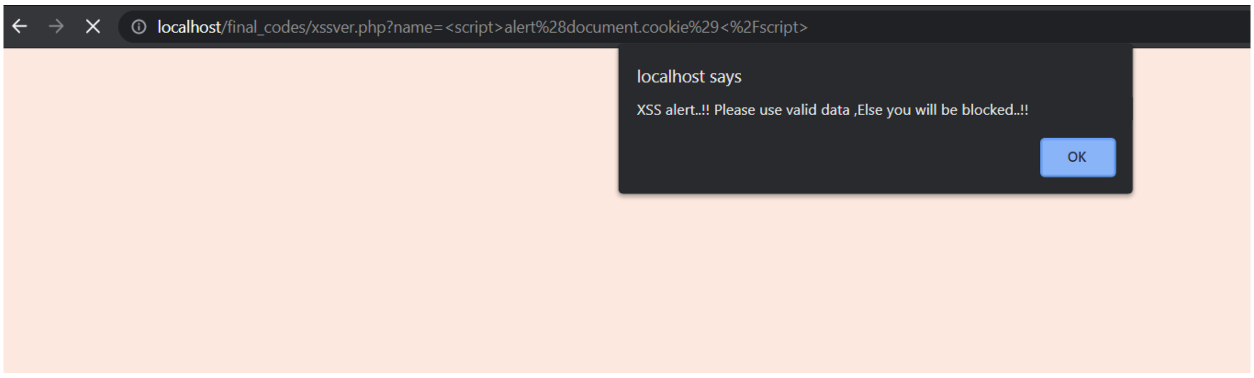
Post/Tweet something

If you are passionate about writing and sharing, we help you build a community of active readers and commenters

```
<script>alert(document.cookie)</script>
```

Figure 7.5: XSS script as input

Figure 7.6 is the alert notification for the XSS attack attempt



```
localhost/final_codes/xssver.php?name=<script>alert%28document.cookie%29<%2Fscript>
```

localhost says

XSS alert..!! Please use valid data ,Else you will be blocked..!!

Figure 7.6: Alert for XSS attack

Figure 7.7 is the Registration page for users

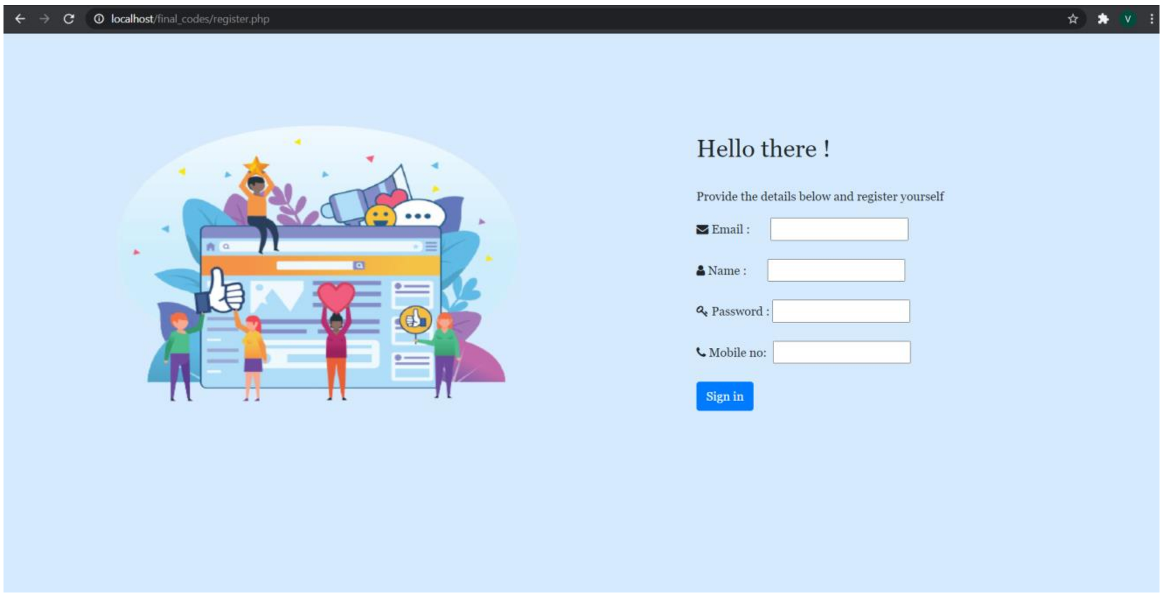
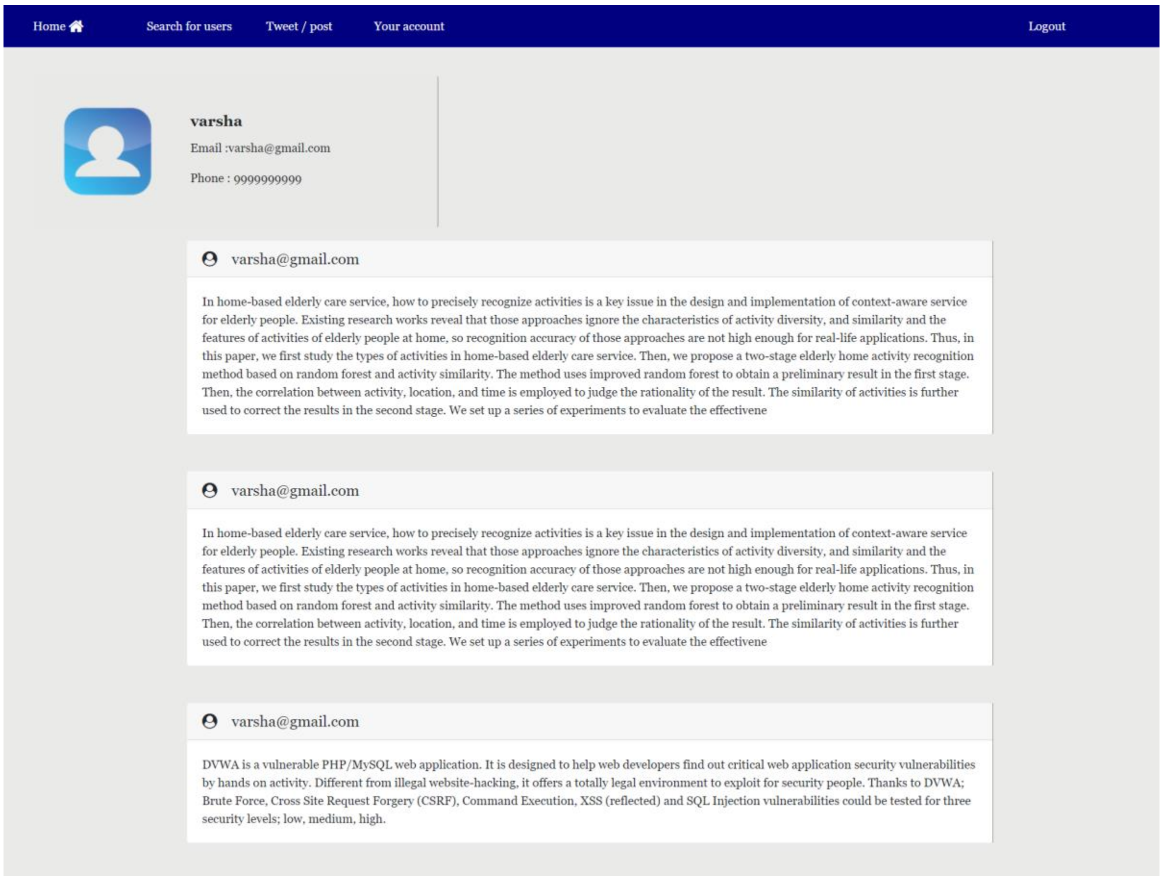

localhost/final_codes/register.php
Hello there !
Provide the details below and register yourself
Email :
Name :
Password :
Mobile no:
Sign in

Figure 7.7: User registration page

Figure 7.8 is Profile page of the user



In home-based elderly care service, how to precisely recognize activities is a key issue in the design and implementation of context-aware service for elderly people. Existing research works reveal that those approaches ignore the characteristics of activity diversity, and similarity and the features of activities of elderly people at home, so recognition accuracy of those approaches are not high enough for real-life applications. Thus, in this paper, we first study the types of activities in home-based elderly care service. Then, we propose a two-stage elderly home activity recognition method based on random forest and activity similarity. The method uses improved random forest to obtain a preliminary result in the first stage. Then, the correlation between activity, location, and time is employed to judge the rationality of the result. The similarity of activities is further used to correct the results in the second stage. We set up a series of experiments to evaluate the effectivene

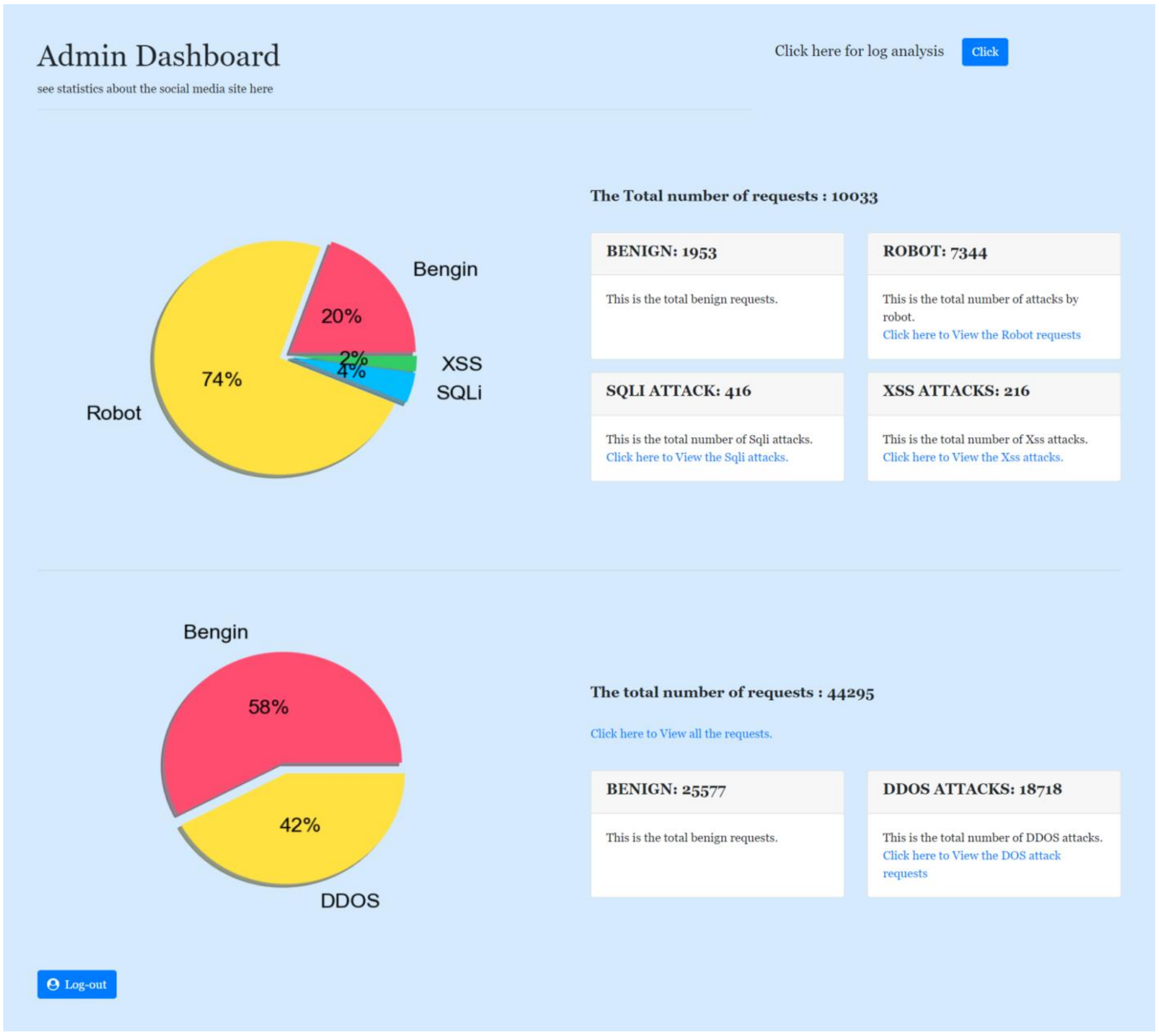
In home-based elderly care service, how to precisely recognize activities is a key issue in the design and implementation of context-aware service for elderly people. Existing research works reveal that those approaches ignore the characteristics of activity diversity, and similarity and the features of activities of elderly people at home, so recognition accuracy of those approaches are not high enough for real-life applications. Thus, in this paper, we first study the types of activities in home-based elderly care service. Then, we propose a two-stage elderly home activity recognition method based on random forest and activity similarity. The method uses improved random forest to obtain a preliminary result in the first stage. Then, the correlation between activity, location, and time is employed to judge the rationality of the result. The similarity of activities is further used to correct the results in the second stage. We set up a series of experiments to evaluate the effectivene

DVWA is a vulnerable PHP/MySQL web application. It is designed to help web developers find out critical web application security vulnerabilities by hands on activity. Different from illegal website-hacking, it offers a totally legal environment to exploit for security people. Thanks to DVWA; Brute Force, Cross Site Request Forgery (CSRF), Command Execution, XSS (reflected) and SQL Injection vulnerabilities could be tested for three security levels; low, medium, high.

Figure 7.8: User profile page

7.2 Module-2 Snapshots

Figure.7.9 shows the attack statistics that will be rendered after performing the log analysis in the admin portal.



Admin Dashboard

see statistics about the social media site here

Click here for log analysis Click

The Total number of requests : 10033

BENIGN: 1953

This is the total benign requests.

ROBOT: 7344

This is the total number of attacks by robot.
Click here to View the Robot requests

SQLI ATTACK: 416

This is the total number of Sqli attacks.
Click here to View the Sqli attacks.

XSS ATTACKS: 216

This is the total number of Xss attacks.
Click here to View the Xss attacks.

The total number of requests : 44295

Click here to View all the requests.

BENIGN: 25577

This is the total benign requests.

DDOS ATTACKS: 18718

This is the total number of DDOS attacks.
Click here to View the DOS attack requests

Log-out

Figure 7.9 : Data computed after log analysis

CHAPTER 8

TESTING

8.1 Software Testing

Software Testing is a process of evaluating the functionality of a software application to find any software bugs. It checks whether the developed software met the specified requirements and identifies any defect in the software in order to produce a quality product. It is basically executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements. Software testing is now a very significant and integral part of software development. Ideally, it is best to introduce software testing in every phase of software development life cycle. Actually, a majority of software development time is now spent on testing. It is also stated as the process of verifying and validating a software product. It checks whether the software product:

- Meets the business and technical requirements that guided its design and development
- Works as per the requirement
- Can be implemented with the same characteristics

8.1.1 Principles of Software Testing

Testing of software is exceptionally imaginative and an intellectual task for testers to perform. Testing of software or applications consist of some principles that play a significant role for a software tester while testing the project.

The Principles of Software Testing are as follows:

a) **Software testing can help in detecting bugs**: Testing any software or project can help in revealing a few or some defects that may or may not be detected by developers. However, testing of software alone cannot confirm that your developed project or software is error free. Hence, it’s essential to devise test cases and find out as many defects as possible.

b) **Testing with effectiveness is impossible**: Until your project or application under test has a straightforward structure having limited input, it won't be likely or achievable to check and test all feasible sets of data, modules, and scenarios.

c) **Early testing**: The earlier you will begin to test your project or software the better you will find to utilize your existing time.

d) **Defect in clustering**: At the time of testing, you can observe that majority of the defects or bugs reported are because of a small number of modules inside your software or system.

e) **Software testing is context-dependent**: Various methods, procedures, and kinds of testing are there which defines the type and characteristics of the application. For example, an application related to health device needs more testing and doctor-based feedbacks than a game or small software.

f) **Error free or Bug-free software is a myth**: Just because when a tester tested an application and didn't detect any defects in that project, doesn't indicate or imply that your software is ready for shipping.

8.2 TYPES OF TESTING

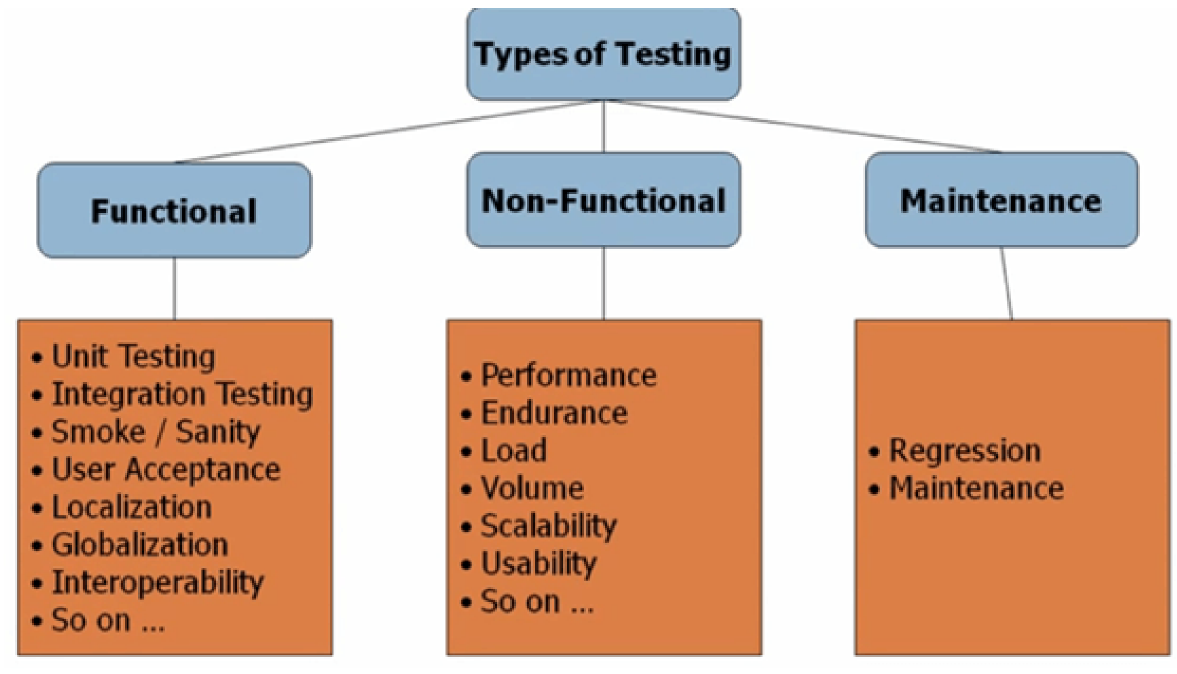


Figure 8.1: Types of Testing

a) **UNIT TESTING:** Testing of an individual software component or module is termed as Unit Testing. It is typically done by the programmer and not by testers, as it requires detailed knowledge of the internal program design and code. It may also require developing test driver modules or test harnesses.

b) **USABILITY TESTING:** Under Usability Testing, User-friendliness check is done. The application flow is tested to know if a new user can understand the application easily or not, Proper help documented if a user gets stuck at any point. Basically, system navigation is checked in this testing.

c) **INTEGRATION TESTING:** Testing of all integrated modules to verify the combined functionality after integration is termed as Integration Testing. Modules are typically code modules, individual applications, client and server applications on a network, etc. This type of testing is especially relevant to client/server and distributed systems.

d) **SYSTEM TESTING:** Under System Testing technique, the entire system is tested as per the requirements. It is a Black-box type Testing that is based on overall requirement specifications and covers all the combined parts of a system.

e) **REGRESSION TESTING:** Testing an application as a whole for the modification in any module or functionality is termed as Regression Testing. It is difficult to cover all the system in Regression Testing, so typically Automation Testing Tools are used for these types of testing.

f) **RECOVERY TESTING:** It is a type of testing which validates how well the application or system recovers from crashes or disasters. Recovery Testing determines if the system is able to continue the operation after a disaster.

g) **PERFORMANCE TESTING:** This term is often used interchangeably with ‘stress' and ‘load' testing. Performance Testing is done to check whether the system meets the performance requirements. Different performance and load tools are used to do this testing.

h) **SECURITY TESTING:** It is a type of testing performed by a special team of testers. A system can be penetrated by any hacking way. Security Testing is done to check how the software or application or website is secure from internal and external threats. This testing includes how much software is secure from the malicious program, viruses and how secure and strong the authorization and authentication processes are. It also checks how software behaves for any hackers attack and malicious programs and how software is maintained for data security after such a hacker attack.

i) **LOAD TESTING**: It is a type of Non-Functional Testing and the objective of Load Testing is to check how much load or maximum workload a system can handle without any performance degradation. Load Testing helps to find the maximum capacity of the system under specific load and any issues that cause software performance degradation. Load testing is performed using tools like JMeter, LoadRunner, Web Load, Silk performer, etc.

8.3 Test Cases

8.3.1 Module-1

Table 8.1: Test cases for module 1

Input	Expected result	Actual result	Correctness
User login with SQLi attack in username field	Alert notification with access revoked	Alert notification with access revoked	Yes
User login with SQLi attack in password field	Alert notification with access revoked	Alert notification with access revoked	Yes
User login with SQLi attack in both the fields	Alert notification with access revoked	Alert notification with access revoked	Yes
User login with proper credentials	Access granted to further pages	Access granted to further pages	Yes
User search with SQLi attack	Alert notification with access revoked	Alert notification with access revoked	Yes
Valid user search	Search results displayed	Search results displayed	Yes

Post articles with XSS attack	Alert notification with access revoked	Alert notification with access revoked	Yes
Post articles with valid details	Access to posted page	Access to posted page	Yes

8.3.2 Module-2

Table 8.2: Test cases for module 2

Input	Expected result	Actual result	correctness
Attack requests from DOS tools	Classification as DOS attack	Classification as DOS attack	Yes
Attack from robot	Classification as Robot attack	Classification as Robot attack	Yes
SQLi attack requests	Classification as SQLi attack	Classification as SQLi attack	Yes
XSS attack requests	Classification as XSS attack	Classification as XSS attack	Yes
Benign requests	Classification as Benign requests	Classification as Benign requests	Yes

CHAPTER 9

CONCLUSION

Cyber-crime is constantly on the rise, and many smaller businesses are extremely vulnerable as a result of ineffective cyber security due to enormous usage of internet. The impact of cyber-crime can range from financial losses to business losses, due to which different industries can also suffer many disastrous consequences as a result of criminal cyber-attacks. SQL injection can leave the application at a high-risk of compromise resulting in a threat to the integrity, and confidentiality of data as well as authorization and authentication aspects of the application.

A successful attack may end in the unauthorized viewing of user lists, the deletion of tables and, in certain cases, the attacker gaining admin rights to a database, all of which are highly deleterious to a business. The effect of XSS attack ranges from user's Session Hijacking to disclosure of sensitive data, Cross site request forgery (CSRF) attacks and other security vulnerabilities. In times like these it is crucial to protect the sensitive personal and business information through prevention, detection and response to different online attacks. To overcome the disadvantages of traditional machine learning approach, the proposed method uses web log files and deep learning algorithm to detect SQLi, XSS, DOS attacks.

The proposed system uses deep learning algorithm which scans the data to search for features that correlate and combine them to enable faster learning without being explicitly told to do so. The server generates a log file automatically created and maintained by a web server. Every time any browser or user-agent, Google included, requests any resource—pages, images, java script file from the server, the server adds a line in the log. In relation to cyber security, log data points out to the red flags in the systems: unusual behaviour, unauthorized access, extreme traffic and any suspicious activities by the attackers. The proposed system designed aim at log monitoring, detect malicious attack signatures and threats, thereby increasing the performance and accuracy of the system.

REFERENCES

[1] Sonali Mishra, “SQL Injection Detection Using Machine Learning”, 2019, San José State University.

[2] Musaab Hasan, Zayed Balbahaith, and Mohammed Tarique, “Detection of SQL Injection Attacks: A Machine Learning Approach”, 2019, International Conference on Electrical and Computing Technologies and Applications (ICECTA).

[3] Harsh Bhagwani,“Log based Dynamic Intrusion Detection of Web Applications”, Indian Institute of Technology, Kanpur, 2019.

[4] Xin Xie, Chunhui Ren, Yusheng Fu , Jie Xu , and Jinhong Guo “SQL Injection Detection for Web Applications Based on Elastic-Pooling CNN”, 2019 IEEE.

[5] Rajashree A. Katole, Dr. Swati S. Sherekar, Dr. Vilas M. Thakare, “Detection of SQL Injection Attacks by Removing the Parameter Values of SQL Query”, 2018, Proceedings of the Second International Conference on Inventive Systems and Control (ICISC 2018) and IEEE.

[6] Zhuang Chen, Min Guo, Lin zhou, “Research on SQL injection detection technology based on SVM”, 2018, MATEC Web of Conferences.

[7] Fawaz A. Mereani and Jacob M. Howe, “Detecting Cross-Site Scripting Attacks using Machine Learning”, 2018, City, University of London Institutional Repository.

[8] Melody Moh*, Santhosh Pininti, Sindhusha Doddapaneni, and Teng-Sheng Moh, “Detecting Web Attacks Using Multi-Stage Log Analysis Year”, 2017, San Jose State University, USA.

[9] Kunal Gupta, Rajni Ranjan Singh, Manish Dixit, “CROSS SITE SCRIPTING (XSS) attack detection using intrusion detection system”, 2017, International Conference on Intelligent Computing and Control Systems.

[10] Shailendra Rathore, Pradip Kumar Sharma, Jong Hyuk Park, “XSSClassifier: An Efficient XSS Attack Detection Approach Based on Machine Learning Classifier on SNSs”, 2017, Journal of information processing systems.

WEB LINKS

[11] https://towardsdatascience.com/multi-layer-neural-networks-with-sigmoid-function-deep-learning-for-rookies-2-bf464f09eb7f

[12] https://machinelearningmastery.com/build-multi-layer-perceptron-neural-network-models-keras/

[13] https://towardsdatascience.com/building-a-convolutional-neural-network-cnn-in-keras-329fbbadc5f5

[14] https://machinelearningmastery.com/convolutional-layers-for-deep-learning-neural-networks/

[15] https://www.screamingfrog.co.uk/22-ways-to-analyse-log-files/

[16] https://sematext.com/blog/log-analysis/

[17] https://www.acunetix.com/websitesecurity/sql-injection2/

[18] https://portswigger.net/web-security/cross-site-scripting

[19] https://owasp.org/www-community/Types_of_Cross-Site_Scripting

[20] https://sqlmap.org/

[21] https://www.cloudflare.com/en-in/learning/ddos/ddos-attack-tools/how-to-ddos/

[22] https://owasp.org/

[23] https://owasp.org/www-project-top-ten/2017/