Project Report

on

# Threat detection and neutralization

Submitted as a part of course curriculum for

**Bachelor of Technology**

in

## Computer Science



**Submitted by**

SHIKHAR RAJ (2000290120144)

SHUBHI (2000290120160)

ABHIJEET KANNAUJIA (2000290120007)

**Under the Supervision of**

Prof. SREESH GAUR

## KIET Group of Institutions, Ghaziabad Department of Computer Science

**Dr. A.P.J. Abdul Kalam Technical University**

**2023-2024**

## TABLE OF CONTENTS

|  |  |
| --- | --- |
| DECLARATION……………………………………………………………………. | i. |
| CERTIFICATE……………………………………………………………………… | ii. |
| ACKNOWLEDGEMENT…………………………………………………………. | iii. |
| ABSTRACT……………………………………………………………………….... | iv. |
| LIST OF FIGURES…………………………………………………………………. | v. |
| LIST OF TABLES…………………………………………………………………… | vi. |
| LIST OF ABBREVIATIONS………………………………………………………. | vii. |
| CHAPTER 1 INTRODUCTION  1.1 Introduction to Project  1.2 Project Category    CHAPTER 2 LITERATURE REVIEW  2.1 Literature Review  2.2 Problem Formulation  2.3 Objectives    CHAPTER 3 PROPOSED SYSTEM |  |

3.1 Proposed System

3.2 Unique Features of The System (Difference from Existing System) 3.3 Structure of Report

CHAPTER 4 (REQUIREMENT ANALYSIS AND SYSTEM SPECIFICATION)

4.1 Feasibility Study (Technical, Economical, Operational)

4.2 Software Requirement Specification Document Which Must Include the Following:

(Data Requirement, Functional Requirement, Performance Requirement,

Maintainability Requirement, Security Requirement)

4.3 SDLC Model to Be Used

4.3.1 System Design Using DFD Level 0 And Level 1

4.3.2 Use Case Diagram

4.3.3 Database Design

4.3.4 ER Diagram

CHAPTER 5 IMPLEMENTATION

5.1 Introduction to Languages, Tools and Technologies Used for Implementation.

CHAPTER 6 TESTING, AND MAINTENANCE

6.1 Testing Techniques and Test Cases Used

CHAPTER 7 (RESULTS AND DISCUSSIONS)

7.1 User Interface Representation (Of Respective Project)

7.2 Brief Description of Various Modules of The System

7.3 Snapshots of System with Brief Detail of Each

7.4 Back Ends Representation (Database to Be Used)

7.5 Snapshots of Database Tables with Brief Description

CHAPTER 8 CONCLUSION REFERENCES

CONCLUSION

REFERENCES (Only **in IEEE Style**)

## DECLARATION

We hereby declare that this submission is our work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text.

Signature of Students

Name: Shikhar raj Shubhi Abhijeet Kannaujia

Roll No.: 2000290120144 2000290120160 2000290120007

Date:

## CERTIFICATE

This is to certify that Project Report entitled “**THREAT DETECTION AND NEUTRALIAZATION**” which is submitted by **SHIKHAR RAJ, SHUBHI, ABHIJEET KANNAUJIA** in partial Fulfilment of the requirement for the award of degree B. Tech. in Department of Computer Science of Dr A.P.J. Abdul Kalam Technical University, Lucknow is a record of the candidates own work carried out by them under my supervision. The matter embodied in this report is original and has not been submitted for the award of any other degree.

**Date: Supervisor**

**Signature**

Prof. Sreesh Gaur

## ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the synopsis of the B.Tech Mini Project undertaken during B.Tech. Third Year. We owe a special debt of gratitude to Prof. Sreesh Gaur, Department of Computer Science, KIET Group of Institutions, Delhi- NCR, Ghaziabad, for her constant support and guidance throughout the course of our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only his cognizant efforts that our endeavours have seen the light of the day.

We also take the opportunity to acknowledge the contribution of Dr. Ajay Kumar Shrivastava, Head of the Department of Computer Science, KIET Group of Institutions, Delhi- NCR, Ghaziabad, for his full support and assistance during the development of the project. We also do not like to miss the opportunity to acknowledge the contribution of all the faculty members of the department for their kind assistance and cooperation during the development of our project.

Last but not the least, we acknowledge our friends for their contribution to the completion of the project.

Signature:

Date :

Name : Shikhar raj Shubhi Abhijeet kannaujia

Roll No: 2000290120144 2000290120160 2000290120007

## ABSTRACT

The network size and related data have greatly increased as a result of the quick developments in the internet and communication areas. As a result, a lot of new assaults are created, which has made it difficult for network security to precisely identify breaches. Additionally, the fact that the invaders are there cannot be disregarded with the intention of launching numerous attacks within the network. Strong Intrusion Detection Systems (IDS) are now essential due to the quick spread of networked systems and the growing sophistication of cyberattacks. This work presents a sophisticated intrusion detection system that can dynamically classify the type of attack in real time in addition to detecting possible incursions. Utilizing machine learning techniques, the system improves its capacity to adjust to changing cyberthreats. The key innovation lies in the incorporation of machine learning classifiers, which enable the system to dynamically categorize detected intrusions into specific attack types.

The system can generalize and adjust to new and emerging threats since it has been trained on past Défense data covering a variety of assault scenarios. By employing ensemble learning, the classification accuracy is significantly improved and a strong Défense against false positives and false negatives is provided.

To validate the proposed system, extensive experiments are conducted using benchmark datasets and real-world network traffic. The results demonstrate the system's efficacy in accurately identifying and classifying a wide range of attacks, including Denial of Service (DoS), Probe, R2L etc.

By offering a dynamic and flexible defence mechanism, the intelligent intrusion detection system described in this study represents a substantial leap in the detection of cyber threats. Modern network infrastructures can be more secured thanks to the system's ability to adapt to the constantly changing cyber threat landscape thanks to the integration of machine learning.

## LIST OF FIGURES

### 1. WORKFLOW

****

### 2. DIAGRAMS

****

****

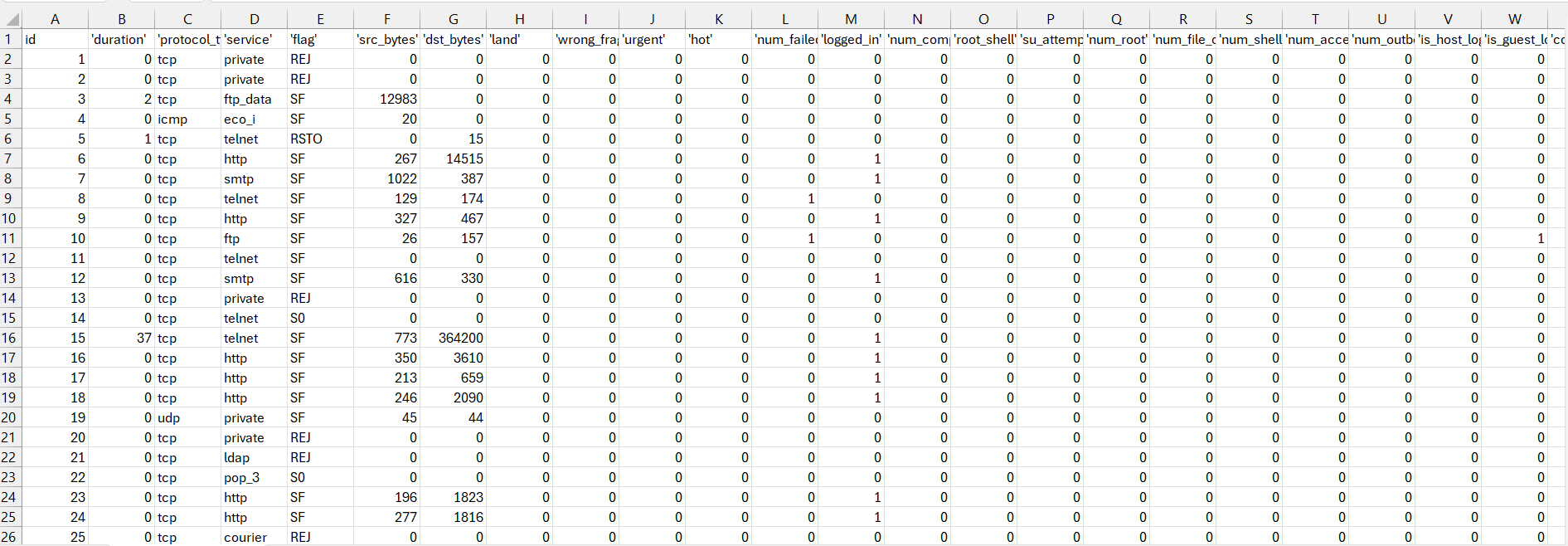
### 3. FLOWCHART

A diagram of a process

Description automatically generated

## LIST OF TABLES

### 1. DATASET TABLE



### 2. TEST CASES TABLE

A screenshot of a computer

Description automatically generated

## LIST OF ABBREVIATIONS

DOS – Denial of Service

R2L – Remote-to-Local

U2R – User-to-root

NIDS – network intrusion detection system

Keylogger

Firewall

HTTP – hypertext transfer protocol

LF – last flag

## CHAPTER 1: INTRODUCTION

### 1.1 INTRODUCTION

The increasing danger of cyberattacks is a serious threat to information network security in today's world of networked technologies and digital communication. An essential line of defence against these attacks is provided by intrusion detection systems (IDS), which are designed to recognise and react to unauthorised access, questionable activity, and possible security breaches. Traditionally, intrusion detection systems (IDS) have not delved into the finer points of attack type classification, instead concentrating on identifying predetermined attack patterns or abnormalities.

This research aims to address the limitations of existing Intrusion Detection Systems by proposing an intelligent system capable of not only detecting intrusions but also dynamically classifying the specific type of attack. One cannot stress how important it is to be able to distinguish between different types of attacks in order to respond in a focused and efficient manner. Security experts that comprehend the type of intrusion are better able to modify their mitigation tactics, use resources wisely, and strengthen network defences against certain threats. Extensive experiments with synthetic and real-world datasets are carried out to verify the efficacy of the suggested system. The outcomes of these tests show that the system can correctly detect and categorise a wide range of assaults, such as Denial of Service (DoS), Probe, R2L, and other types of malware distribution.

The use of machine learning classifiers to dynamically classify detected incursions into particular attack types is what makes this research novel. Through the examination of several feature sets derived from packet payloads, system logs, and network traffic, the system adjusts to the constantly changing threat environment. This flexibility is essential because cyber adversaries are always improving their strategies, methods, and processes.

### 1.2 PROJECT CATEGORY

The "Threat Detection and Neutralization" project can fit into various categories depending on its scope and focus. Here are several possible project categories based on different aspects:

**1: System Development**:

* **Application Development:** Create a dedicated application that monitors and detects threats in real-time. The application could have a user interface for administrators to view and respond to threats.
* **System Development**: Develop a comprehensive system that integrates threat detection mechanisms, automated responses, and reporting functionalities.

**2: Internet-Based**:

* **Web Security:** Focus on threats originating from the internet, such as malware, phishing attacks, or web-based vulnerabilities. Develop a solution that can identify and neutralize these threats.
* **Network Security**: Implement a solution that monitors network traffic for suspicious activities and responds to potential threats, ensuring the security of the connected systems.

**3: Research-Based:**

* **Threat Intelligence:** Conduct research on emerging threats, develop algorithms or models to predict potential future threats, and integrate this intelligence into the system for proactive defense.
* **Behavioral Analysis:** Research and implement techniques for analyzing the behavior of users and systems to detect anomalies and potential security threats.

**4: Industry Automation**:

* **Industrial Control Systems (ICS) Security:** Develop a system that ensures the security of critical infrastructure by monitoring and neutralizing threats to industrial control systems.
* **Automation Security:** Focus on securing automated processes within industries, ensuring that any threats are detected and neutralized to prevent disruptions.

**5: Network or System Administration:**

* **Security Administration:** Develop tools or systems that aid network and system administrators in managing security policies, monitoring for potential threats, and responding to incidents.
* **Incident Response:** Create a system that streamlines the incident response process, allowing administrators to quickly identify, contain, and neutralize security threats.

**1.3 OBJECTIVES**

1. **Improved security**: Incorporate the latest security measures to prevent unauthorized access and ensure the confidentiality of files.
2. **Real-Time Monitoring:** Use real-time monitoring technology to track and control the usage of files in real-time.
3. **Remote Computer Control:** Provide the ability to control remote computers, ensuring that security threats can be responded to in real-time.
4. **Antivirus Integration:** Integrate with existing antivirus tools to provide a more comprehensive security solution.
5. **Remote Mouse and Keyboard Access:** Provide the ability to remotely access and control the mouse and keyboard, making it possible to respond to security threats and intervene if necessary.

**1.4 PROBLEM FORMULATION**

* **The current file monitoring systems may have the following limitations or problems:**

1. Inadequate security measures that may lead to unauthorized access and usage of files.
2. Lack of real-time monitoring and control, making it difficult to track and stop malicious activity.
3. Insufficient ability to control remote computers, making it difficult to respond to security threats.
4. Limited integration with antivirus tools, leading to reduced security and protection for the system.
5. Inability to remotely access and control the mouse and keyboard, making it difficult to intervene and respond to security threats.

Our project aims to address these limitations and provide a comprehensive solution that offers improved security, real-time monitoring and control, and the ability to control remote computers, integrate with antivirus tools, and access mouse and keyboard remotely.

|  |
| --- |
| **1.5 PROPOSED SYSTEM**  **Proposed System:**  In Proposed system supervised method is used for detecting the Intrusion in the system. In order to increase the detection ability of IDS and prevent the service providers from attack, we propose an efficient ML based IDS using Light gradient boosting method and Random Forest algorithms. In order to overcome the problem of class imbalance, feature selection based on CFS-BA is used to determine a subset of the original features to eliminate irrelevant features. The detection framework of the proposed ML- Based consists of three stages including: feature selection, build and train the ensemble classifier and attack recognition. Detailed information about the framework: Feature Selection: The aim of feature selection is to find a subset of the attributes from the original set which are representative enough for the data, and the attributions in the subset are highly relevant to the prediction. Feature selection approaches can be mainly categorized into wrapper, filter and embedded approaches. While filter approaches assess the relevance of the features from the dataset and the selection of the features is based on the statistics, the classification.The performance is used in wrapper approaches as a part of the feature subsets evaluation and selection processes. In contrast to wrapper approaches, embedded approaches are computationally less intensive than wrappers because they incorporate an interaction between feature selection and learning process. Modern intrusion detection datasets inevitably contain plenty of redundant and irrelevant attributes. Redundant and irrelevant attributes can lower the efficiency of data mining algorithms, causing uninterruptable results. Therefore, the first step in this study is to reduce the dimensionality and select the feature subset of the utilized dataset. In this paper, a hybrid approach by combining CFS with BA is proposed to optimize the efficiency of the feature selection process and enhance the accuracy of the classification.  The main concept of this approach is to evaluate the relevance and the redundancy of the selected feature subset which is searched in the given search space for the optimal solution.  ● **Correlation-based feature selection (CFS):** CFS is one of classical filter algorithms that choose features according to the result of the heuristic (correlation-based) assessment function. The preference of this function is to select subsets whose features are extraordinarily related with the class but uncorrelated with each other.  **● Bat algorithm (BA):** The original bat algorithm was developed by Xin-She Yang in 2010 . The main inspirations for these works were the echolocation behavior of micro bats.  Accuracy score for gradient boosting algorithm is 65.6%  **Target variable**: The target variable of a dataset is the feature of a dataset about which you want to gain a deeper understanding. A supervised machine learning algorithm uses historical data to learn patterns and uncover relationships between other features of your dataset and the target. Here we use target variable known as ‘Has detection’. Using this target variable we can know whether the system is intruded or not. The final output which we get is in the form of 0’s and 1’s.It means that 0 indicates that system has no detections,1 indicates that the system is having intrusion for detection. Total result is obtained by using confusion matrix. The confusion matrix shows the ways in which our classification model is confused when it makes predictions.  **Advantages:**  Works on group of models make weaker models become stronger models and hence better accuracy hence it is called ensemble mod |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| **Fig 1.1 Proposed System**  **1.6 UNIQUE FEATURES OF THE SYSTEM**  To make your threat detection and neutralization system stand out, you can incorporate unique features that enhance its effectiveness, efficiency, and adaptability. Here are some potential unique features:  **1. Adversarial Learning:**  Integrate techniques from adversarial machine learning to enhance the system's resilience against adversarial attacks. This involves training the model to recognize and adapt to malicious attempts to manipulate or deceive the system.  **2. Explainability and Transparency:**  Provide transparent and interpretable explanations for the system's decisions. This can help build trust and facilitate human understanding of the reasons behind threat identifications.  **3. Automated Response Planning:**  Develop a feature that not only detects threats but also automatically generates response plans based on the severity and nature of the threat. This can include predefined actions for specific threat scenarios.  **4. Behavioral Analysis:**  Incorporate behavioral analysis to understand the normal behavior of systems, networks, or users. Deviations from established behavior patterns can trigger threat alerts, allowing for more nuanced threat detection.  **5. Contextual Awareness:**  Enhance the system's awareness by considering contextual information, such as time of day, user roles, and environmental conditions. This can reduce false positives and improve the accuracy of threat assessments.  **6. Self-Learning Mechanisms:**  Implement self-learning mechanisms that allow the system to adapt to new threats over time. This could involve continuous model retraining using incoming data or leveraging reinforcement learning techniques.  **7. Integration of Threat Intelligence Feeds:**  Connect the system to external threat intelligence feeds to stay updated on the latest threat information. This can enhance the system's ability to detect and respond to emerging threats in real-time.  **8. Human-in-the-Loop Integration:**  Include a human-in-the-loop mechanism that enables security analysts to collaborate with the system. This allows human expertise to complement machine learning capabilities, especially in complex or ambiguous situations.  **9. Privacy-Preserving Techniques:**  Integrate privacy-preserving techniques, such as federated learning or differential privacy, to ensure that sensitive data is protected while still allowing for effective threat detection.  **10. Scalability and Cloud Compatibility:**  Design the system to be scalable and compatible with cloud infrastructure. This enables easy deployment and management across diverse and dynamic environments.  **11. Real-time Threat Visualization:**  Develop a user-friendly interface that provides real-time threat visualization, allowing security analysts to quickly understand the current threat landscape and make informed decisions.  **CHAPTER 2: REQUIREMENT ANALYSIS AND SYSTEM SPECIFICATION**  **2.1: Feasibility Study:** A Feasibility Study is a preliminary study undertaken before the real work of a project starts to ascertain the likely hood of the projects success. It is an analysis of possible alternative solutions to a problem and a recommendation on the best alternative. Economic Feasibility Technical Feasibility Operational Feasibility Economic Feasibility It is defined as the process of assessing the benefits and costs associated with the development of project. A proposed system, which is both operationally and technically feasible, must be a good investment for the organization. With the proposed system the users are greatly benefited as the users can be able to detect the fakeness from the real news and are aware of most real and most fake news published in the recent years. This proposed system does not need any additional software and high system configuration. Hence the proposed system is economically feasible.   * **Technical Feasibility:** The technical feasibility in the proposed system deals with the technology used in the system. It deals with the hardware and software used in the system whether they are of latest technology or not. It happens that after a system is prepared a new technology arises and the user wants the system based on that technology. Technical Feasibility The technical feasibility infers whether the proposed system can be developed considering the technical issues like availability of the necessary technology, technical capacity, adequate response and extensibility. The project is decided to build using Python. Jupiter Note Book is designed for use in distributed environment of the internet and for the professional programmer it is easy to learn and use effectively. As the developing organization has all the resources available to build the system therefore the proposed system is technically feasible. * **Economical Feasibility:** Economic analysis is the most frequently used method for evaluating theeffectiveness of a new system. More commonly known as cost/benefit analysis. * **Operational Feasibility:** The project has been developed in such a way that it becomes very easy even for a person with little computer knowledge to operate it. This software is very user friendly and does require any technical person to operate. Thus, the project is even operationally feasible. . Operational feasibility is defined as the process of assessing the degree to which a proposed system solves business problems or takes advantage of business opportunities.   **2.2: SOFTWARE REQUIREMENT SPECIFICATION** |
| **Hardware Requirement:**   1. CPU: Intel Core i5 or higher. 2. RAM: 8GB or higher. 3. Storage: 256GB SSD or higher. 4. Network Interface: Ethernet or Wi-Fi. 5. Display: 1080p or higher.   **Software Requirement:**   1. Operating System: Windows 10 or higher. 2. Node.js: version 14.0.0 or higher. 3. MongoDB: version 4.0.0 or higher. 4. Visual Studio Code: version 1.56.0 or higher. 5. NPM: version 7.0.0 or higher.   **Communication Requirement:**   1. Ethernet: The system will use Ethernet for communication between the client and server. 2. Wi-Fi: The system will use Wi-Fi for communication between the client and server. 3. Remote Desktop Protocol (RDP): The system will use RDP for remote access and control of computers. 4. Secure Socket Layer (SSL): The system will use SSL for secure data transmission between the client and server. 5. Transmission Control Protocol/Internet Protocol (TCP/IP): The system will use TCP/IP for communication between the client and server. |
|  |
| **Functional requirements:**   1. File Monitoring: The system should be able to monitor all file access and usage on the system, including file opening, modification, and deletion. 2. Security Features: The system should have the ability to detect and alert the user of any suspicious activity, including file access from unknown sources, modifications to critical system files, and attempts to access restricted files. 3. Remote Mouse and Keyboard Control: The system should have the ability to control the mouse and keyboard of remote computers, allowing for remote access and troubleshooting. 4. RDP Functionality: The system should be built using JavaScript and Node.js, enabling easy integration with other systems and the ability to extend to RDP functionality. 5. User Management: The system should allow for the creation and management of user accounts with varying levels of access and permissions. 6. Reporting: The system should have the ability to generate detailed reports on file access and usage, security events, and remote access activity.               **Non-functional requirements:**   1. Performance: The system should be able to handle high volumes of file access and usage with minimal impact on system performance. 2. Security: The system should be designed with security in mind, including secure data transmission, encryption, and secure user authentication. 3. Usability: The system should be user-friendly, with a simple and intuitive interface. 4. Reliability: The system should be reliable and available at all times, with minimal downtime and maintenance. 5. Scalability: The system should be designed to handle a growing number of users and systems. 6. Compatibility: The system should be compatible with a wide range of operating systems, file types, and network configurations. |

**2.3 SDLC MODEL TO BE USED**

**Structured Phases:** The Waterfall model follows a clear and sequential structure, with defined phases for requirements, design, implementation, testing, deployment, and maintenance. This structured approach aligns well with projects that have well-defined and stable requirements.

**Thorough Planning Emphasis:** Waterfall places a strong emphasis on meticulous planning and extensive documentation in the initial stages. This proves critical for your project, where a comprehensive understanding of vital components such as the chat dashboard, user authentication, and location tracking features is imperative.

**Detailed Requirements Management:** The Waterfall model excels when requirements are firmly established and less likely to undergo significant alterations throughout the project lifecycle. For functionalities like the intrusion tracking in our case, detailed specifications that undergo infrequent revisions are pivotal.

**Systematic Testing Approach:** Waterfall's phased approach allows for structured testing after the development phase concludes. This is particularly beneficial for your project, ensuring that individual components like user authentication undergo rigorous testing before progressing to subsequent phases (operational network intrusion system).

### 3.3 STRUCTURE OF THE REPORT

Chapter 1: Introduction

1.1 Introduction to Project

1.2 Project Category

Chapter 2: Literature Review

2.1 Literature Review

2.2 Problem Formulation

2.3 Objectives

Chapter 3: Proposed System

3.1 Proposed System

3.2 Unique Features of The System

3.3 Structure of Report (This section)

Chapter 4: Requirement Analysis and System Specification

4.1 Feasibility Study (Technical, Economical, Operational)

4.2 Software Requirement Specification Document

4.3 SDLC Model to Be Used

4.3.1 System Design Using DFD Level 0 And Level 1

4.3.2 Use Case Diagram

4.3.3 Database Design

4.3.4 ER Diagram

Chapter 5: Implementation

Chapter 6: Testing and Maintenance

Chapter 7: Results and Discussions

7.1 User Interface Representation (Of Respective Project)

7.2 Brief Description of Various Modules of The System

7.3 Snapshots of System with Brief Detail of Each

7.4 Back Ends Representation (Database to Be Used)

7.5 Snapshots of Database Tables with Brief Description

Chapter 8: Conclusion and Future Scope

References

.  **CHAPTER 3. SYSTEM DESIGN**

**3.1: DETAIL DESIGN**

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer’s goal is to produce a model or representation of an entity that will later be built. Beginning, once systemrequirement have been specified and analyzed, system design is the first of the three technical activities design, code and test that is required to build and verify software. The importance can be stated with a single word “Quality”. Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess quality. Design is the only way that we can accurately translate a customer’s view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage. The purpose of the design phase is to plan a solution of the problem specified by the requirement document. This phase is the first step in moving from the problem domain to the solution domain. In other words, starting with what is needed; design takes us toward how to satisfy the needs. The design of a system is perhaps the most critical factor affection the quality of the software; it has a major impact on the later phase, particularly testing, maintenance. The output of this phase is the design document. This document is similar to a blueprint for the solution and is used later during implementation, testing and maintenance. The design activity is often divided into two separate phases System Design and Detailed Design. System Design also called top-level design aims to identify the modules that should be in the system, the specifications of these modules, and how they interact with each other to produce the desired results. At the end of the system design all the major data structures, file formats, output formats, and the major modules in the system and their specifications are decided. During, Detailed Design, the internal logic of each of the modules specified in system design is decided. During this phase, the details of the data of a module are usually specified in a highlevel design description language, which is independent of the target language in which the software will eventually be implemented. In system design the focus is on identifying the modules, whereas during detailed design the focus is on designing the logic for each of the modules. In other works, in system design the attention is on what components are needed, while in detailed design how the components can be implemented in software is the issue. Design is concerned with identifying software components specifying relationships among components. Specifying software structure and providing blue print for the document phase. Modularity is one of the desirable properties of large systems. It implies that the system is divided into several parts. In such a manner, the interaction between parts is minimal clearly specified. During the system design activities, Developers bridge the gap between the requirements specification, produced during requirements elicitation and analysis, and the system that is delivered to the user. Design is the place where the quality is fostered in development. Software design is a process through which requirements are translated into a representation of software

**3.2 SYSTEM DESIGN USING DFD LEVEL 0 AND LEVEL 1**

A diagram of a software company

Description automatically generated

**3.3 USE CASE DIAGRAM**

A diagram of a diagram

Description automatically generated

### 3.4 SOFTWARE REQUIREMENT SPECIFICATION DOCUMENT

The Software Requirement Specification (SRS) document outlines the functional and nonfunctional requirements of the proposed system. It includes the following components:

1. Data Requirement:

Specification of data sources, formats, and quality requirements.

Identification of data preprocessing steps, including cleaning, normalization, and feature extraction.

Description of data storage and retrieval mechanisms, including database schemas.

1. Functional Requirement:

Description of system functionality, including user interactions, data processing workflows.

Specification of functional modules, their inputs, outputs, and interdependencies.

Use case scenarios and diagrams illustrating system behavior under various conditions.

1. Performance Requirement:

Definition of performance metrics, including accuracy, latency, throughput, and scalability.

Specification of performance targets and benchmarks against which the system will be evaluated.

Identification of performance optimization strategies, such as parallel processing and caching.

1. Maintainability Requirement:

Description of system architecture, design principles, and coding standards to facilitate maintenance.

Documentation of software components, APIs, and interfaces for ease of understanding and modification.

Implementation of version control, bug tracking, and testing frameworks to support ongoing maintenance activities.

1. Security Requirement:

Identification of security threats and vulnerabilities associated with the system, including data breaches, unauthorized access, and data integrity risks.

Specification of security controls, such as authentication, encryption, and access control mechanisms.

Compliance with relevant data protection regulations and industry standards to ensure the confidentiality and integrity of sensitive information.

## CHAPTER 4: IMPLEMENTATION

**4.1 INTRODUCTION TO LANGUAGES, TOOLS AND TECHNOLOGIES USED**

### FOR IMPLEMENTATION

Here's an introduction to the languages, tools, and technologies commonly used in such projects:

1. Programming Languages:

Python: Python is widely favored for its simplicity, readability, and extensive libraries for data manipulation, machine learning, and visualization. Libraries such as NumPy, Pandas, scikit-learn, and TensorFlow are commonly used in heart disease prediction projects.

1. Machine Learning Libraries:

scikit-learn: This is a widely used machine learning library in Python. It provides simple and efficient tools for data mining and data analysis and supports various algorithms for classification, regression, and clustering, which are crucial for heart disease prediction.

TensorFlow/Keras: TensorFlow is an open-source machine learning framework developed by Google. Keras, built on top of TensorFlow, offers a high-level API for building and training deep learning models, which can be beneficial for more complex predictive tasks.

XGBoost/LightGBM: These are gradient boosting frameworks known for their efficiency and accuracy in handling structured data. They are often employed in heart disease prediction projects to enhance model performance.

1. Data Processing and Visualization Tools:

NumPy and Pandas: NumPy provides support for large, multi-dimensional arrays and matrices, while Pandas offers data structures and functions for data manipulation and analysis. Together, they form the backbone of data preprocessing in Python.

Matplotlib and Seaborn: These are Python libraries for creating static, animated, and interactive visualizations in Python. They are essential for exploring and presenting data insights derived from heart disease datasets.

1. Development Environments:

Jupyter Notebooks: Jupyter Notebooks provide an interactive computing environment that allows for easy prototyping, visualization, and documentation of code. They are widely used in data science projects, including heart disease prediction, due to their versatility and ease of sharing.

Integrated Development Environments (IDEs): IDEs such as PyCharm, Visual Studio Code, and RStudio offer advanced features like debugging, code completion, and version control integration, making them suitable for developing and maintaining predictive models.

1. Database Management Systems (DBMS):

SQL and NoSQL Databases: Depending on the project requirements, SQL databases like MySQL or PostgreSQL, and NoSQL databases like MongoDB, may be used for storing and managing structured and unstructured data related to heart disease patients' records.

Libraries used:

* + Numpy
  + Pandas
  + Matplotlib
  + seaborn
  + train\_test\_split

Algorithms used:

* + K Neighbour Classifier
  + Logistic Regression
  + Random Forest Classifier
  + Support vector classifier
  + Decision tree classifier

## CHAPTER 6: TESTING AND MAINTENANCE

### 6.1 TESTING TECHNIQUES AND TEST CASES USED

1. Unit Testing:

Test Case: Test each function or method individually to ensure they produce the expected output for a given input. For example, test the data preprocessing functions to verify that they handle missing values, outliers, and feature scaling correctly.

1. Integration Testing:

Test Case: Verify the interactions between different components of the system, such as data preprocessing, feature engineering, model training, and prediction. For instance, test the integration between the data loading module and the feature selection module to confirm that selected features are passed correctly to the model.

1. Regression Testing:

Test Case: Repeatedly test the system after making changes or updates to ensure that new modifications do not introduce regressions or unintended side effects. For example, after updating the model architecture or hyperparameters, re-run validation tests to confirm that the model's performance has not deteriorated.

1. Performance Testing:

Test Case: Assess the system's performance under different conditions, such as varying dataset sizes or computational resources. Measure metrics like training time, inference time, and memory usage to identify potential bottlenecks and optimize the system for efficiency.

**A screenshot of a computer

Description automatically generated**

## CHAPTER 7: RESULTS AND DISCUSSIONS

. We used the Random Forest Classifier to train our dataset after extracting the features from WEKA. We also trained our model using KNN and Decision Trees, but the Random Forest Classifier produced the best results. Our model was trained using 65% of the KDD dataset. The trained model was evaluated on the remaining 35% of the dataset; it produced the best accuracy of 99% with a low false positive rate.

We developed a packet sniffer to collect real-time packets from the network for real-time analysis of the model. Next, our feature extractor script decodes the collected data to extract the necessary features and stores it in a CSV file. Finally, our trained model evaluates the real-time data and groups the packets into five categories: Normal, DOS, U2R, R2L, and Probe.

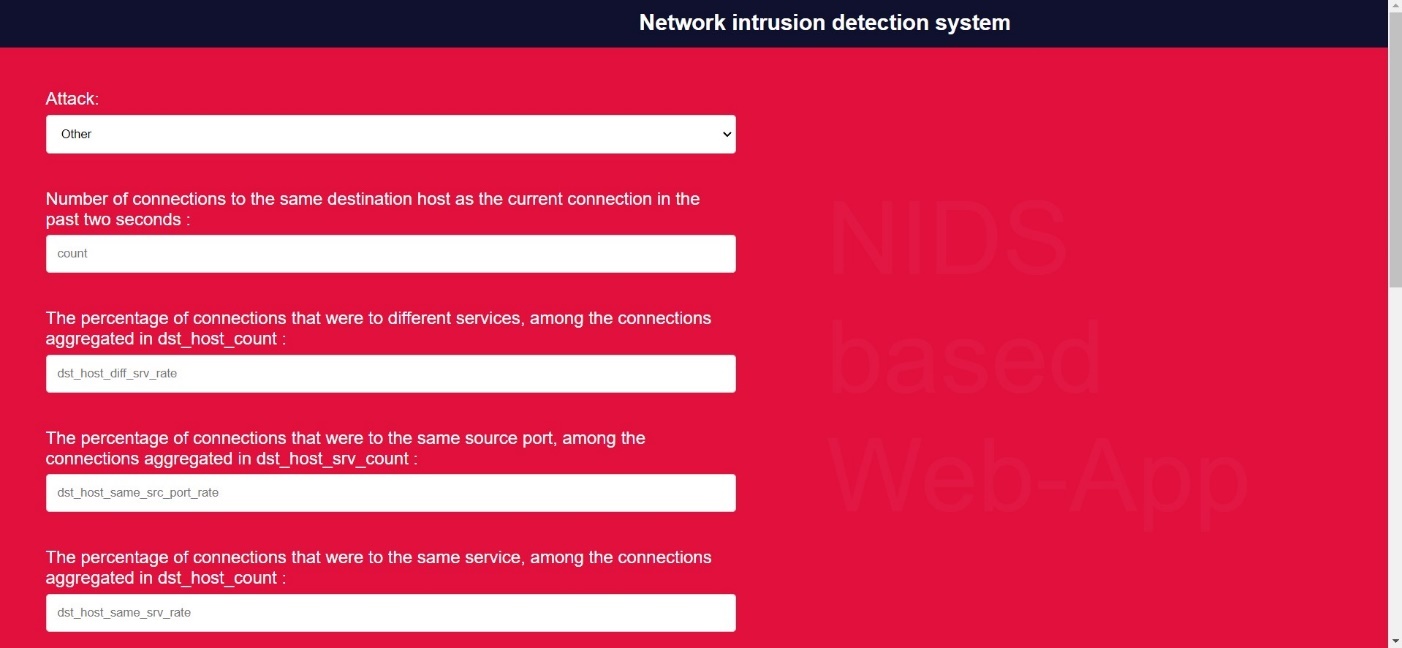
The code, as it is, will create a heatmap that visually represents the confusion matrix, making it easier to understand how well the Bernoulli Naive Bayes classifier's predictions align with the actual labels. It helps in evaluating the classifier's performance in terms of true positives, true negatives, false positives, and false negatives.

In summary, the code demonstrates the use of the SGD Classifier, an efficient and scalable classifier, for training a linear SVM model using hinge loss and L2 regularization. It then evaluates the model's performance by calculating accuracy and exploring the effect of changing the number of iterations on the model's performance.

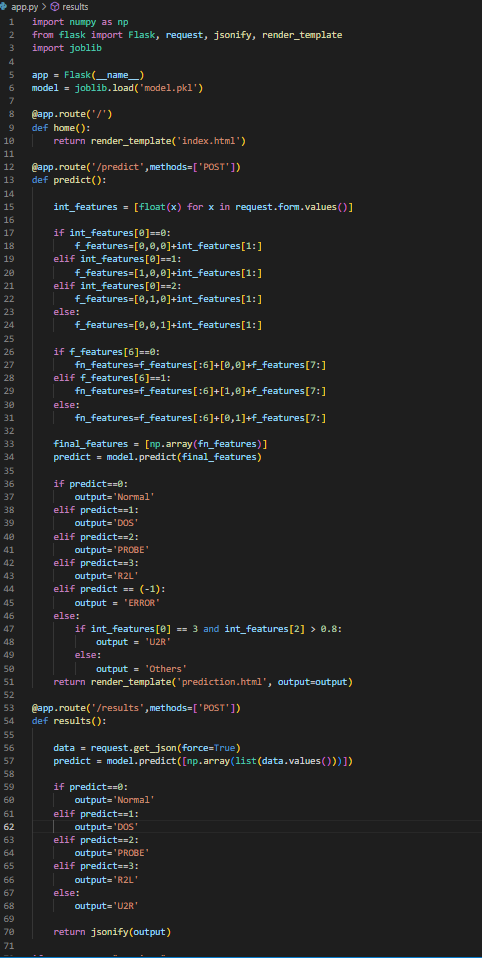
### 7.2 VARIOUS MODULES OF THE SYSTEM

1. Feature Selection module - Feature selection is a process of identifying relevant characteristics and eliminating unnecessary ones to accurately characterize a problem. This approach enhances machine learning algorithms' performance, helps understand data, reduces storage requirements, and potentially saves costs. It also offers simplicity, allowing for more basic models and acceleration. Overall, feature selection is a crucial aspect of machine learning.
2. Data Preprocessing module -The data consists of 43 columns with various attributes like duration, protocol type, service, and flag. The data is identified using column names and summarizes each numerical feature in the num\_summary file. Outlier capping is applied to restrict extreme values to the first and 99th percentiles to address outliers and enhance model resilience. This process helps maintain model resilience.
3. Model Selection module- The Random Forest Classifier is an ensemble machine learning model that can handle both classification and regression tasks. It is built using a series of steps in the code. The train\_test\_split function from Scikit-Learn is used to divide the data into training and testing sets. The hyperparameters are used to generate the model, which is imported from Scikit-Learn. Using the fit() method, the model is trained on the training set of data, gaining the ability to recognize patterns in the data and generate predictions. Following training, the accuracy\_score function is used to evaluate the model's performance on the testing set of data. This guarantees that the model can efficiently handle complicated datasets. It uses 70% for the training and remaining 30 percent for testing.
4. Intrusion Detection module - The finalized ML model is then used to predict whether the packet is regular or attack type. Basically the attack classification is based on normal , dos , probe , U2r and R2L.

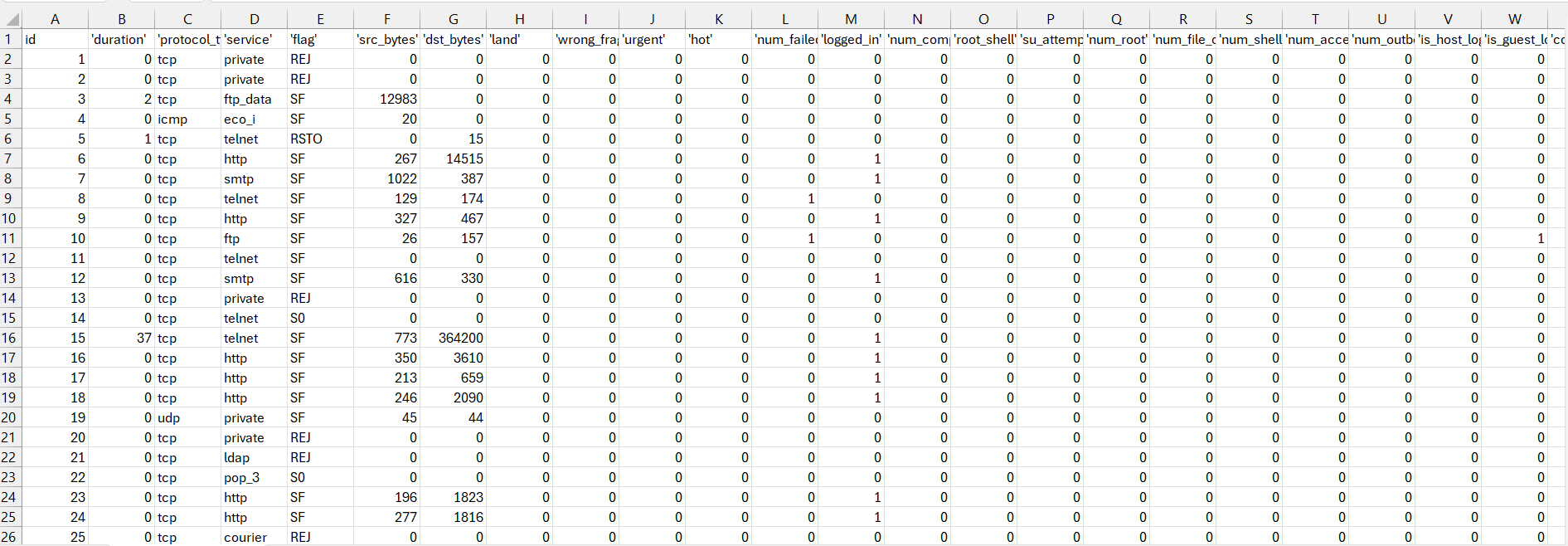
### 7.3 SNAPSHOTS OF SYSTEM



### 7.4 BACK END REPRESENTATION (CODE)

****

### 7.5 SNAPSHOTS OF DATABASE TABLE



## CHAPTER 8: CONCLUSION AND REFERENCES

### CONCLUSION

Our network intrusion detector model, which has an accuracy of 99% and can be used in an organization for security purposes, can be used here to predict the legitimacy of the packets because we live in a digital age where we frequently use the Internet and connect to networks with ports and services. As a result, there is a high risk of being attacked by a hacker who uses these kinds of attacks to gain access to monitors.

We have presented an overview of IDS detection techniques, strategies, and technologies. Every technique has its advantages and disadvantages, so we should exercise caution when choosing the methods. Consider the pattern-based intrusion detection system (IDS). While it is easy to use and highly efficient in monitoring known attacks, it is not very good at identifying unknown attacks, attacks that are hidden by evasion techniques, and numerous variations of known attacks. Also, a number of rule-based strategies have been put forth to identify unknown attacks. These methods, however, could lead to the issue of hardening and updating the knowledge for given attacks. Heuristic-based techniques also have the advantage of not requiring prior knowledge of attacks, but their high computational complexity prevents them from performing well in real-time applications. Thus, before making any practical use, it is essential to have a thorough understanding of IDSs and application requirements. Furthermore, we suggest a more thorough analysis of IDSs. A summary of the tables and figures makes it easy to understand the overall picture. Additionally, we briefly introduce two well-known, open-source tools for IDS research.

In this work, the Random Forest (RF) method is used to identify four different kinds of attacks: DOS, probe, U2R, and R2L.Ten cross-validation applications were used for classification. The data set is subjected to feature selection in order to eliminate superfluous and pointless characteristics and reduce dimensionality. We used symmetrical uncertainty of attributes to solve the information gain issues. The NSL KDD data set is used to assess the suggested methodology. We evaluated the accuracy, DR, FAR, and MCC of our random forest modeling to those of the j48 classier. Our testing results demonstrate that our suggested strategy increases accuracy, DR, and MCC for four different types of attacks. In order to increase the classifier's accuracy even further, we will use evolutionary computing as a feature selection metric in further work

### REFERENCES

1. Bolon-Canedo, V., Sanchez-Marono, N., & Alonso-Betanzos, A. (2011). Feature selection and classification in multiple class datasets: An application to KDD Cup 99 dataset. Expert Systems with Applications, 38(5), 5947-5957.

2.Sekar, R., Guang, Y., Verma, S., & Shanbhag, T. (1999, November). A high-performance network intrusion detection system. In Proceedings of the 6th ACM Conference on Computer and Communications Security (pp. 8-17).

3.Raghunath, B. R., & Mahadeo, S. N. (2008, July). Network intrusion detection system (NIDS). In 2008 First International

Conference on Emerging Trends in Engineering and Technology (pp. 1272-1277). IEEE.

4. Modi, C. N., Patel, D. R., Patel, A., & Rajarajan, M. (2012). Integrating signature apriori based network intrusion detection system (NIDS) in cloud computing. Procedia Technology, 6, 905-912.

5. Samrin, R., & Vasumathi, D. (2017, December). Review on anomaly based network intrusion detection system. In 2017 international conference on electrical, electronics, communication, computer, and optimization techniques (ICEECCOT) (pp. 141-147). IEEE.

6. Garuba, M., Liu, C., & Fraites, D. (2008, April). Intrusion techniques: Comparative study of network intrusion detection systems. In Fifth International Conference on Information Technology: New Generations (itng 2008) (pp. 592-598). IEEE.

7.Shun, J., & Malki, H. A. (2008, October). Network intrusion detection system using neural networks. In 2008 fourth international conference on natural computation (Vol. 5, pp. 242-246). IEEE.

8. Farnaaz, N., & Jabbar, M. A. (2016). Random forest modeling for network intrusion detection system. Procedia Computer Science, 89, 213-217.

9. Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Transactions on Emerging Telecommunications Technologies, 32(1), e4150.

10. Khraisat, A., Gondal, I., Vamplew, P., & Kamruzzaman, J. (2019). Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity, 2(1), 1-22.