MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR
(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

Project Report

on

SQL Injection and HTTP Flood DDOS Attack

Detection and Classification based on Log Data

Submitted By:

Kapil Patel

0901CS181050

Faculty Mentor:

Dr. Rajni Ranjan Singh Makwana, Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE
GWALIOR - 474005 (MP) est. 1957

MAY-JUNE 2022

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

Project Report

on

SQL Injection and HTTP Flood DDOS Attack

Detection and Classification based on Log Data

A project report submitted in partial fulfilment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by:

Kapil Patel

0901CS181050

Faculty Mentor:

Dr. Rajni Ranjan Singh Makwana, Assistant Professor

Submitted to:

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE
GWALIOR - 474005 (MP) est. 1957

MAY-JUNE 2022

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

CERTIFICATE

This is certified that **Kapil Patel** (0901CS181050) has submitted the project report titled **SQL Injection and HTTP Flood DDOS AttackDetection and Classification based on Log Data** under the mentorship of **Prof. Rajni Ranjan Singh Makwana**, in partial fulfilment of the requirement for the award of degree of Bachelor of Technology in Computer Science and Engineering from Madhav Institute of Technology and Science, Gwalior.

26/05/2022

Dr.Rajni Ranjan Singh Makwana
Faculty Mentor
Assistant Professor
Computer Science and Engineering

26/05/22

Dr. Manish Dixit
Professor and Head,
Computer Science and Engineering

Dr. Manish Dixit
Professor & HOD
Department of CSE
M.I.T.S. Gwalior

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

DECLARATION

I hereby declare that the work being presented in this project report, for the partial fulfilment of requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering at Madhav Institute of Technology & Science, Gwalior is an authenticated and original record of my work under the mentorship of **Dr. Rajni Ranjan Singh Makwana**, **Assistant Professor**, Computer Science and Engineering

I declare that I have not submitted the matter embodied in this report for the award of any degree or diploma anywhere else.

Kapil Patel
0901CS181050
IVYear
Computer Science and Engineering

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

ACKNOWLEDGEMENT

The full semester project has proved to be pivotal to my career. I am thankful to my institute, **Madhav Institute of Technology and Science** to allow me to continue my disciplinary/interdisciplinary project as a curriculum requirement, under the provisions of the Flexible Curriculum Scheme (based on the AICTE Model Curriculum 2018), approved by the Academic Council of the institute. I extend my gratitude to the Director of the institute, **Dr. R. K. Pandit** and Dean Academics, **Dr.Manjaree Pandit** for this.

I would sincerely like to thank my department, **Department of Computer Science and Engineering, for allowing** me to explore this project. I humbly thank **Dr. Manish Dixit**, Professor and Head, Department of Computer Science and Engineering, for his continued support during the course of this engagement, which eased the process and formalities involved.

I am sincerely thankful to my faculty mentors. I am grateful to the guidance of**Dr. Rajni Ranjan Singh Makwana**, Assistant Professor, Computer Science and Engineering, for his continued support and guidance throughout the project. I am also very thankful to the faculty and staff of the department.

Kapil Patel
0901CS181050
IV Year
Computer Science and Engineering

ABSTRACT

Due to continuous growth in Tech. Industry and also due to COVID - 19 there is a huge increase in web servers and web-applications, almost every office work and educational application are online, due to this there is huge increase in cyber-attacks. So, it is important to detect them as early as possible to stop it before it causes much damage to web-application and data loss. During running of web server, it generates lots of auto generated records on every update and log files are some of them. Since web application generates huge number of logs data it is boring and difficult task to do analysis of log data manually by person. But log data is important to monitor as it contains computer generated records, which contain data about every activity and operations, so analysing these can help in early detection of some types of attacks like SQL injection, DDOS attack, brute force attack, and cross-site scripting (XSS) etc. To improve old method of manually inspection of log analysis in this project, an anomaly detection and classification model has been proposed, which can also be used for early attack detection by analysing log data. In this model I have used machine learning decision tree algorithm to classify the data into three categories like normal logs, SQL injected logs and DDOS attack logs. Here for training and building model I have taken data from real web-applications which I have hosted on local XAMPP server and performed Different SQL injection attacks and DOS attacks on web-applications. After comparing log with trained model, it successfully classifies the logs into different categories and also detects live attacks on applications.

Keyword: Log data, Machine-Learning, Decision Tree, SQL Injection, HTTP Flood DOS Attack

सार:

प्रौद्योगिकी उद्योग में निरंतर वृद्धि के कारण और साथ ही COVID-19 के कारण वेब सर्वर और वेब-अनुप्रयोगों में भारी वृद्धि हुई है, लगभग हर कार्यालय का काम और शैक्षिक अनुप्रयोग ऑनलाइन हैं, इसके कारण साइबर हमलों में भारी वृद्धि हुई है। इसलिए वेब-एप्लिकेशन और डेटा हानि को बहुत नुकसान पहुंचाने से पहले इसे रोकने के लिए जितनी जल्दी हो सके उनका पता लगाना महत्वपूर्ण है। वेब सर्वर चलाने के दौरान यह हर अपडेट पर बहुत सारे ऑटो जेनरेटेड रिकॉर्ड बनाता है और लॉग फाइल उनमें से कुछ हैं। चूंकि वेब एप्लिकेशन बड़ी संख्या में लॉग डेटा उत्पन्न करता है, इसलिए व्यक्ति द्वारा मैन्युअल रूप से लॉग डेटा का विश्लेषण करना उबाऊ और कठिन काम है। लेकिन लॉग डेटा मॉनिटर करने के लिए महत्वपूर्ण है क्योंकि इसमें कंप्यूटर जनित रिकॉर्ड होते हैं, जिसमें हर गतिविधि और संचालन के बारे में डेटा होता है, इसलिए इनका विश्लेषण करने से SQL इंजेक्शन, DDOS अटैक, ब्रूट फोर्स अटैक और क्रॉस- जैसे कुछ प्रकार के हमलों का जल्द पता लगाने में मदद मिल सकती है। साइट स्क्रिप्टिंग (XSS) आदि। इस परियोजना में लॉग विश्लेषण के मैन्युअल निरीक्षण की पुरानी पद्धति में सुधार करने के लिए, एक विसंगति का पता लगाने और वर्गीकरण मॉडल का प्रस्ताव किया गया है, जिसका उपयोग लॉग डेटा का विश्लेषण करके शुरुआती हमले का पता लगाने के लिए भी किया जा सकता है। इस मॉडल में मैंने डेटा को सामान्य लॉग्स, SQL इंजेक्टेड लॉग्स और DDOS अटैक लॉग्स जैसी तीन श्रेणियों में वर्गीकृत करने के लिए मशीन लर्निंग डिसीजन ट्री एल्गोरिथम का उपयोग किया है। यहां प्रशिक्षण और निर्माण मॉडल के लिए मैंने वास्तविक वेब-अनुप्रयोगों से डेटा लिया है जिसे मैंने स्थानीय एक्सएएमपीपी सर्वर पर होस्ट किया है और वेब-अनुप्रयोगों पर विभिन्न एसक्यूएल इंजेक्शन हमलों और डॉस हमलों का प्रदर्शन किया है। प्रशिक्षित मॉडल के साथ लॉग की तुलना करने के बाद यह सफलतापूर्वक लॉग को विभिन्न श्रेणियों में वर्गीकृत करता है और अनुप्रयोगों पर लाइव हमलों का भी पता लगाता है।

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

Symbol	Description
$\sum$	Denote a sum of multiple terms

LIST OF ABBREVIATIONS

Abbreviation	Description
DDOS	Distributed Denial-Of-Service
HTTP	Hypertext Transfer Protocol
XAMPP	X (Cross Platform), Apache, Mysql, PHP, Perl
SQL	Structured Query Language
GUI	Graphical User Interface
IDE	Integrated Development Environment
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative

Chapter 1: PROJECT OVERVIEW

1.1. Introduction

Now a days due to improvement in cyber world and easily available tools web applications faces many suspicious activities and attacks because of script kiddies, they generally perform scanning and attacking a website using an automated vulnerability scanner tools or trying to fuzz a parameter for SQL injection, cross-site scripting (XSS) etc. and often performs DDOS attacks to down the server working etc. In many such cases, logs on the web-server have to be monitored and analyzed to figure out what is going on. If it is a serious case and suspicious matter then require a cyber expert for forensic investigation. Since running server generates huge amount and different types of logs data it is very difficult to monitor manually, even though it is not efficient enough to filter different logs data into different categories and also demand many hours to inspect the data.

Because most of these systems are meant to run around the clock, serving millions of online users throughout the world, high availability and reliability are essential. Any mishaps involving these systems, such as service outages and quality of service degradation, will cause applications to break down and result in severe revenue loss.

In this project a SQL injection and HTTP flood DOS attack log anomaly detection and classification model have been proposed based on machine learning to classify the logs data into categories based on anomaly data present in logs and which further can be used to detect live attacks. For building classification model, a simple rule-based decision tree classifier has been used which is enough to meet demands and successfully classify logs data. Decision tree algorithm comes under supervised machine learning algorithm, in which labeled training data of both cases normal and unusual situations is used for training model. To build the model logs data files is selected in the first step and then the parsing of logs components is done using parsing technique, then labeling and encoding of components according to presence of some patterns or data presence in components is done and after that the labeled components of log is passed to the decision tree classifier to predict the type of logs based on log components data into three categories: Normal logs data, SQL injected logs data, HTTP Flood DDOS Attack logs data.

Experiments with 45897 logs data from real web- application hosted on local XAMPP server, building and testing model shows overall accuracy of 98.88% classification and detection accuracy.In summary, this project focuses on log pattern analysis of normal logs, SQL injected logs and HTTP flood DOS attack logs and extracting interested features from logs and labeling them based on presence of some features and predicting type of log based on labeled data through decision tree classifier.

1.2. Background

In this part of the project report brief introduction of HTTP flood DDOS attack, followed by SQL injection, and then followed by the introduction to the web logs is presented.

1.2.1. HTTP Flood DDOS Attack

DDOS-simply stands for Distributed Denial of Service. It could be of any kind like hijacking a server, port overloading, denying internet-based services etc. HTTP flood DDOS attack is an application layer volumetric attack, mainly focus on crashing the web servers and online web applications. These attacks are comparatively sophisticated, here a huge number of legitimate looking HTTP GET, or POST requests are used to flood the server in this type of attack. This in return causes a denial of services. Figure 1 depicts working view of attack.

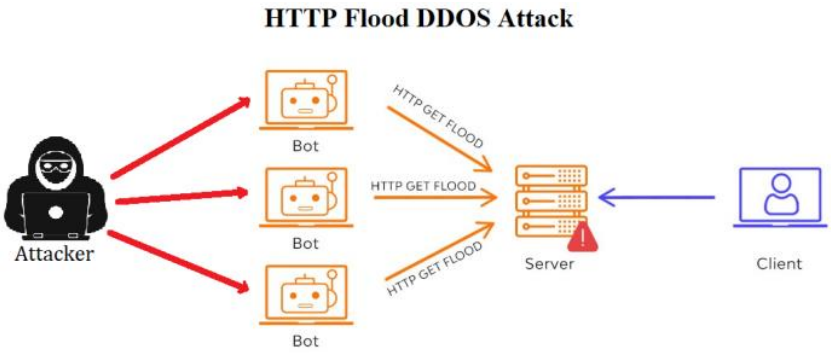


Fig -1: HTTP Flood DDOS Attack

1.2.2. SQL Injection Attack

SQL is a high-level scripting language which is used to store, access and maintain database systems. SQL injection is most common type of SQL attacks in which malicious SQL code is used to read, modify, and delete database information that are not allowed to access for normal users. Even it can also execute administrative operations. Because server contains huge amount of users' data which makes servers valuable target for attackers. By using SQL injection attack hackers also can bypass authentication system, compromised data and can-do information disclosure. Even SQL injections can also be pivoted into remote command execution. Figure 2 depicts a view how SQL query is injected and attack is performed.

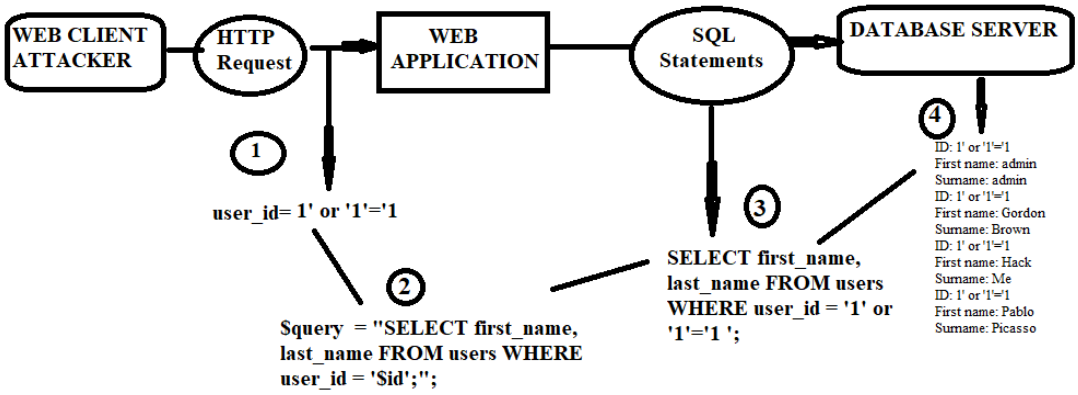


Fig -2: SQL Query Injected Attack

1.2.3. Web server logs

Logs are generally computer automatic generated records, which record every progress and problems during running of applications and systems. A web server log files usually contains the records of user access, requests and errors record of each time whenever user requests from web server. Mainly there are four types of log files in XAMPP Apache Server file that are: access log file, error log file, php error log file, and ssl_request log file. In this project only access log file has taken to predict the logs type based on their content present. Apache access log file records all events that are requested, and response sent by server. Generally,Apache server follows Common Log Format specification by default, so each HTTP request is

written in separate line and composed of several tokens which are separated by spaces, blank values of tokens are represented by a hyphen (-). For demonstration a single log record and its specifications are given in figure 3.

192.168.169.17 - - [26/Mar/2022:21:53:29 +0530] "GET /dvwa/vulnerabilities/fi/?page=include.php HTTP/1.1" 200 4183 "http://192.168.169.107/dvwa/instructions.php" "Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0"		
Host	**192.168.169.17**	The IP address of the client.
Identity	-	The identity information reported by the client.
User	-	The user name of a successful HTTP authentication.
Date	**[26/Mar/2022:21:53:29 +0530]**	The date and time of the request.
Request	**"GET /dvwa/vulnerabilities/fi/?page=include.php HTTP/1.1"**	The request line from the client is given in double quotes.
Status	**200**	The three-digit HTTP status code generated in response to the clients request.
Bytes	**4183**	The number of bytes in the object returned to the client.
Request Header Referer	**"http://192.168.169.107/dvwa/instructions.php"**	The HTTP request header referer contains an absolute or partial address of the page that makes the request.
Request Header User Agent	**"Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0"**	The user agent identifies the application, operating system, vendor and/or version of the requesting user agent.

Fig -3: Example of a Log record

1.3. Objectives and Scope

Organizations that want to improve their cyber security capabilities should create log analysis capabilities that can help them identify and respond to cyber-attacks more quickly. Organizations that use log analysis to successfully monitor their cyber security can make their network assets more difficult to hack. Cyber security monitoring can also assist firms satisfy compliance requirements for cyber security by reducing the frequency and severity of cyber-attacks, promoting quicker reaction to threats, and reducing the frequency and severity of cyber assaults.

Prime objectives of the project are:

- In this project filtering, predicting and classifying machine learning model on the basis of logs text data analysis and pattern matching has been built.
- This project is able to detect and classify normal logs data and anomaly logs data.
- This project applies simplemachine learning algorithmto classify the data into three categories like normal logs, SQL injected logs, DDOS attack logs.
- This project is able to detect live attacks and also able to predict the attack type too.

Scopeoftheprojectisveryvast some of them are.

- This project model can be implemented with running server for early detection and sending warning message to the owner.

- This project model also can be implanted with other API to send message along with malicious activities logs data, so owner can take further action with that user.
- Using project techniques other attack detection model can be also built based on log data analysis.

1.4. Project Features

This project provides GUI along source code so user can modify code according to its need as well as can useit without modifying the source code, which is very easy to use.

1. This project provides option to create visual plot to view the log data, which can be further used to analyse that whether the attack has been performed on web application or not.
2. This project also provides features to view log data after classification in structured view.
3. This project also can be implemented to Detect the live attack on application based on log data analysis.

In project's GUI Tool there are four types of Tools are available which can used for various purpose accordingly. GUI View of Menu of Tool for Log Analysis is shown in figure 4.

Fig -4: GUI View of Menu of Tool for Log Analysis

- Tool 1 Analysis of log data can be done by plotting plot of log data based on no. of request log data is generated on server in that timestamp. This can help in gaining much information about server load. GUI View of option Analysis of Log Datais shown in figure 5.

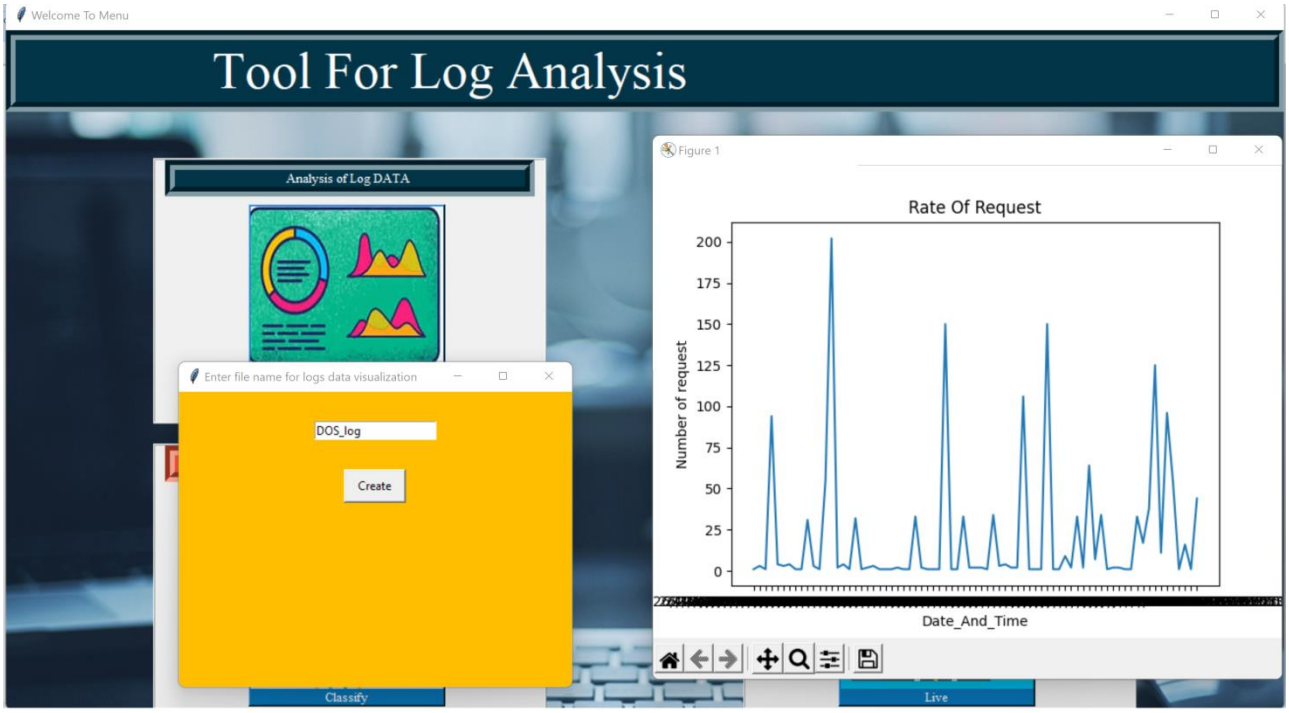


Fig -5: GUI View of option Analysis of Log Data

- In tool option 2 users can view the trained model view of Decision tree model and that tell how decision are being made. Decision tree model is shown in figure 15.
- In tool option 3 the user can enter log file name along with path to classify that file into three categories and create new filtered data files which are Normal_logs.csv, SQLInfected_logs.csv, and DOSAttack_logs.csv. After creating files this option also generates and opens GUI view of that files in new window. GUI View of option Classify the Log Data is shown in figure 6.

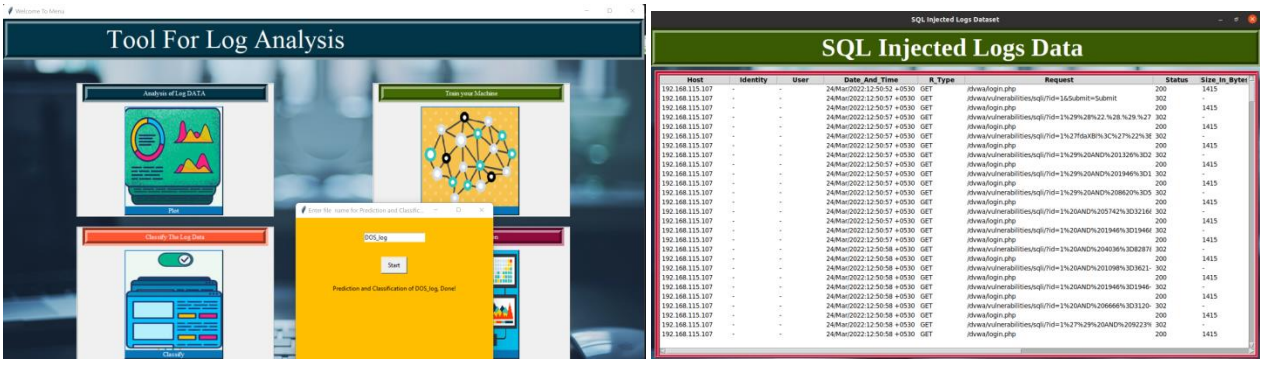


Fig -6: GUI View of option Classify the Log Data

- In 4 options for this Tool provides feature to use this project model for live implementation on server to detect live attack and to generate warning during attack for the user.View of Live Implementation of Attack Detection is shown in figure 7.

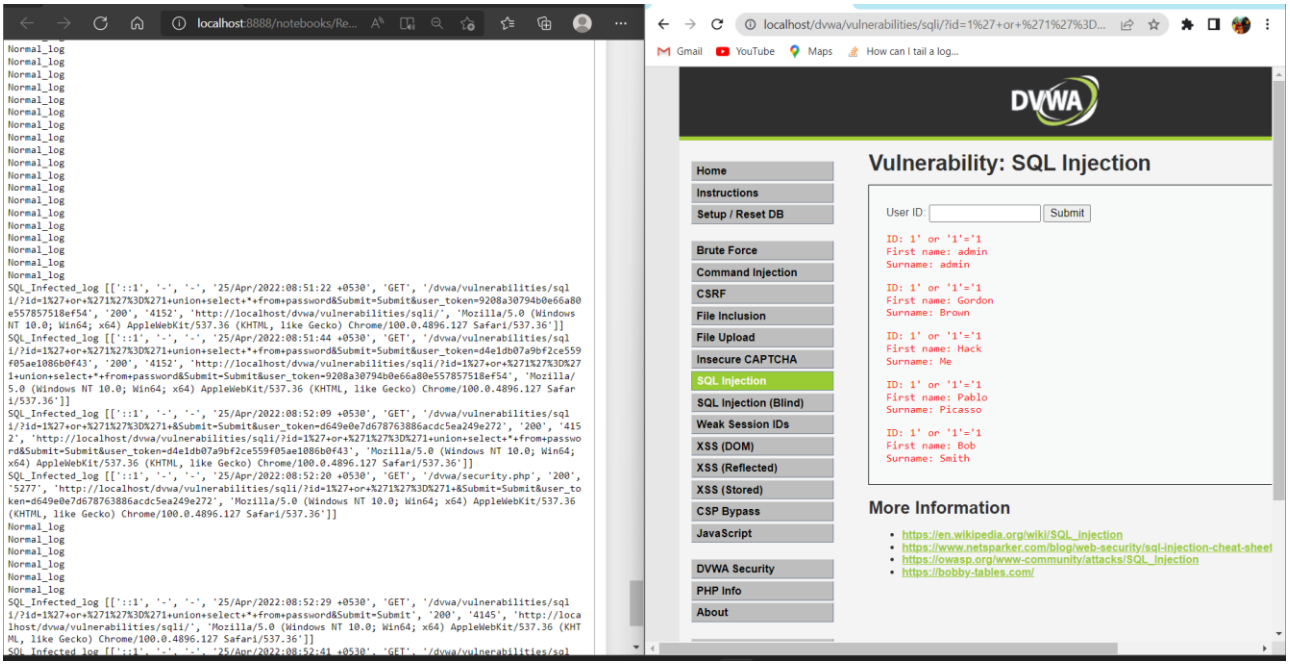


Fig -7: View of Live Implementation of Attack Detection

1.5. Feasibility

SystemFeasibility

The system feasibility study is a critical stage in the design process. A feasibility study evaluates a system proposal for its influence on the organisation, ability to meet user needs, and resource utilisation. The feasibility study determines whether or not the system has been correctly developed.

Softwareimplementationlanguage/technology

ForFrontEnd

TheTechnologiesare

1. Tkinter:The standard Python interface to the Tcl/Tk GUI toolkit is the tkinter package ("Tk interface"). Tkinter is the de facto technique to develop graphical user interfaces (GUIs) in Python, and it comes with all standard Python distributions. It's the only framework that comes with the Python standard library.

2. JupyterNotebook:Jupyter Notebook is a web-based open-source editor for creating and sharing documents that include live code, text, equations, visualisations, and images, etc.

ForBackEnd

TheTechnologiesare

3. Python: Python is a general-purpose programming language that is high-level and interpreted. Its design philosophy prioritises code readability by employing a large amount of indentation. Python is garbage-collected and typed dynamically. It supports a variety of programming paradigms, object-oriented, including procedural and functional programming. It is frequently referred to as a "batteries included" language due to its vast standard library. Python is

a programming language used to build websites and web applications, automate tasks, and perform data analysis.

4. Scikit-Learn: Scikit-learn is a Python-based machine learning library that is available for free. It includes gradient boosting, random forests, support-vector machines, k-means, and DBSCAN, among other regression, classification, and clustering techniques, and is designed to work with the Python numerical and scientific libraries NumPy and SciPy. Scikit-learn is a financially supported NumFOCUS project.

5. Matplotlib: Matplotlib is a graphing package for Python with NumPy, the Python numerical mathematics extension. It is an object-oriented API which is used for embedding charts and plots into applications utilising GUI toolkits such as Tkinter, wxPython, Qt, or GTK.

6. CSV: Comma Separated Values file is a form of plain text file that organises tabular data using particular structuring. It can only contain actual text data—that is separated by comma and contains printable ASCII or Unicode characters—because it is a plain text file. It is generally used for storing data into structured form.

1.6. System Requirement

1.6.1. Requirements

RecommendedSystemRequirements:

- AgoodCPUandaGPUwithat least2GBmemory.
- At least8GBofRAM.
- Microsoft Windows 10 OS
- An active internet connection so that the system can use links to access online resources.

RecommendedSoftwareRequirements:

- PyCharm IDE
- Jupyter Notebook
- XAMPP Server
- Damn Vulnerable Web App (DVWA)

Required Libraries for this project along with their version numbers used while makingandtestingofthisproject:

- Matplotlib 3.5.2
- Pandas 1.4.2
- Scikit-Learn 1.1.0
- Python-csv 0.0.13
- Pre-installed Python packages (tkinter, PIL, os, re, time, copy)

1.6.2. Tools used

Toolsforproject–

1. Jupyter Notebook: Jupyter Notebook is a web-based open-source editor for creating and sharing documents that include live code, text, equations, visualisations, and images, etc.

2. PyCharm IDE: PyCharm is editor for computer programming which provides integrated development environment that focuses on the Python programming language. JetBrains, a Czech firm, developed it.

3. Damn Vulnerable Web App (DVWA): DVWA is an extremely susceptible PHP/MySQL web application. Its main objectives are to assist security professionals in putting their skills and tools to the test in a legal environment, to assist web developers in better understanding the mechanisms of securing web applications, and to assist teachers/students in teaching/learning web application security in a classroom.

4. XAMPP Server: XAMPP is an open and free cross-platform web server solution stack package created by Apache Friends, comprising mostly of the MariaDB database,Apache HTTP Server, and interpreters for Perl and PHP scripts.

5. Sqlmap: sqlmap is a free software security testing tool for discovering and exploiting SQL injection problems and gaining control of database servers. It includes a powerful detection engine, numerous niche functionalities for the ultimate tester, and a wide range of switches that cover everything from database fingerprinting to data retrieval from databases to accessing the underlying file system and running commands on the operating system.

6. Slowloris Script: Slowloris Script focuses on application layer DDoS attack that leverages partial HTTP requests to generate connections between a single machine and a specified Web server, then keeps those connections running as long as possible, overloading and delaying the target.

7. Pentmenu:Pentmenu is an automation tool based on the PentBox that may be used to do numerous network pen-testing tasks, such as network attacks, DOS Attacks etc.

Chapter 2: Literature Review

Anomaly detection refers to find unusual patterns in data that do not follow regular patterns or give unexpected behavior. In past, there had been a lot of traditional intrusion detection models were proposed, on top of that, traditional intrusion detection technology requires programmers or operators to extract attack features manually, and recognize patterns of typical attacks merely based on keyword search and rule match [1]. In other words, the traditional method cannot recognize unknown attacks and leads to many false positives. To compensate for the drawbacks of older methods many machine learning techniques are now used in log-based anomaly detection as a result of the advancement of machine learning [2]. Anomaly detection approaches can be classified into two categories based on the data type and machine learning technology used: supervised anomaly detection [3] and unsupervised anomaly detection [4, 5]. In both regular and abnormal conditions, the supervised technique requires standard training data that is precisely specified.Unsupervised approaches, on the other hand, do not require labels.

The issue may be increased if developers lack the machine learning background knowledge required to comprehend these methods. However, to our knowledge, no open-source log-based anomaly detection technologies are currently available. A comparison of existing anomaly detection systems is also lacking.As a consequence, automated anomaly detection solutions based on log analysis are in high demand.

In 2004, where a Decision Tree model was applied to find error detection for web request log system in [6].In 2010 based on web log data K. R. Suneetha, R. Krishnamoorth build a classification model to identify interested users using decision trees in [7]. Similarly in 2017 based on web access log data a machine learning model is applied by Qimin Cao and YinrongQiao to detect anomalies in web log file in [8]. Influenced by these papers, this project adopted a more enhanced filtering and classifying supervised machine learning model with simple text analysis and pattern matching approach that can be used to detect and classify both normal logs data and anomaly logs data present into categories.

Chapter 3: Preliminary design

3.1. Model Framework Overview

Figure 11 demonstrates the complete model framework overview. To detect anomaly based on web access logs data the project mainly involves four phases: log collection, log parsing, features extraction and labelling and then training decision tree classifier and prediction of attacks.

Log collection: During run time of web-application it generates logs on every request. These valuable records could be utilized for various purposes but here used for anomaly detection, and so logs data are collected in first step for further usage. For building model data is collected by self-hosting web application on local XAMPP server. Figure 8 depicts a view log collection unstructured logs data.

```
['::1 - - [10/Mar/2022:20:57:29 +0530] "GET /dvwa/vulnerabilities/sqli/?id=1&Submit=Submit HTTP/1.1" 200 4105 "htt
p://localhost/dvwa/vulnerabilities/sqli/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, li
ke Gecko) Chrome/99.0.4844.51 Safari/537.36"', '::1 - - [10/Mar/2022:20:57:32 +0530] "GET /dvwa/vulnerabilities/sq
li/?id=2&Submit=Submit HTTP/1.1" 200 4106 "http://localhost/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" "Mozill
a/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.51 Safari/537.36"',
'::1 - - [10/Mar/2022:20:57:34 +0530] "GET /dvwa/vulnerabilities/sqli/?id=3&Submit=Submit HTTP/1.1" 200 4101 "htt
p://localhost/dvwa/vulnerabilities/sqli/?id=2&Submit=Submit" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKi
t/537.36 (KHTML, like Gecko) Chrome/99.0.4844.51 Safari/537.36"', '::1 - - [10/Mar/2022:20:57:38 +0530] "GET /dvw
a/vulnerabilities/sqli/?id=4&Submit=Submit HTTP/1.1" 200 4107 "http://localhost/dvwa/vulnerabilities/sqli/?id=3&Su
bmit=Submit" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.51
Safari/537.36"', '::1 - - [10/Mar/2022:20:57:41 +0530] "GET /dvwa/vulnerabilities/sqli/?id=5&Submit=Submit HTTP/1.
1" 200 4103 "http://localhost/dvwa/vulnerabilities/sqli/?id=4&Submit=Submit" "Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.51 Safari/537.36"', '::1 - - [10/Mar/2022:20:57:44 +0
530] "GET /dvwa/vulnerabilities/sqli/?id=6&Submit=Submit HTTP/1.1" 200 4046 "http://localhost/dvwa/vulnerabilitie
s/sqli/?id=5&Submit=Submit" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chro
me/99.0.4844.51 Safari/537.36"', '::1 - - [10/Mar/2022:20:59:15 +0530] "GET /dvwa/vulnerabilities/sqli/?id=%27+or
+%271%27%3D%271&Submit=Submit HTTP/1.1" 200 4388 "http://localhost/dvwa/vulnerabilities/sqli/?id=6&Submit=Submit"
"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.51 Safari/537.3
6"', '::1 - - [10/Mar/2022:21:01:37 +0530] "GET /dvwa/vulnerabilities/sqli/?id=%27+or+%271%27%3D%271+union+select+
*+from+password&Submit=Submit HTTP/1.1" 200 4046 "http://localhost/dvwa/vulnerabilities/sqli/?id=%27+or+%271%27%3
D%271&Submit=Submit" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.
```

Fig -8: Unstructured Logs Data

***Log parsing*:** logs are continuously written records and generally unstructured data which have to be parsed to extract useful information out of them and to make them structured. For splitting logs and extracting features out of them regular expressions are used so that information can be easily stored and manipulated. Figure 9 depicts a view structured logs data.

Host	Identity	User	Date_And_Time	R_Type	Request	Status	Size_In_Bytes	Request_Header_Referer	Request_Header_User_Agent
168.115.107	-	-	24/Mar/2022:12:50:37 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit	302	-	-	sqlmap/1.5.8#stable (http://sqlmap.org)
168.115.107	-	-	24/Mar/2022:12:50:52 +0530	GET	/dvwa/login.php	200	1415	-	sqlmap/1.5.8#stable (http://sqlmap.org)
168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit	302	-	-	sqlmap/1.5.8#stable (http://sqlmap.org)
168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/login.php	200	1415	-	sqlmap/1.5.8#stable (http://sqlmap.org)
168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1%29%28%22.%28....	302	-	-	sqlmap/1.5.8#stable (http://sqlmap.org)
...	...	...	...	...	...	...	...	...	...
168.115.107	-	-	24/Mar/2022:12:53:27 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit...	302	-	-	sqlmap/1.5.8#stable (http://sqlmap.org)
168.115.107	-	-	24/Mar/2022:12:53:27 +0530	GET	/dvwa/login.php	200	1415	-	sqlmap/1.5.8#stable (http://sqlmap.org)
2.168.169.17	-	-	26/Mar/2022:21:47:00 +0530	GET	/favicon.ico	404	301	-	Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko...
2.168.169.17	-	-	26/Mar/2022:21:47:12 +0530	GET	/Project-Online-Shopping-Website-master/res/Th...	200	20158	-	Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko...
2.168.169.17	-	-	26/Mar/2022:21:49:37 +0530	GET	/dvwa/favicon.ico	200	1406	-	Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko...

Fig -9: Structured Logs Data

Feature extraction and labelling: After parsing logs into separate parts, we need to further search some patterns in logs parts and based on those patterns, label them and then encode them into numerical feature arrays, whereby machine learning models can be applied. Figure 10 depicts a view labelled encoded logs data.

	Request	Request_Header_Referer	Request_Header_User_Agent	R_Type_n	Status_n
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
...	...	...	...	...	...
45889	0	0	0	0	0
45890	0	0	0	0	0
45891	0	0	0	0	0
45892	0	0	0	0	0
45893	0	0	0	0	0

Fig -10: Labeled Encoded Logs Data

Training decision tree classifier and prediction: Now, the feature arrays of data can be used to feed to machine learning decision tree classifier models for training and to generate a model that can be used for prediction and classification. The trained model now can be further used to classify a new log data into a normal log, SQL injected log, DDOS attack log based on anomaly present in data.

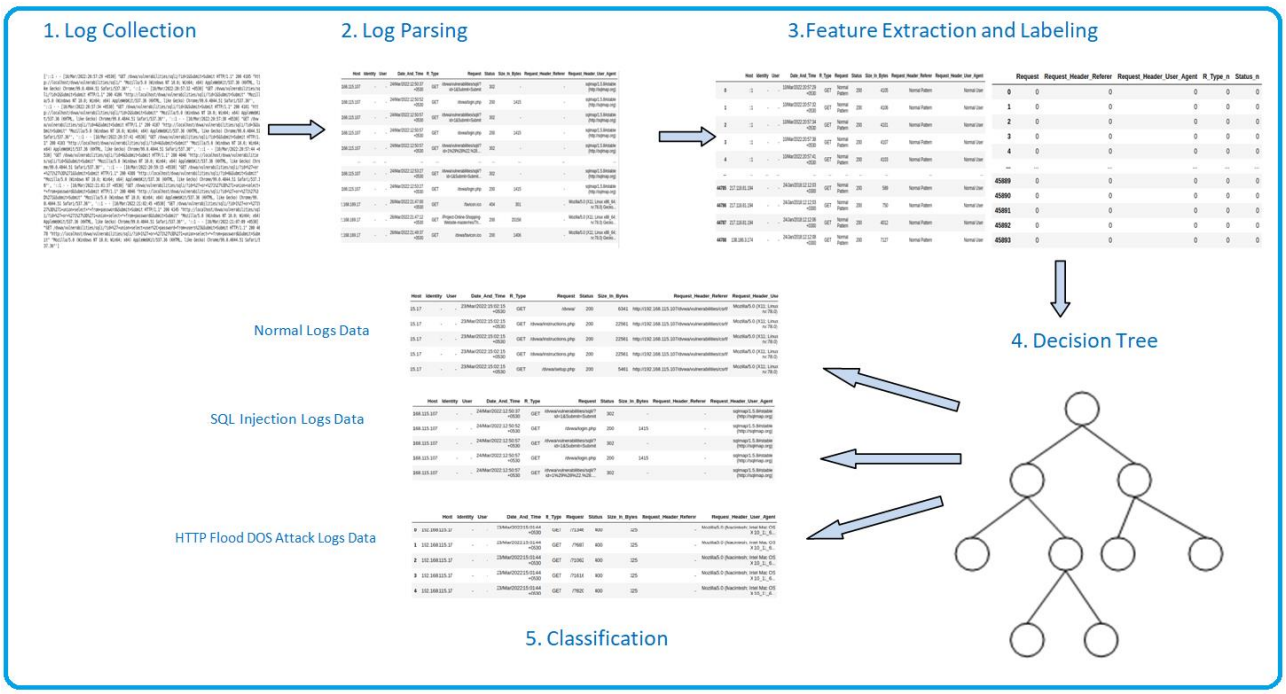


Fig -11: Anomalous Log Detection and Classification System Framework Overview

3.2. Model Design and Implementation

In this part of the project report detailed explanation of model framework is presented. In this section, it tells how to extract useful features and find patterns in them and label data based on that pattern. After labelling them the decision tree classifier is used to predict type of log data and to classify the log data present in file into different categories.

3.2.1. Data preprocessing

***Features extraction**:* After parsing the logs file and making them structured it is important to select only important features out of them which are not much variable can provide valuable information about the log data. The Features that are taken are: HTTP status code, request type, request, request header referrer and request header user-agent which give enough information to detect presence of anomaly. HTTP request types are generally divided into two categories GET and POST. GET is used to get something from server without changing it and carries request without hiding parameter details in URL, whereas POST is used to make changes in data based on request and is more secure. HTTP status code tells whether a request has completed or not, status code has special meaning, it is generated based on responsive status of a web server. Some of the commonly found HTTP status code and information are listed in figure 12.

HTTP Status code	Information
200	OK
206	Created
301	Moved Permanently
302	Found
304	Not Modified
400	Bad Request
403	Forbidden
404	Not Found
414	URI Too Large

Fig -12: Some common HTTP status code

It is often found that in SQL injection attack logs data, malicious SQL query is injected into some parameters to performthe attack, so to filter out such query this model uses request, request header referer, request header user-agent of log data to filter out such injected query. To distinguish features between normal data and SQL injected data few examples of such logs are taken in figure 13 and some features are also underlined to filter out.

Fig -13: SQL Query Injected logs.

Similarly in HTTP flood DDOS attack log data also has some patterns that are common in request data, status code, request header referer, request header user-agent of log data. Few examples of such logs are taken in figure 14 and some features are also underlined to filter out.

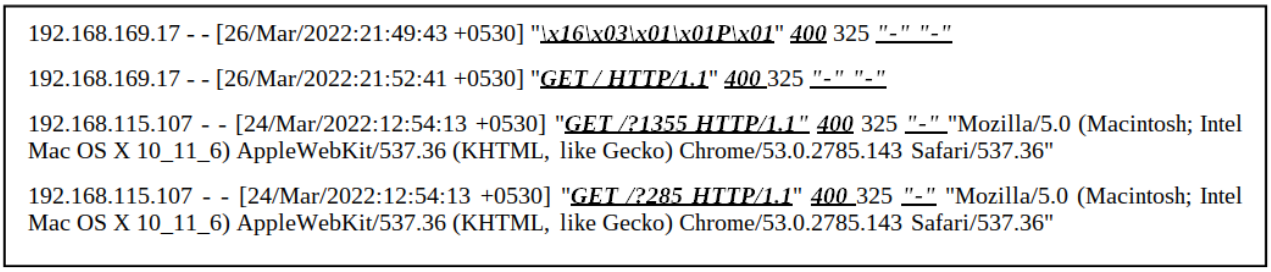


Fig -14: HTTP Flood DOS Attack logs.

Data labelling:Based on patterns found in logs data different type of labels are assigned to log data tokens. Labels taken for Labelling for each unit of data are listed in table 1.

Table -1: Labels used for each unit of data

Log Tokens	Label based on Patterns		
Request Type	GET	POST	
Request	Normal Pattern	SQL Injected Pattern	DOS Attack Pattern
Request Header Referer	Normal Pattern	SQL Injected Pattern	Empty
Request Header User Agent	Normal User	Special User	No User
Log Classification Label	Normal log	SQL Injected log	DOS Attack log

Based on collection of logs data during attack and normal situation and by analysing each unit of data, labels have been assigned manually into three categories for classification are: normal log, SQL injected log, DOS attack log. After labelling based on these patterns, these labels have to be encoded into numerical features to train the model. To do so, I have encoded normal data as ‘0’, SQL injected data as ‘1’and DOS attack data as ‘2’. Now after encoding these data are passed to train machine learning models.

3.2.2. Decision Tree Classifier

After data parsing, processing and preparing training dataset to build a model, now a suitable machine learning algorithm has to be selected which best fits for the model, to do so a simple rule-based decision tree classifier is enough to predict and classify the log data.

Decision Tree algorithm is a classification algorithm, it adds the data point to a particular labelled group on the basis of some conditions. Decision Tree graphically represents flowchart-like structure which demonstrates all the possible solutions that can be used to take a decision. Decisions are generally taken based on some conditions and which can be easily explained. Splitting of decision tree in this model is done by based on Gini impurity, which is used to split nodes when categorical data has to be predicted.

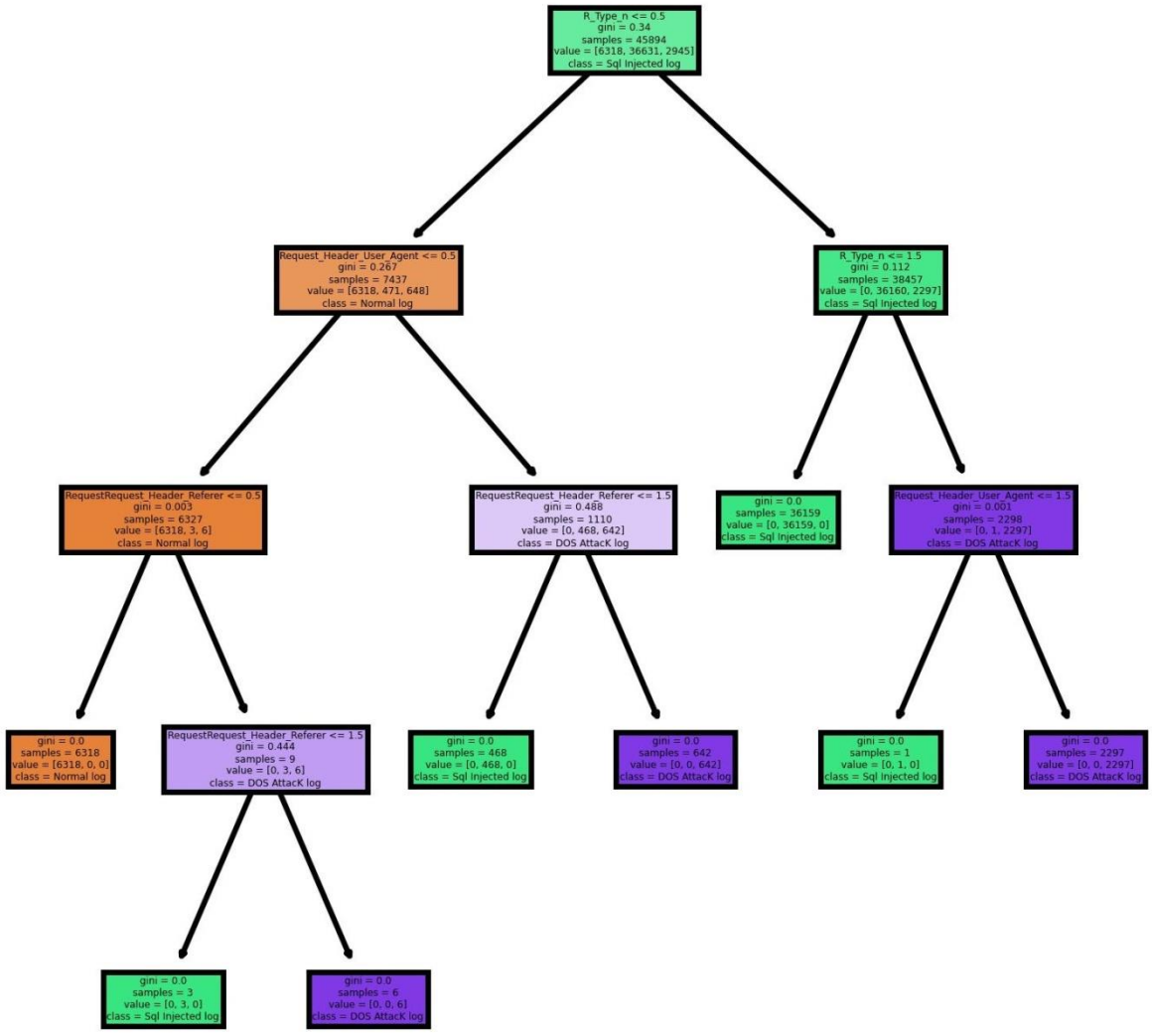


Fig -15: Decision Tree Classifier of Model

Gini impurity is used to divide data into multiple branches in decision trees. For classification and regression, decision trees are utilised. Impurity is used in decision trees to determine which characteristic is the best at each phase. The extent of the discrepancy between the set of points that the attribute has and the number of points that it does not have is the impurity of the attribute. The attribute impurity is zero when the number of points it has equals the number of points it does not have.To compute the Gini Impurity, first grasp its fundamental mechanism. First, we'll choose any data point from the dataset at random. Then, based on the class distribution in the given dataset, we'll classify it at random. Determine the Gini Impurity for both the ideal and imperfect split.The optimal split is obtained in a Decision Tree method by maximising the Gini Gain, which is determined by formula below:

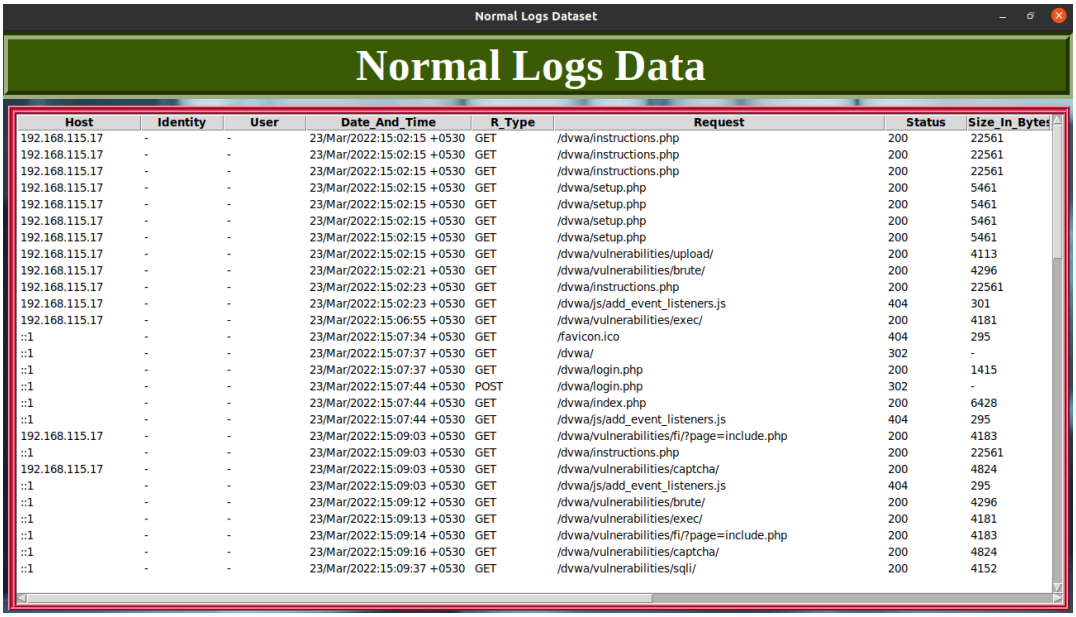
$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$

C = total number of categories

P(i) = probability of picking the piece of data with the category i.

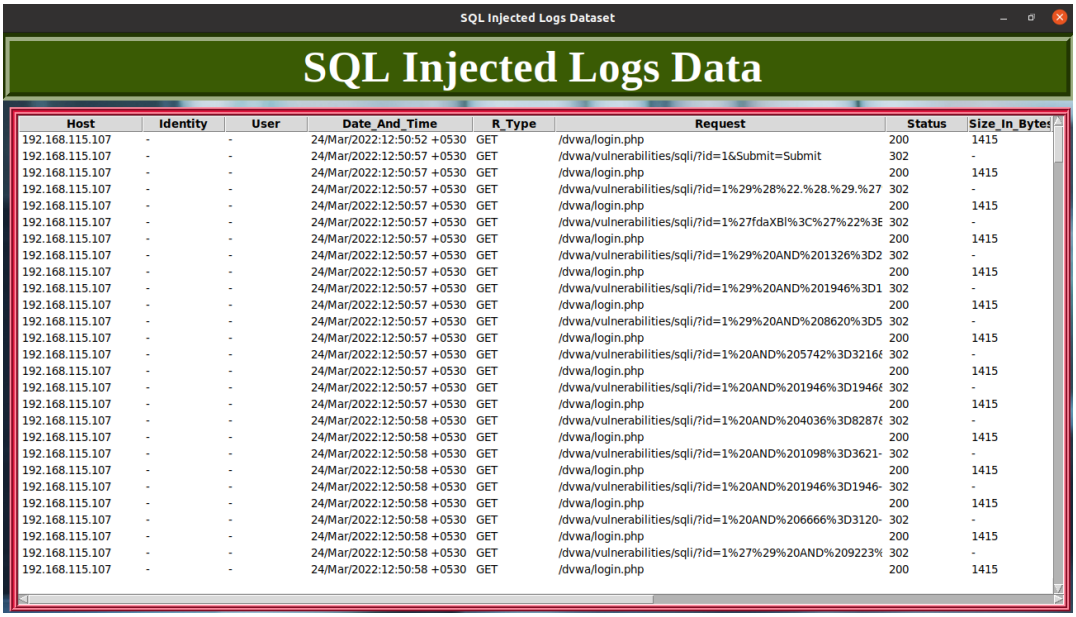
3.2.3. Prediction and Classification

After training decision tree classifier of model, now model is tested with new dataset to make predictions, which successfully classifies the logs dataset into the normal data set, SQL injected dataset and DOS attack dataset shown in figures 16, 17, 18. After getting these datasets, inspection of each piece of logs data is done manually to make sure that there is not any false prediction, classification and labelling. After checking all datasets, this model is further used for live attack detection of SQL injection attack and HTTP flood DOS attack and found that the model is predicting perfectly.



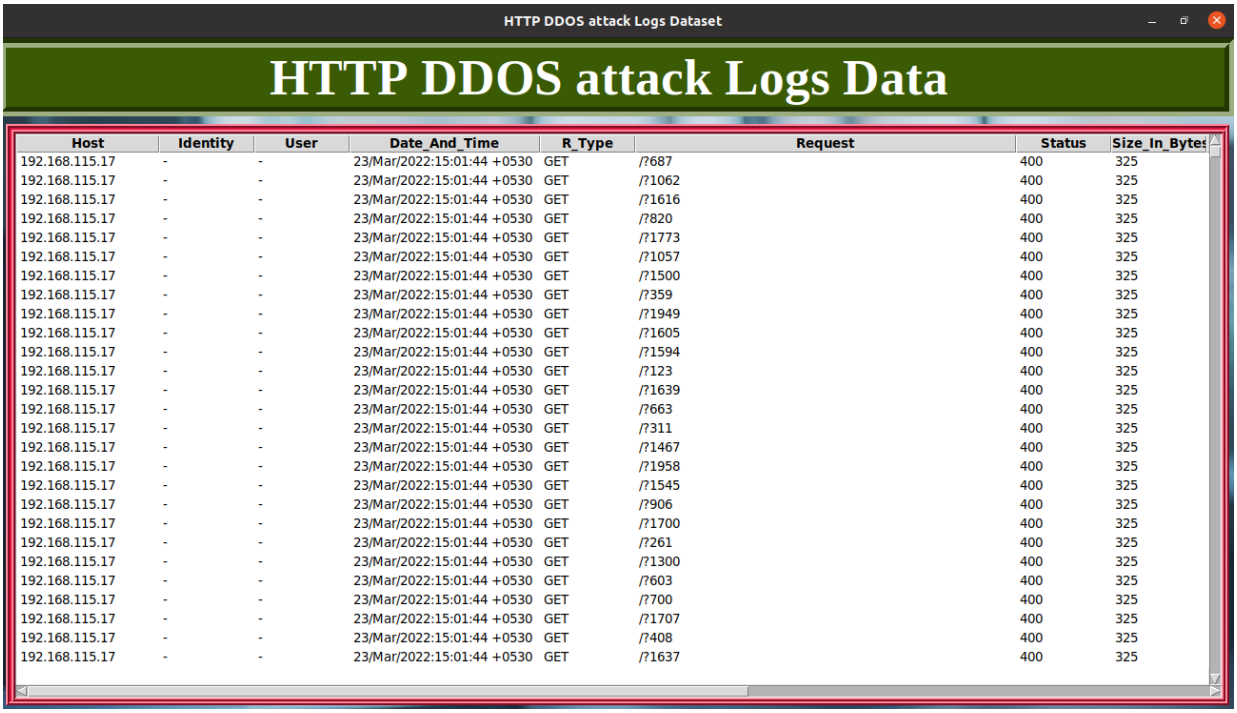
Host	Identity	User	Date_And_Time	R_Type	Request	Status	Size_In_Bytes
192.168.115.17	-	-	23/Mar/2022:15:02:15 +0530	GET	/dvwa/instructions.php	200	22561
192.168.115.17	-	-	23/Mar/2022:15:02:15 +0530	GET	/dvwa/instructions.php	200	22561
192.168.115.17	-	-	23/Mar/2022:15:02:15 +0530	GET	/dvwa/instructions.php	200	22561
192.168.115.17	-	-	23/Mar/2022:15:02:15 +0530	GET	/dvwa/setup.php	200	5461
192.168.115.17	-	-	23/Mar/2022:15:02:15 +0530	GET	/dvwa/setup.php	200	5461
192.168.115.17	-	-	23/Mar/2022:15:02:15 +0530	GET	/dvwa/setup.php	200	5461
192.168.115.17	-	-	23/Mar/2022:15:02:15 +0530	GET	/dvwa/setup.php	200	5461
192.168.115.17	-	-	23/Mar/2022:15:02:15 +0530	GET	/dvwa/vulnerabilities/upload/	200	4113
192.168.115.17	-	-	23/Mar/2022:15:02:21 +0530	GET	/dvwa/vulnerabilities/brute/	200	4296
192.168.115.17	-	-	23/Mar/2022:15:02:23 +0530	GET	/dvwa/instructions.php	200	22561
192.168.115.17	-	-	23/Mar/2022:15:02:23 +0530	GET	/dvwa/js/add_event_listeners.js	404	301
192.168.115.17	-	-	23/Mar/2022:15:06:55 +0530	GET	/dvwa/vulnerabilities/exec/	200	4181
::1	-	-	23/Mar/2022:15:07:34 +0530	GET	/favicon.ico	404	295
::1	-	-	23/Mar/2022:15:07:37 +0530	GET	/dvwa/	302	-
::1	-	-	23/Mar/2022:15:07:37 +0530	GET	/dvwa/login.php	200	1415
::1	-	-	23/Mar/2022:15:07:44 +0530	POST	/dvwa/login.php	302	-
::1	-	-	23/Mar/2022:15:07:44 +0530	GET	/dvwa/index.php	200	6428
::1	-	-	23/Mar/2022:15:07:44 +0530	GET	/dvwa/js/add_event_listeners.js	404	295
192.168.115.17	-	-	23/Mar/2022:15:09:03 +0530	GET	/dvwa/vulnerabilities/fi/?page=include.php	200	4183
::1	-	-	23/Mar/2022:15:09:03 +0530	GET	/dvwa/instructions.php	200	22561
192.168.115.17	-	-	23/Mar/2022:15:09:03 +0530	GET	/dvwa/vulnerabilities/captcha/	200	4824
::1	-	-	23/Mar/2022:15:09:03 +0530	GET	/dvwa/js/add_event_listeners.js	404	295
::1	-	-	23/Mar/2022:15:09:12 +0530	GET	/dvwa/vulnerabilities/brute/	200	4296
::1	-	-	23/Mar/2022:15:09:13 +0530	GET	/dvwa/vulnerabilities/exec/	200	4181
::1	-	-	23/Mar/2022:15:09:14 +0530	GET	/dvwa/vulnerabilities/fi/?page=include.php	200	4183
::1	-	-	23/Mar/2022:15:09:16 +0530	GET	/dvwa/vulnerabilities/captcha/	200	4824
::1	-	-	23/Mar/2022:15:09:37 +0530	GET	/dvwa/vulnerabilities/sqli/	200	4152

Fig -16: Normal Logs Data after Classification



Host	Identity	User	Date_And_Time	R_Type	Request	Status	Size_In_Bytes
192.168.115.107	-	-	24/Mar/2022:12:50:52 +0530	GET	/dvwa/login.php	200	1415
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit	302	-
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/login.php	200	1415
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1%29%28%22.%28.%29.%27	302	-
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/login.php	200	1415
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1%27fdaXBl%3C%27%22%3E	302	-
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/login.php	200	1415
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1%29%20AND%201326%3D2	302	-
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/login.php	200	1415
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1%29%20AND%201946%3D1	302	-
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/login.php	200	1415
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1%29%20AND%208620%3D5	302	-
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/login.php	200	1415
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1%20AND%205742%3D3216	302	-
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/login.php	200	1415
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1%20AND%201946%3D1946	302	-
192.168.115.107	-	-	24/Mar/2022:12:50:57 +0530	GET	/dvwa/login.php	200	1415
192.168.115.107	-	-	24/Mar/2022:12:50:58 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1%20AND%204036%3D8287	302	-
192.168.115.107	-	-	24/Mar/2022:12:50:58 +0530	GET	/dvwa/login.php	200	1415
192.168.115.107	-	-	24/Mar/2022:12:50:58 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1%20AND%201098%3D3621-	302	-
192.168.115.107	-	-	24/Mar/2022:12:50:58 +0530	GET	/dvwa/login.php	200	1415
192.168.115.107	-	-	24/Mar/2022:12:50:58 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1%20AND%201946%3D1946-	302	-
192.168.115.107	-	-	24/Mar/2022:12:50:58 +0530	GET	/dvwa/login.php	200	1415
192.168.115.107	-	-	24/Mar/2022:12:50:58 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1%20AND%206666%3D3120-	302	-
192.168.115.107	-	-	24/Mar/2022:12:50:58 +0530	GET	/dvwa/login.php	200	1415
192.168.115.107	-	-	24/Mar/2022:12:50:58 +0530	GET	/dvwa/vulnerabilities/sqli/?id=1%27%29%20AND%209223%	302	-
192.168.115.107	-	-	24/Mar/2022:12:50:58 +0530	GET	/dvwa/login.php	200	1415

Fig -17: SQL Injected Logs Data after Classification



Host	Identity	User	Date_And_Time	R_Type	Request	Status	Size_In_Bytes
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?687	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1062	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1616	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?820	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1773	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1057	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1500	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?359	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1949	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1605	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1594	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?123	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1639	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?663	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?311	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1467	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1958	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1545	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?906	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1700	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?261	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1300	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?603	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?700	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1707	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?408	400	325
192.168.115.17	-	-	23/Mar/2022:15:01:44 +0530	GET	/?1637	400	325

Fig -18: HTTP DDOS Attack Logs Data after Classification

Chapter 4: Final Analysis and Design

4.1. Results

4.1.1. Dataset

The data has been collected by self-hosting web applications on local XAMPP server and by performing HTTP flood DOS attack using pentmenu tool and slowloris script and SQL Injection by sqlmap tool and burp suite, etc. This dataset consists of three types of logs dataset during normal browse, SQL injection attack and HTTP DOS attack which is visualized in figure 19, 20, 21.

For training dataset 45897 logs data are taken from web access log file, which has data during normal running of web server and also during attack, which is enough for creating training and testing dataset for the model.

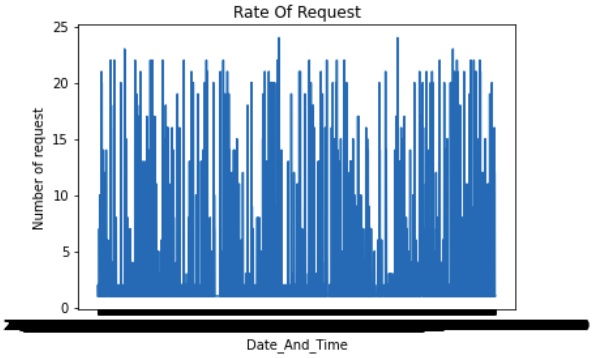


Fig -19:Logs during normal browse.

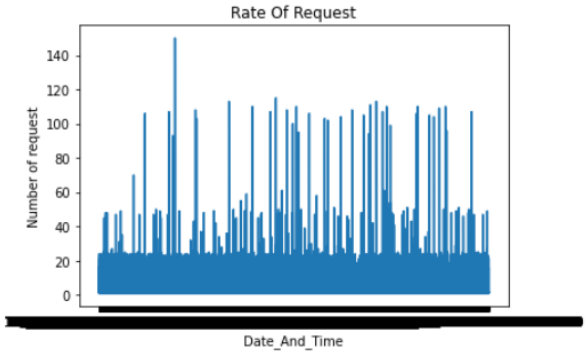


Fig -20: Logs during SQL injection attack.

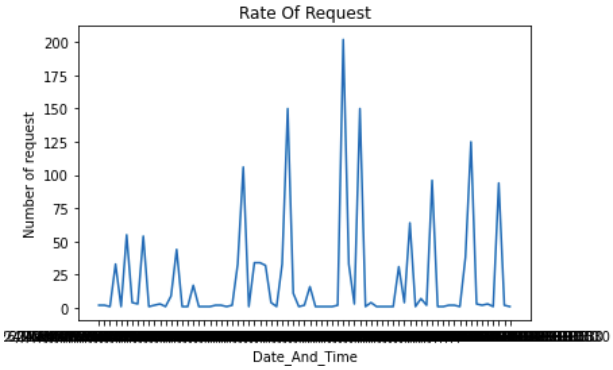


Fig -21: Logs during HTTP DOS attack.

4.1.2. Confusion Metrics:

Confusion metrics provides a holistic view of the classification model. Confusion matrix is one of the classification matrices, used to evaluate performance of classification algorithms. To examine the anomaly detection and classification system, this project first examines the model by using confusion matrix, which describe the performance of detection and classification model in detail. Actual values: column in confusion matrix that are real values. Predicted values: rows in confusion matrix are the predictions. Based on actual and predicted logs confusion metrics for model is shown in table 2.

Table -2: Confusion Matrix

	Actual Label			
Predicted Label		Normal logs	SQL Injected logs	DOS Attack logs
	Normal logs	6301(TP)	18	0
	SQL Injected logs	470	36162(TP)	0
	DOS Attack logs	20	6	2920(TP)

4.2. Result Analysis

The result calculated based on confusion matrix is presented in table 3. Accuracy tells the percentage of total number of predictions and classifications of log data that were correctly predicted and classified. Precision tells the proportion of really positive prediction and classification of logs data. Recall tells the measure of identifying true positives of predictions and classifications of logs data. F1-Score gives harmonic mean of the precision of the classifier model and recall of the classifier model. Based on this analysis it is found that the model successfully predicts and classifies the log data with (TP=45383) out of (45897) with overall accuracy of 98.88 %.

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{TP} + \boldsymbol{TN}}{\boldsymbol{TP} + \boldsymbol{TN} + \boldsymbol{FP} + \boldsymbol{FN}}$$

$$\boldsymbol{Precision} = \frac{\boldsymbol{TruePositive}}{\boldsymbol{TruePositive} + \boldsymbol{FalsePositive}}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}}$$

$$F_1\text{−score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}$$

Table -3: Result of Prediction and Classification

	Accuracy	Precision	Recall	F1-score
Normal logs	98.89%	1.0	0.93	0.96
SQL Injected logs	98.92%	0.99	1.0	0.99
DOS Attack logs	99.94%	0.99	1.0	1.0

4.3. Application

1. This project provides option to create visual plot to view the log data, which can be further used to analyse that whether the attack has been performed on web application or not.
2. This project also gives provides features to view log data after classification in structured view, which in help easy analysis of log data.
3. This project can be implemented to Detect the live attack on application based on log data analysis shown in figure 22.

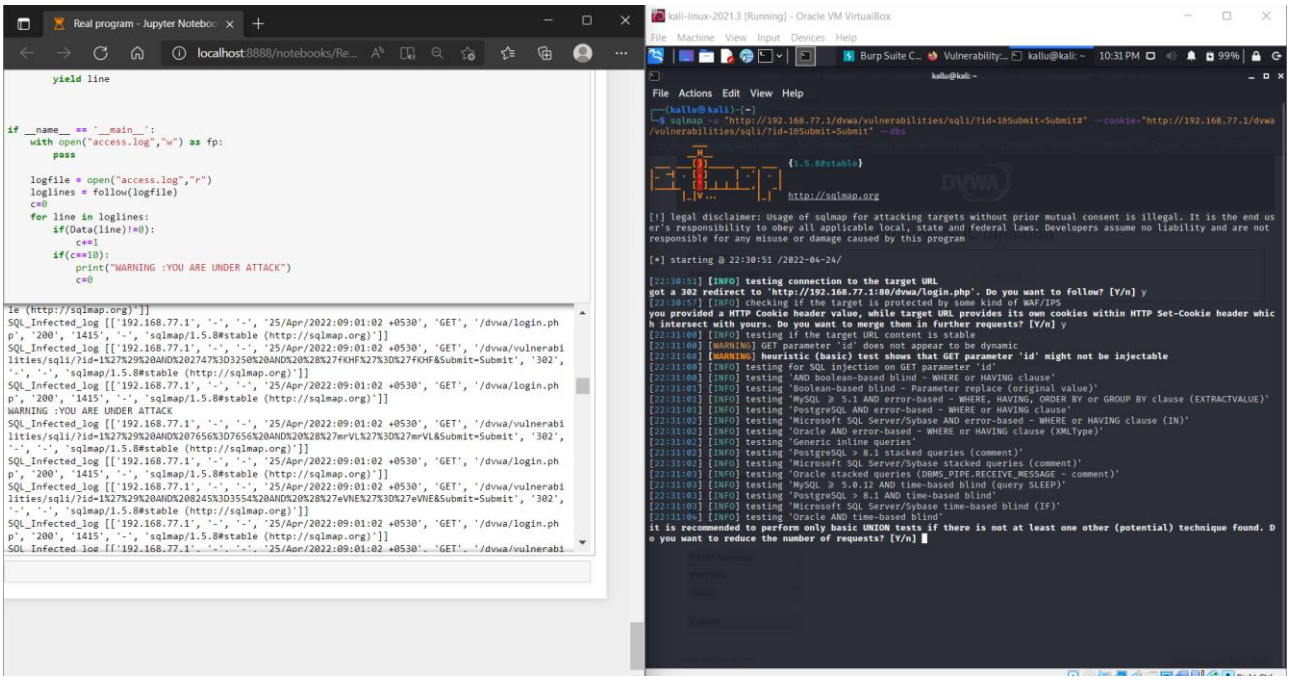


Fig -22: Live Attack Detection on Application

4. This project model can be implemented with running server for early detection and sending warning message to the owner.
5. This project model also can be implanted with other API to send message along with malicious activities logs data, so owner can take further action with that user.
6. Using project techniques other attack detection model can be also built based on log data analysis.

4.4. Problems faced

The problems faced during the project development are:

1. No priordataset is available for building model.
2. No proper idea how to build the model at initial stage.
3. For collect log data different types of web application is hosted on XAMPP Server.
4. To get log data during attack different types of attacking tool are used which for which kali Linux has to be installed.
5. For making log data structured different types of regular expression and parsing methods are applied.
6. For filtering and labelling different types of method was applied for better accuracy.
7. Lots of research along withstudy work is done to build this project.

4.5. Limitations

The Limitations of projects are:

1. As this project focus on log data analysis, unless data is not written in log file, project model cannot predict whether log data is malicious or not.
2. As this project works on log data, which if modified this may lead to wrong analysis.
3. Over accuracy of project model is 98.88%, which there is also chance of false prediction of log data.
4. Since model is built with log data which are generated by easily available attacking tools, this project may lack in accurate prediction in some cases with different attacking pattern tool in attack detection.
5. As script kiddies' useseasily available tools but professional hacker and cracker prefer to write their own script so, they may overpass this detection mechanism.

4.6. Conclusion

This projectbuilds a detection and classification model based on supervised machine learning for web server access log file. The model is able to analyse, detect and classify the log data based on anomaly present in access log data. The data for building model is collected data from real web-applications which has been hosted on local XAMPP server and performed different SQL injection attacks and DOS attacks on that web server. In this project report first extraction of valuable features have been done, based on some patterns and presence of some features and then labelling and encoding of those features have been presented, based on prepared dataset a decision tree classifier has been built and trained. Based on results it is found that this model successfully classifies web access log file data into three categories normal log file, SQL injected log file, and DOS log file. After testing this model, it was found that it achieved overall 98.88 % accuracy of detection and classification.

REFERENCES

[1] A. J. Oliner, A. Ganapathi, and W. Xu, “Advances and challenges in log analysis,” Commun. ACM, vol. 55, pp. 55–61, 2012.

[2] W. Xu, L. Huang, A. Fox, D. A. Patterson, and M. I. Jordan, “Detecting large-scale system problems by mining console logs,” in Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 2124, 2010, Haifa, Israel, 2010, pp. 37–46.

[3] Waseem Ahmed, Yong Wei Wu, "Reliability Prediction Model for SOA Using Hidden Markov Model", ChinaGrid Annual Conference (ChinaGrid) 2013 8th, pp. 40-45, 2013.

[4] Y. Liang, Y. Zhang, H. Xiong, and R. K. Sahoo, “Failure prediction in IBM bluegene/l event logs,” in Proceedings of the 7th IEEE International Conference on Data Mining

[5] J. Lou, Q. Fu, S. Yang, Y. Xu, and J. Li, “Mining invariants from console logs for system problem detection,” in USENIX Annual Technical Conference, 23-25, 2010.

[6] M. Y. Chen, A. X. Zheng, J. Lloyd, M. I. Jordan, and E. A. Brewer, “Failure diagnosis using decision trees,” in 1st International Conference on Autonomic Computing), 2004, pp.36–43.

[7] K. R. Suneetha, R. Krishnamoorthi, “Classification of Web Log Data to Identify Interested Users Using Decision Trees”, 2010.

[8] Qimin Cao and YinrongQiao, “Machine Learning to Detect Anomalies in Web Log Analysis,” in 2017 3rd IEEE International Conference on Computer and Communications.

e-ISSN: 2395-0056 p-ISSN: 2395-0072

International Research Journal of Engineering and Technology (IRJET)

(An ISO 9001 : 2008 Certified Journal)

Is hereby awarding this certificate to

Kapil Patel

In recognition the publication of the manuscript entitled

SQL Injection and HTTP Flood DDOS Attack Detection and Classification Based on Log Data

published in our Journal Volume 9 Issue 5 May 2022

Editor in Chief

E-mail : editor@irjet.net

Impact Factor : 7.529

www.irjet.net

International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 09 Issue: 05 | May 2022 www.irjet.net p-ISSN: 2395-0072

SQL Injection and HTTP Flood DDOS Attack Detection and Classification Based on Log Data

Kapil Patel[1], Prof. Rajni Ranjan Singh Makwana[2]

[1]B. Tech. Student, Dept. of Computer Science and Engineering, Madhav Institute of Technology and Science, Madhya Pradesh, India
[2]Assistant Professor, Dept. of Computer Science and Engineering, Madhav Institute of Technology and Science, Madhya Pradesh, India

---

Abstract - *Due to continuous growth in Tech. Industry and also due to COVID - 19 there is a huge increase in web servers and web-applications, almost every office work and educational applications are online, due to this there is huge increase in cyber attacks. So it is important to detect them as early as possible to stop it before it causes much damage to a web-application and data loss. Since web application generates huge number of logs data it is boring and difficult task to do analysis of log data manually by person. But log data is important to monitor as it contains computer generated records, which contain data about every activity and operations, so analyzing these can help in early detection of some types of attacks like SQL injection, DDOS attack, brute force attack, and cross-site scripting (XSS), etc. To improve old method of manually inspection of log analysis in this paper, an anomaly detection and classification model has been proposed, which can also be used for early attack detection by analyzing log data. To build the model machine learning decision tree algorithm has been used to classify the data into three categories like normal logs, SQL injected logs and DDOS attack logs. After comparing logs data with trained model it successfully classifies the logs into different categories and also detects attacks.*

Key Words: **Log data, Machine Learning, Decision Tree, SQL Injection, HTTP Flood DDOS Attack**

1. INTRODUCTION

Now a days due to improvement in cyber world and easily available tools web applications faces many suspicious activities and attacks because of script kiddies, they generally performs scanning and attacks a website using an automated vulnerability scanner tools or trying to fuzz script (code) into a parameter for SQL injection, cross-site scripting (XSS) etc. and often performs DDOS attacks to down the server working etc. In many such cases, logs on the web-server have to be monitored and analyzed to figure out what is going on. If it is a serious case and suspicious matter then requires a cyber expert for forensic investigation. Since running server generates huge amount and different types of logs data it is very difficult to monitor manually, even though it is not efficient enough to filter different logs data into different categories and also demand many hours to inspect the data.

In this paper a SQL injection and HTTP flood DDOS attack log anomaly detection and classification model have been proposed based on machine learning to classify the logs data into categories based on anomalous data present in logs and which further can be used to detect attacks. For building classification model a simple rule-based decision tree classifier has been used which is enough to meet demands and successfully classify the logs data. Decision tree algorithm comes under supervised machine learning algorithm, in which labeled training data of both cases normal and unusual situations is used for training model. To build the model logs data files is selected in the first step and then the parsing of logs components is done using parsing techniques, then labeling and encoding of components according to presence of some patterns or data presence in components is done and after that the labeled components of log is passed to the decision tree classifier to predict the type of log based on anomalous data present in log into three categories: Normal logs data, SQL injected logs data, HTTP Flood DDOS Attack logs data.

Experiments with 45897 logs data from real web-application hosted on local XAMPP server, building and testing model shows overall accuracy of 98.88% of classification and detection.

2. BACKGROUND

In this part of the paper brief introduction of HTTP flood DDOS attack , followed by SQL injection, and then followed by the introduction to the web logs is presented.

2.1 HTTP Flood DDOS Attack

DDOS-simply stands for Distributed Denial Of Service. It could be of any kind like hijacking a server, port overloading, denying internet based services etc. HTTP flood DDOS attack is an application layer volumetric attack, mainly focus on crashing the web servers and online web applications. These attacks are comparatively sophisticated, here a huge number of legitimate looking HTTP GET, or POST requests are used to flood the server in

this type of attack. This in return causes a denial of services. Figure 1 depicts working view of attack.

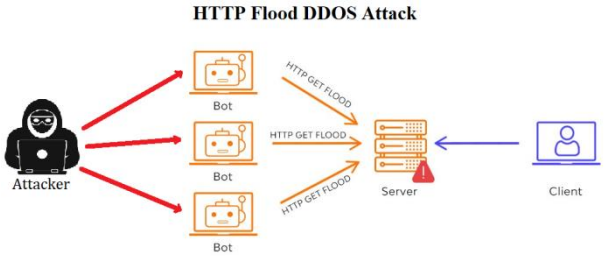


Fig -1: HTTP Flood DDOS Attack

2.1 SQL Injection Attack

SQL is a high-level scripting language which is used to store, access and maintain database systems. SQL injection is most common type of SQL attacks in which malicious SQL code is used to read, modify, and delete database information that are not allowed to access for normal users. Even it can also execute administrative operations. Because server contains huge amount of users' data which makes servers valuable target for attackers. By using SQL injection attack hackers also can bypass authentication system, compromised data and can do information disclosure. Even SQL injections can also be pivoted into remote command execution. Figure 2 depicts a view how SQL query is injected and attack is performed.

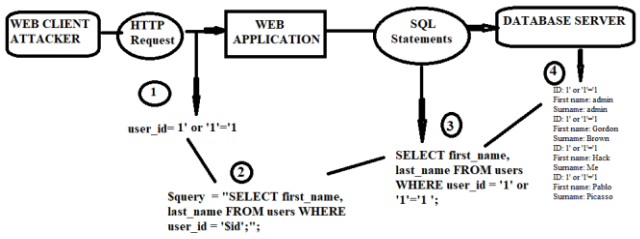


Fig -2: SQL Query Injected Attack

2.3 Web server logs

Logs are generally computer automatic generated records, which record every progress and problems during running of applications and systems. A web server log files usually contains the records of user access, requests and errors record of each time whenever user requests from web server. Mainly there are four types of log files in XAMPP Apache Server file that are: access log file, error log file, php error log file, and ssl_request log file. In this paper only access log file have taken to predict the logs type based on their content present. Apache access log file records all events that are requested, and response sent by server. Generally Apache server follows Common Log Format specification by default, so each HTTP request is written in separate line and composed of several tokens which are separated by spaces, blank values of tokens are represented by a hyphen (-). For demonstration a single log record and its specifications are given in figure 3.

192.168.169.17 - - [26/Mar/2022:21:53:29 +0530] "GET /dvwa/vulnerabilities/fi/?page=include.php HTTP/1.1" 200 4183 "http://192.168.169.107/dvwa/instructions.php" "Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0"		
Host	**192.168.169.17**	The IP address of the client.
Identity	-	The identity information reported by the client.
User	-	The user name of a successful HTTP authentication.
Date	**[26/Mar/2022:21:53:29 +0530]**	The date and time of the request.
Request	**"GET /dvwa/vulnerabilities/fi/?page=include.php HTTP/1.1"**	The request line from the client is given in double quotes.
Status	**200**	The three-digit HTTP status code generated in response to the clients request.
Bytes	**4183**	The number of bytes in the object returned to the client.
Request Header Referer	**"http://192.168.169.107/dvwa/instructions.php"**	The HTTP request header referer contains an absolute or partial address of the page that makes the request.
Request Header User Agent	**"Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0"**	The user agent identifies the application, operating system, vendor and/or version of the requesting user agent.

Fig -3: Example of a Log record

3. MODEL FRAMEWORK OVERVIEW

Figure 4 demonstrates the complete model framework overview. To detect anomaly based on web access logs data the paper mainly involves four phases: log collection, log parsing, features extraction and labeling and then training decision tree classifier and prediction of attacks.

Log collection: During run time of web-application it generates logs on every request. These valuable records could be utilized for various purposes but here used for anomaly detection, and so logs data are collected in first step for further usage. For building model data is collected by self hosting web application on local XAMPP server.

***Log parsing*:** logs are continuously written records and generally unstructured data which have to be parsed to extract useful information out of them and to make them structured. For splitting logs and extracting features out of them regular expressions are used so that information can be easily stored and manipulated.

Feature extraction and labeling: After parsing logs into separate parts, we need to further search some patterns in logs parts and based on that patterns, label them and then encode them into numerical feature arrays, whereby machine learning models can be applied.

Training decision tree classifier and prediction: Now, the feature arrays of data can be used to feed to machine

learning decision tree classifier models for training and to generate a model that can be used for prediction and classification. The trained model now can be further used to classify a new log data into a normal log, SQL injected log, DDOS attack log based on anomaly present in data.

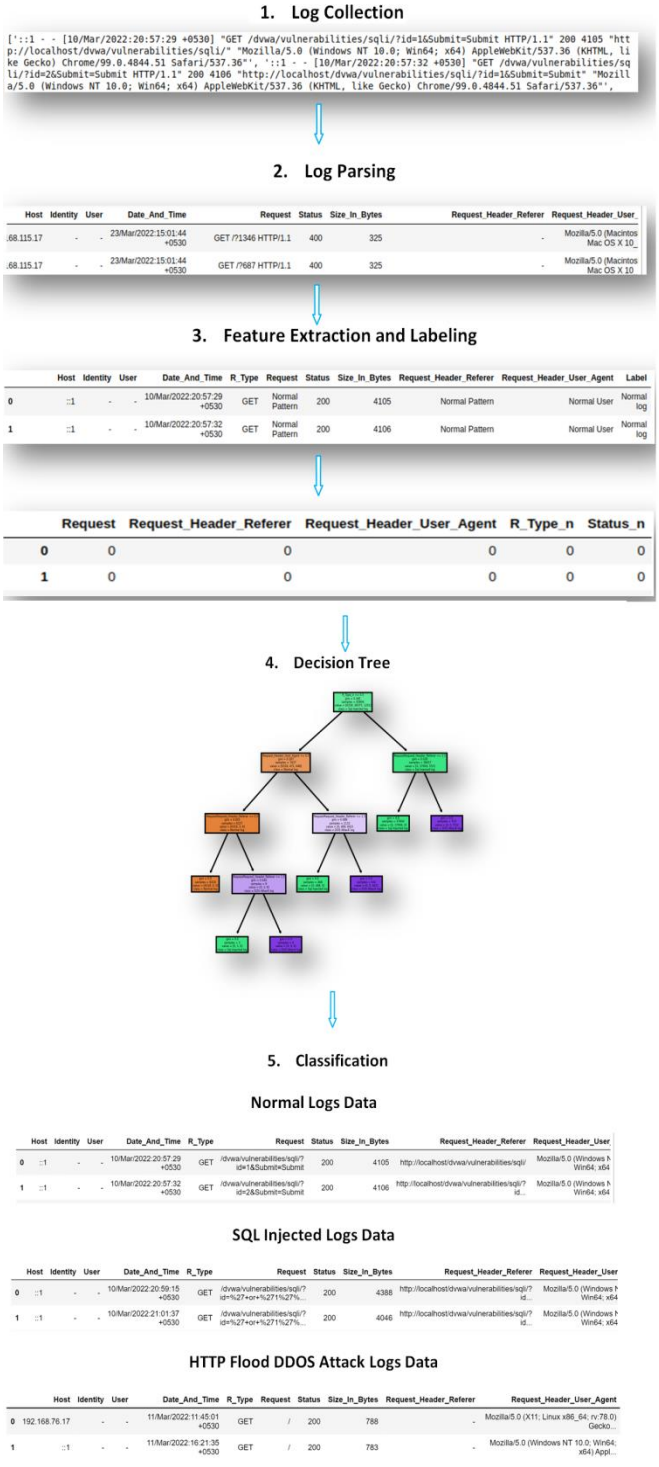


Fig -4: Anomalous Log Detection and Classification System Framework Overview

4. MODEL DESIGN AND IMPLEMENTATION

In this part of the paper detailed explanation of model framework is presented. In this section, it tells how to extract useful features and find patterns in them and label data based on that pattern. After labeling them the decision tree classifier is used to predict type of log data and to classify the log data present in file into different categories.

4.1 Data preprocessing

Features extraction: After parsing the logs file and making them structured it is important to select only important features out of them which are not much variable can provide valuable information about the log data. The Features that are taken are: HTTP status code, request type, request, request header referrer and request header user-agent which give enough information to detect presence of anomaly. HTTP request type are generally divide into two categories GET and POST. GET is used to get something from server without changing it and carries request without hiding parameter details in URL, whereas POST is used to make changes in data based on request and is more secure. HTTP status code tells whether a request has completed or not, status code has special meaning, it is generated based on responsive status of a web server. Some of the common found HTTP status code and information are listed in figure 5.

HTTP Status code	Information
200	OK
206	Created
301	Moved Permanently
302	Found
304	Not Modified
400	Bad Request
403	Forbidden
404	Not Found
414	URI Too Large

Fig -5: Some common HTTP status code

It is often found that in SQL injection attack logs data, malicious SQL query is injected into some parameters to perform the attack, so to filter out such query this model uses request, request header referer, request header user-agent of log data to filter out such injected query. To distinguish features between normal data and SQL injected data few examples of such logs are taken in figure 6 and some features are also underlined to filter out.

Fig -6: SQL Query Injected logs.

Similarly in HTTP flood DDOS attack log data also has some patterns that are common in request data, status code, request header referer, request header user-agent of log data. Few examples of such logs are taken in figure 7 and some features are also underlined to filter out.

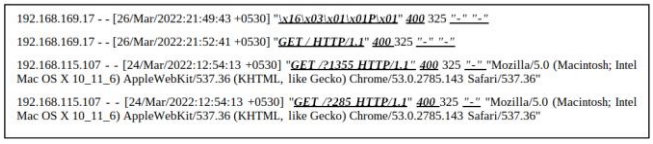


Fig -7: HTTP Flood DOS Attack logs.

Data labeling: Based on patterns found in logs data different type of labels are assigned to log data tokens. Labels taken for Labeling for each unit of data are listed in table 1.

Table -1: Labels used for each unit of data

Log Tokens	Label based on Patterns		
Request Type	GET	POST	
Request	Normal Pattern	SQL Injected Pattern	DOS Attack Pattern
Request Header Referer	Normal Pattern	SQL Injected Pattern	Empty
Request Header User Agent	Normal User	Special User	No User
Log Classification Label	Normal log	SQL Injected log	DOS Attack log

Based on collection of logs data during attack and normal situation and by analyzing each unit of data, labels have been assigned manually into three categories for classification are: normal log, SQL injected log, DOS attack log. After labeling based on these patterns, these labels have to be encoded into numerical features to train the model. To do so, I have encoded normal data as '0', SQL injected data as '1'and DOS attack data as '2'. Now after encoding these data are passed to train machine learning models.

4.2 Decision Tree Classifier

After data parsing, processing and preparing training dataset to build a model, now a suitable machine learning algorithm has to be selected which best fits for the model, to do so a simple rule-based decision tree classifier is enough to predict and classify the log data.

Decision Tree algorithm is a classification algorithm, it adds the data point to a particular labeled group on the basis of some conditions. Decision Tree graphically represents flowchart-like structure which demonstrates all the possible solutions that can be used to take a decision. Decisions are generally taken based on some conditions and which can be easily explained. Splitting of decision tree in this model is done by based on Gini impurity, which is used to split nodes when categorical data has to be predicted.

4.3 Prediction and Classification

After training decision tree classifier of model, now model is tested with new dataset to make predictions, which successfully classifies the logs dataset into the normal data set, SQL injected dataset and DOS attack dataset. After getting these datasets, inspection of each piece of logs data is done manually to make sure that there is not any false prediction, classification and labeling. After checking all datasets, this model is further used for live attack detection of SQL injection attack and HTTP flood DOS attack and found that the model is predicting perfectly.

5. EXPERIMENT

5.1 Dataset

The data has been collected by self hosting web applications on local XAMPP server and by performing HTTP flood DOS attack using pentmenu tool and slowloris script and SQL Injection by sqlmap tool and burp suite, etc. This dataset consists of three types of logs dataset during normal browse, SQL injection attack and HTTP DOS attack which is visualized in figure 8, 9, 10.

For training dataset 45897 logs data are taken from web access log file, which has data during normal running of web server and also during attack, which is enough for creating training and testing dataset for the model.

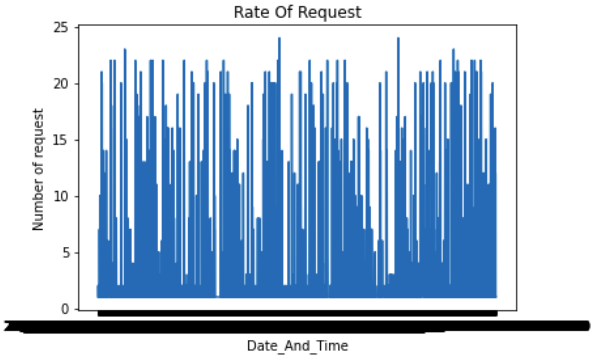


Fig -8: Logs during normal browse.

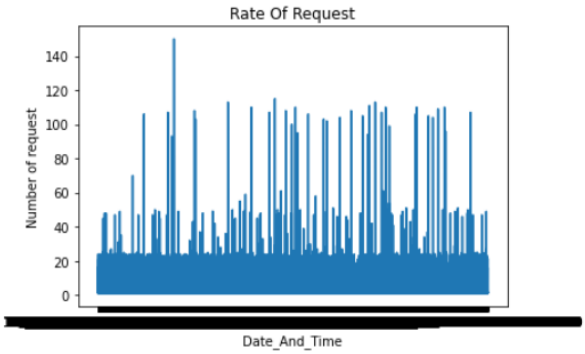


Fig -9: Logs during SQL injection attack.

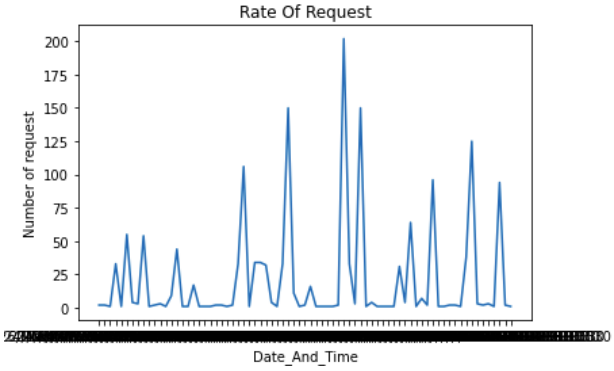


Fig -10: Logs during HTTP flood DOS attack.

5.2 Confusion Metrics:

Confusion metrics provides a holistic view of the classification model. Confusion matrix is one of the classification matrices, used to evaluate performance of classification algorithms. To examine the anomaly detection and classification system, this paper first examines the model by using confusion matrix, which describe the performance of detection and classification model in detail. Actual values: column in confusion matrix that are real values. Predicted values: rows in confusion matrix are the predictions. Based on actual and predicted logs confusion metrics for model is shown in table 2.

5.3 Results

The result calculated based on confusion matrix is presented in table 3. Accuracy tells the percentage of total number of predictions and classifications of log data that were correctly predicted and classified. Precision tells the proportion of really positive prediction and classification of logs data. Recall tells the measure of identifying true positives of predictions and classifications of logs data. F1-Score gives harmonic mean of the precision of the classifier model and recall of the classifier model. Based on these analysis it is found that the model successfully predicts and classifies the log data with (TP=45383) out of (45897) with overall accuracy of 98.88 %.

Table -2: Confusion Matrix

		Actual Label		
Predicted Label		Normal logs	SQL Injected logs	DOS Attack logs
	Normal logs	6301(TP)	18	0
	SQL Injected logs	470	36162(TP)	0
	DOS Attack logs	20	6	2920(TP)

Table -3: Result of Prediction and Classification

	Accuracy	Precision	Recall	F1-score
Normal logs	98.89%	1.0	0.93	0.96
SQL Injected logs	98.92%	0.99	1.0	0.99
DOS Attack logs	99.94%	0.99	1.0	1.0

6. RELATED WORK

Anomaly detection refers to find unusual patterns in data that do not follow regular patterns or give unexpected behavior. In past, there had been a lot of anomaly detection models were proposed. In 2004, where a Decision Tree model was applied to find error detection for web request log system in [1].In 2010 based on web log data K. R. Suneetha, R. Krishnamoorth build a classification model to identify interested users using decision trees in [2]. Similarly in 2017 based on web access log data a machine learning model is applied by Qimin Cao and Yinrong Qiao to detect anomalies in web log file in [3]. Influenced by these papers, this paper

adopted a more enhanced filtering and classifying supervised machine learning model with simple text analysis and pattern matching approach that can be used to detect and classify both normal logs data and anomaly logs data present into categories.

7. CONCLUSIONS

This paper describes a detection and classification model based on supervised machine learning for web server access log file. The model is able to analyze, detect and classify the log data based on anomaly present in access log data. The data for building model is collected data from real web-applications which has been hosted on local XAMPP server and performed different SQL injection attacks and DOS attacks on that web server. In this paper first extraction of valuable features have been done, based on some patterns and presence of some features and then labeling and encoding of that features have been presented, based on prepared dataset a decision tree classifier has been built and trained. Based on results it is found that this model successfully classifies web access log file data into three categories normal log file, SQL injected log file, and DOS log file. After testing this model, it was found that it achieved overall 98.88 % accuracy of detection and classification.

REFERENCES

[1] M. Y. Chen, A. X. Zheng, J. Lloyd, M. I. Jordan, and E. A. Brewer, "Failure diagnosis using decision trees," in 1st International Conference on Autonomic Computing), 2004, pp.36–43.

[2] K. R. Suneetha, R. Krishnamoorthi, "Classification of Web Log Data to Identify Interested Users Using Decision Trees", 2010.

[3] Qimin Cao and Yinrong Qiao, "Machine Learning to Detect Anomalies in Web Log Analysis," in 2017 3rd IEEE International Conference on Computer and Communications.