

**University Institute of Information Technology,**

**PMAS-Arid Agriculture University,**

**Rawalpindi Pakistan**

**Intelligent Intrusion Detection System (IIDS)**

***By***

**Talal Ahmad 18-Arid-2715**

**Mohib Younis 18-Arid-2638**

**Adil Riaz 18-Arid-2597**

***Supervisor  
Mr. Zeeshan Javed***

***Bachelor of Science in Computer Science (2018-2022)***

**The candidate confirms that the work submitted is their own and appropriate  
 credit has been given where reference has been made to the work of others**.

**DECLARATION**

We hereby declare that this software, neither whole nor as a part has been copied out from any source. It is further declared that we have developed this software documentation and accompanied report entirely on the basis of our personal efforts. If any part of this project is proved to be copied out from any source or found to be reproduction of some other. We will stand by the consequences. No Portion of the work presented has been submitted of any application for any other degree or qualification of this or any other university or institute of learning.

Mohib Younis Adil Riaz Talal Ahmad

**CERTIFICATE OF APPROVAL**

It is to certify that the final year project of BS (CS) “Intelligent Intrusion Detection System (IIDS)” was developed by **“Mohib Younis (18-Arid-2638)”, “Adil Riaz (18-Arid-2597)” and “Talal Ahmad(18-Arid-2715)”** under the supervision of “**Mr. Zeeshan Javed** ” and that in their opinion it is fully adequate in scope and quality for the degree of Bachelors of Science in Computer Science.

---------------------------------------

**Supervisor**

---------------------------------------

**External Examiner**

---------------------------------------

**Administrator UIIT**

**Executive Summary**

Traditional security mechanisms protect computers and networks from unauthorized use through access controls such as passwords. However, if these access controls can be compromised or bypassed, an attacker can gain unauthorized access, causing significant damage and disruption to system operation. An Intelligent Intrusion Detection System (IIDS) detects malicious user activity through audit trails and packet transmission on the network. An intruder is likely to exhibit a pattern of behavior that is significantly different from that of a legitimate user. By observing this statistically anomalous behavior, you can detect intruders masquerading as legitimate users. This idea is the basis for improving IIDS by monitoring system activity and detecting atypical behavior. These detection systems can detect intrusions that cannot be detected by other means, such as those that exploit unknown vulnerabilities. Additionally, all computer systems and networks have known vulnerabilities that an attacker could exploit. Detecting intrusions that exploit these known vulnerabilities using explicit expert system rules is more efficient than statistical anomaly detection.

**Acknowledgement**

All praise is to Almighty Allah who bestowed upon us a minute portion of His boundless knowledge by virtue of which we were able to accomplish this challenging task.

We are greatly indebted to our project supervisor “**Mr. Zeeshan Javed**” for personal supervision, advice, valuable guidance and completion of this project. We are deeply indebted to him for encouragement and continual help during this work.

And we are also thankful to our parents and family who have been a constant source of encouragement for us and brought us the values of honesty & hard work.

Mohib Younis Adil Riaz Talal Ahmad

**Table of Contents**

**Introduction 1**

1.1 Brief 8

1.2 Relevance of Course Modules 8

1.3 Project Background 8

1.4 Literature Review 10

1.5 Analysis from Literature Review 10

1.6 Methodology and Software Life Cycle 11

**Problem Definition 12**

[2.1 Purpose](#_Toc268523795) 12

[2.2 Product Functions](#_Toc268523796) 12

[2.3 Proposed Architecture](#_Toc268523804) 13

[2.4 Project Deliverables](#_Toc268523806) 13

[2.5 Operating Environment](#_Toc268523807) 13

[2.6 Assumptions and Dependencies](#_Toc268523808) 13

**Requirement Analysis 14**

[3.1 Functional Requirments](#_Toc268523823) 14

[3.2 Non – Functional Requirments](#_Toc268523825) 14

[3.3 Use case Model](#_Toc268523823) 15

[3.3.1 Use Case Diagarm](#_Toc268523787) 15

[3.3.2 Actors Discription](#_Toc268523787) 15

[3.3.3 Use Case Discription](#_Toc268523787) 16

**The Design 17**

[4.1 System Design](#_Toc268523830) 17

[4.1.1 UML Structural Diagrams](#_Toc268523787) 17

[4.1.1.1 Component Diagram](#_Toc268523787) 17

[4.1.1.2 Package Diagram](#_Toc268523830) 18

[4.1.1.3 Deployment Diagram](#_Toc268523830) 19

[4.1.2 UML Behavioral Diagrams](#_Toc268523787) 20

[4.1.2.1 Activity Diagram](#_Toc268523830) 21

[4.1.2 UML Interaction Diagrams](#_Toc268523787) 22

[4.1.2.1 Sequence Diagrams](#_Toc268523830) 22

[4.2 Class Diagram](#_Toc268523830) 23

**Chapter 1: Introduction**

In this chapter we will discuss over view of our whole project, its brief introduction, and how it is relevant to the courses which we have studied during our degree. We will also discuss project literature review and its analysis and methodology that we will use in project.

**1.1. Brief**

Information and communications technology (ICT) systems and networks handle various sensitive user data that are prone by various attacks from both internal and external intruders. These attacks can be manual and machine generated, diverse and are gradually advancing in obfuscations resulting in undetected data breaches. Malicious cyberattacks pose serious security issues that demand the need for a novel, flexible and more intelligent intrusion detection system (IIDS). An IDS is a proactive intrusion detection tool used to detect and classify intrusions, attacks, or violations of the security policies automatically at network-level and host-level infrastructure in a timely manner.

An IIDS may be implemented as a software application running on customer hardware or as a network security appliance. Cloud-based intrusion detection systems are also available to protect data and systems in cloud deployments.

IIDS is intended to provide a system independent mechanism for real time detection of security violations. The core IIDS processing is independent of any particular target system, application environment, level of audit data (e.g., user level or network level), system vulnerability, or type of intrusion, thereby providing a framework for a general purpose intrusion-detection system. An IIDS prototype determines whether user behavior as reported in the audit data is normal with respect to past or acceptable behavior as represented by a user’s historical profile of activity. The IIDS prototype continually updates the historical profiles over time, using the reported audit data t o learn the expected behavior of users of the target systems.

**1.2. Relevance to Course Modules**

Almost everything which we are using in our project “ Intelligent Intrusion Detection System (IIDS) ” is totally relevant to our course materials.

* The system which we are developing in our project is relevant to the subject "Computer Network ".
* The algorithm which we are developing in our project is relevant to the subject “Analysis of Algorithm ".
* The NIDS and HIDS we are using in our project is relevant to the subject “Artifical Intelligence ".

**1.3 Project Background**

Intrusion detection systems are used to detect anomalies with the aim of catching hackers before they do real damage to a network. IDS’s can be either network- or host-based. A host-based intrusion detection system is installed on the client computer, while a network-based intrusion detection system resides on the network.

Intrusion detection systems work by either looking for signatures of known attacks or deviations from normal activity. These deviations or anomalies are pushed up the stack and examined at the protocol and application layer. They can effectively detect events such as Christmas tree scans and Domain Name System (DNS) poisonings.

An IDS may be implemented as a software application running on customer hardware or as a network security appliance. Cloud-based intrusion detection systems are also available to protect data and systems in cloud deployments.

**1.4 . Literature Review**

A study (IeeeXplore [1]) shows that most infiltrations are initiated by unauthorized users named attackers. An attacker could remotely access a computer over the Internet or remotely disable a service. To accurately detect an intrusion, you need to understand how to successfully attack a system. In general, attacks can be classified into five levels. They are reconnaissance, exploitation, reinforcement, consolidation, and pillage.

An IDS study (Atlantis Press [2]) tells that an attack can be detected during the first three phases however once it reaches the fourth or fifth phase then the system will be fully compromised. Thus, it is very difficult to distinguish between a normal behavior and an attack. IDS systems have limited ability to detect attacks related to eavesdropping. 'Probe' attacks can be executed over a network or locally within a system. An attack can now be defined as a set of actions that potentially compromises confidentiality, data integrity, availability, or any kind of security policy of a resource. Basically, IDS systems aim to prevent malicious activity on your computers and networks by detecting all these types of attacks.

A study of botnet traffic detection (IeeeXplore [3]) shows that two datasets were made based on the well-known CTU-13 database. The main problem of CTU-13 is the imbalance between its classes, so they have proposed the Quasi-Balanced-CTU13 (QB-CTU13) as a selection of samples from the different classes of CTU-13 while preserving the less represented ones. The experimentation was made using two datasets QB-CTU13 and EQB-CTU13, which were built to overcome its class imbalance. The Quasi-Balanced based CTU-13 dataset (QB-CTU13) contains the same number of usable TCP flows from the three most represented botnets and all the usable TCP flows from the remaining ones, hence the name ‘Quasi-Balanced’. The Extended QB-CTU13 (EQB-CTU13) is constructed using the same TCP flows from the QB-CTU13, and TCP flows from the Stratosphere IPS of three new botnet classes: Bunitu, Miuref and NotPetya.

**1.5 . Analysis from Literature Review**

The main purpose to make this system is to secure our system from unwanted people or malicious attack . Intrusion detection systems are used to detect anomalies with the aim of catching hackers before they do real damage to a network.An IDS may be implemented as a software application running on customer hardware or as a network security appliance. Cloud-based intrusion detection systems are also available to protect data and systems in cloud deployments . An IDS may be implemented as a software application running on customer hardware or as a network security appliance. Cloud-based intrusion detection systems are also available to protect data and systems in cloud deployments

**1.6 . Software methodology:**

Agile Development Methodology:

The main focus of this methodology is the project/product itself. That is why, it presupposes various constant alterations based on users and customers feedback, as well as internal changes related to the work of engineers. Agile software development methodology is free of rigid frameworks on the one hand. While, on the other hand, the working process is divided into short time boxes, thus offering the real results and feedbacks truly fast.

**1.6.1 Rationale behind Selected Methodology**

. Improved Quality

. Improved Security

. User Satisfication

**Chapter 2: Problem Definition**

# Purpose:

IoT devices and the networks or systems to which they are connected are highly vulnerable to cyberattacks. This is because as the number of IoT devices increases, the complexity required to operate these devices increases, creating new challenges and threats related to security, privacy, and usability. Additionally, the interdependent and interconnected setup of IoT creates new attacks such as Bashlite and Mirai malware attacks launched from surveillance cameras and wireless routers that have paralyzed internet-based services, whereas traditional IDSs have low false alarm rates, low detection rates, and imbalanced data sets. and response time efficiently.

**2.2. Problem statement:**

An intrusion detection system (IDS) is a device or software application that monitors a network for dangerous activity or policy violations. Malicious activity or breaches are typically reported or collected locally using security information and event management systems. Some IDSs can respond to received signals when detected. They are classified as IPS. No unreasonable firewalls, no inaccessible networks. Attackers continue to develop new items and attack modes designed to defend their defenses. Many attacks use other malware programs or social media to obtain user credentials that grant them access to networks and data. Network Intrusion Detection System (NIDS) systems are critical to network security because they can detect and respond to malicious traffic. A key benefit of an access system is that it provides an alert to IT staff when a network attack or intrusion is possible. Network Intrusion Detection System (NIDS) systems monitor both incoming and outgoing network traffic and the intersection of data between systems within the network. Network ID monitors network traffic and traffic alerts when suspicious activity or known threats are detected, allowing IT staff to monitor closely and take appropriate action to prevent or stop the attack.

**2.3. Proposed Architecture:**

By combining both (Network Intrusion Detection System) NIDS and (Host Intrusion Detection System) HIDS collaboratively, an effective deep learning approach is proposed by modeling a deep-neural network (DNN) to detect cyberattacks proactively. The efficacy of various classical machine learning algorithms and DNNs are evaluated on various NIDS and HIDS datasets in identifying whether network traffic behavior is either normal or abnormal due to an attack that can be classified into corresponding attack categories

**2.4. Project Deliverables:**

In this part of deep learning, a model is developed as an optimization algorithm for first-order gradient-based optimization of a stochastic objective function to obtain maximum accuracy of classification ratios for intrusion detection system. This optimizer works on adaptive estimation of low-order instants. Use this method to activate a model developed as a pre-implementation process. Raw data is initially collected from CTU-13 dataset in the PCAP format and subsequently transformed to select the relevant network flow parameters.

**2.5. Operating Environment:**

**Operating System**: We use Linux Operating System to develop this system.

**Datasets :** CTU 13 and CTU-IoT datasets.

**PyCharm :** We used PyCharm Software to use Python language and deep learning . Front end will be designed using tkinter in Python .

**2.6. Assumptions and Dependencies:**

**Assumptions:**

The application developed from this technique is more efficient than others. This software gives perfect result in real time for finding the intrusions . The customer will be satisfied from any malicious attack after using this system.

**Dependencies:**

This software is dependent on internet connectivity.

Application is also dependent on proper placement of hardware.

# Chapter 3: Requirement Analysis

In this chapter we will define all the requirements of proposed system that include functional and non-functional requirements. We will also discuss about use cases of the system and see how our system will respond to various use cases.

# Functional Requirements

For our system to work and facilitate the user number of functional requirements are needed. These functional requirements also be presumed as interface requirements as they are all but interface.

* The system will authenticate the user behaviour through packet transmission on the network.
* The system is constructed with functional requirements like creating monitoring profile.
* The profiles should be uniquely local to each node.
* The system will perform at high level security .
* The system will perform in accurate time.

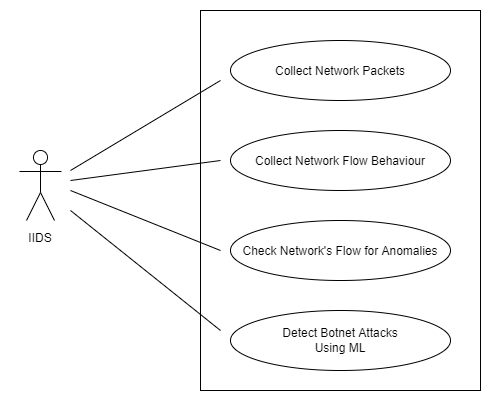
# Non-Functional Requirements

* The system should be able to handle the concurrent requests from different users.
* The system should provide confidentiality for user data.
* The system should be stable and reliable enough to handle the exceptions.
* The system should be efficient enough to handle the concurrent request to the user.
* Interface and the system itself should be user friendly so that the user will feel easy to use it.

# Use Case Model

In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. Following are the use cases of the Intelligent Intrusion Detecting System.

**3.3.1 Use Case Diagram:**

****

**Fig 3.1: Use Case Diagram**

**Actor Description:**

**IIDS:**

An intrusion detection system in which actor assign a duty that monitors a network for malicious activity. System analysis upcoming trafic then differentiate between normal & abnormal trafic. It performs an analysis of passing traffic on the entire network, and matches the traffic that is passed on the network to the datasets used to train the model.

**Use Case Description:**

|  |  |
| --- | --- |
| Use Case ID: | ID-01 |
| Use Case Name | Collect Network Packet |
| Actor: | Host,IIDS |
| Description: | Host will collect packet from that trafic coming from the network. |
| Trigger: | IDS must generate alarms when it detects intrusive activity on your network |
| Precondtions: | Preconditioning including encoding according to data presence analysis algorithm, |
| Postconditions: | Post conditions correspond to effects of executing the action on the system’s state |
| Normal Flow | IDS check the data is normal or abnormal |
| Alternative Flow | In alternative flow it check the data is data is normal is pass the packets if data is abnormal then stop |
| Exceptions: | Nill |
| Special Requirements: | None |
| Assumptions: | Nill |

**Table 3.1: Use Case Description**

# Chapter 4: Design and Architecture

# In this chapter we will discuss the design of our system.

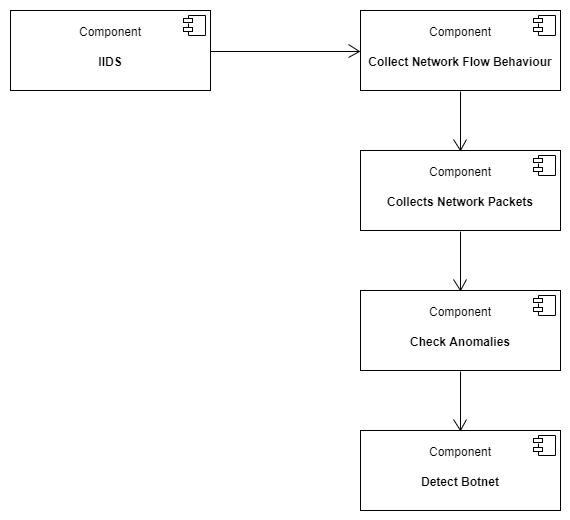
**4.1. System Design**

Systems design is the process of defining elements of a system like components, modules, architecture and their interfaces and data for a system based on the specified requirements. The purpose of the System Design process is to provide sufficient detailed data and information about the system. Following is the system design of the Intelligent Intrusion Detection System.

**4.1.1 UML Structural Diagrams**

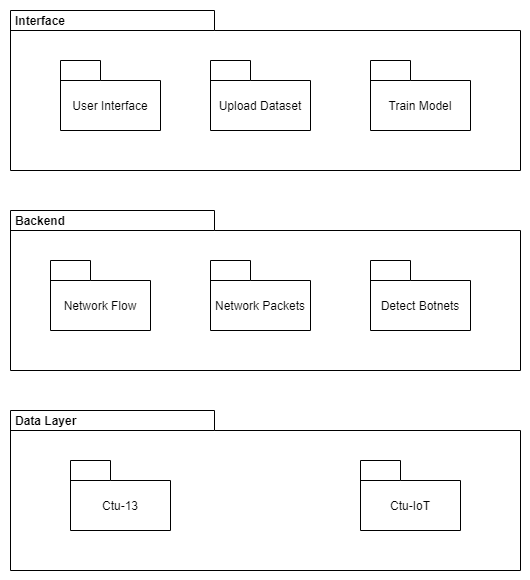
Following are the UML structural diagrams of our system:

**4.1.1.1 Component Diagram**

****

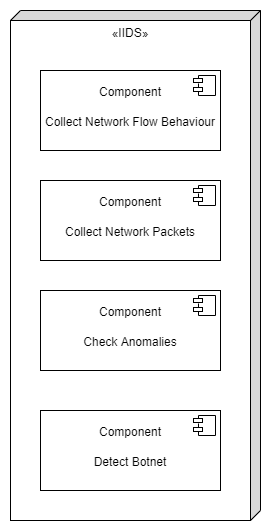
**Fig 4.1: Component Diagram**

**4.1.1.2. Package Diagram**

****

**Fig 4.2: Package Diagram**

**4.1.1.3. Deployment Diagram**

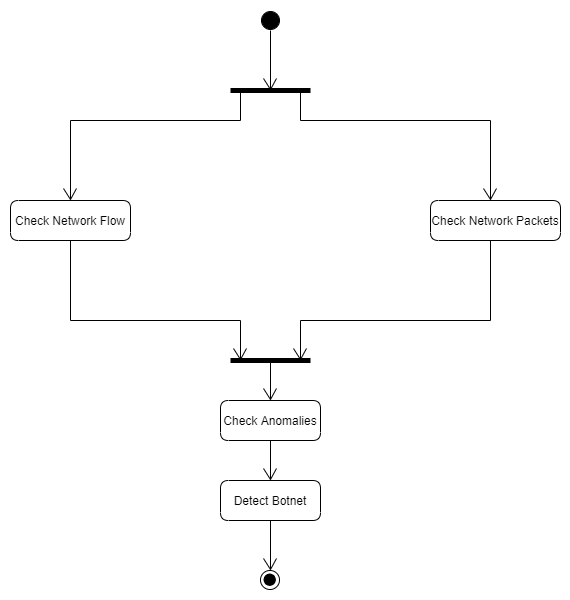
****

**Fig 4.3: Deployment Diagram**

**4.1.2 UML Behavioral Diagrams**

Following are the behavioral diagrams of our system:

**4.1.2.1 Activity Diagram**

****

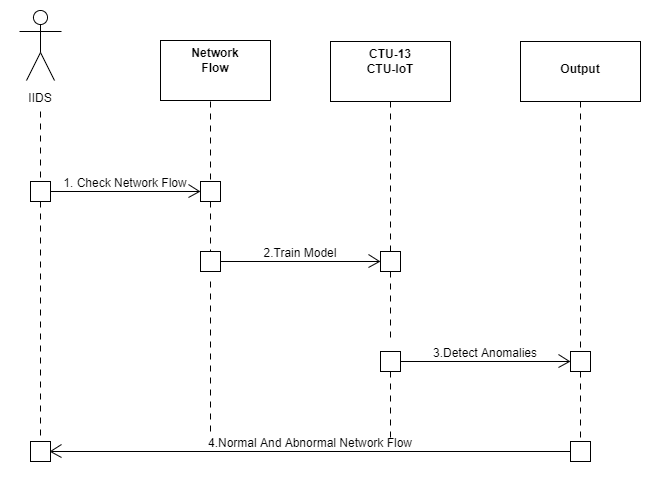
**Fig 4.4: Activity diagram**

**4.1.3 UML Interaction Diagrams**

Following are the UML interaction diagrams of our system:

**4.1.3.1 Sequence Diagrams**

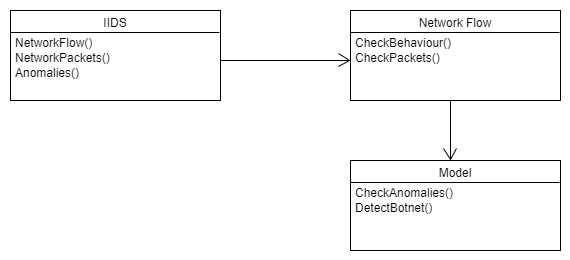
A sequence diagram is a type of interaction diagram because it describes how and in what order a group of objects works together



**Fig 4.5: Sequence Diagram**

**4.2 Class Diagram**

Class diagrams are the main building block in object-oriented modeling. They are used to show the different objects in a system, their attributes, their operations and the relationships among them.

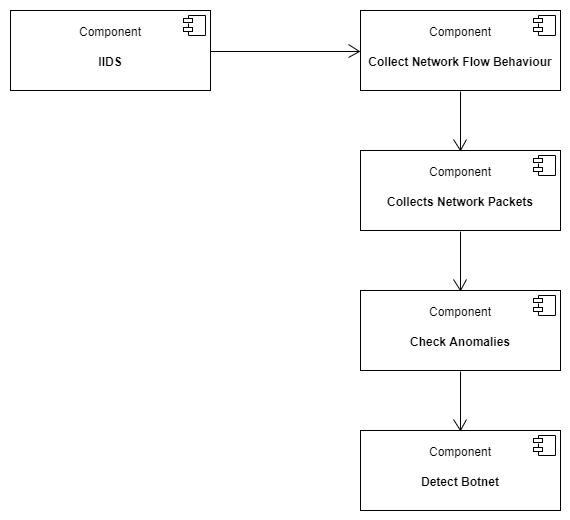


**Fig 4.6: Class Diagram**

Chapter 5: Implementation

This chapter will discuss implementation details supported by UML diagrams (if applicable). You will not put your source code here. Any of the following sections may be included based on your project.

**5.1 Component Diagram:**

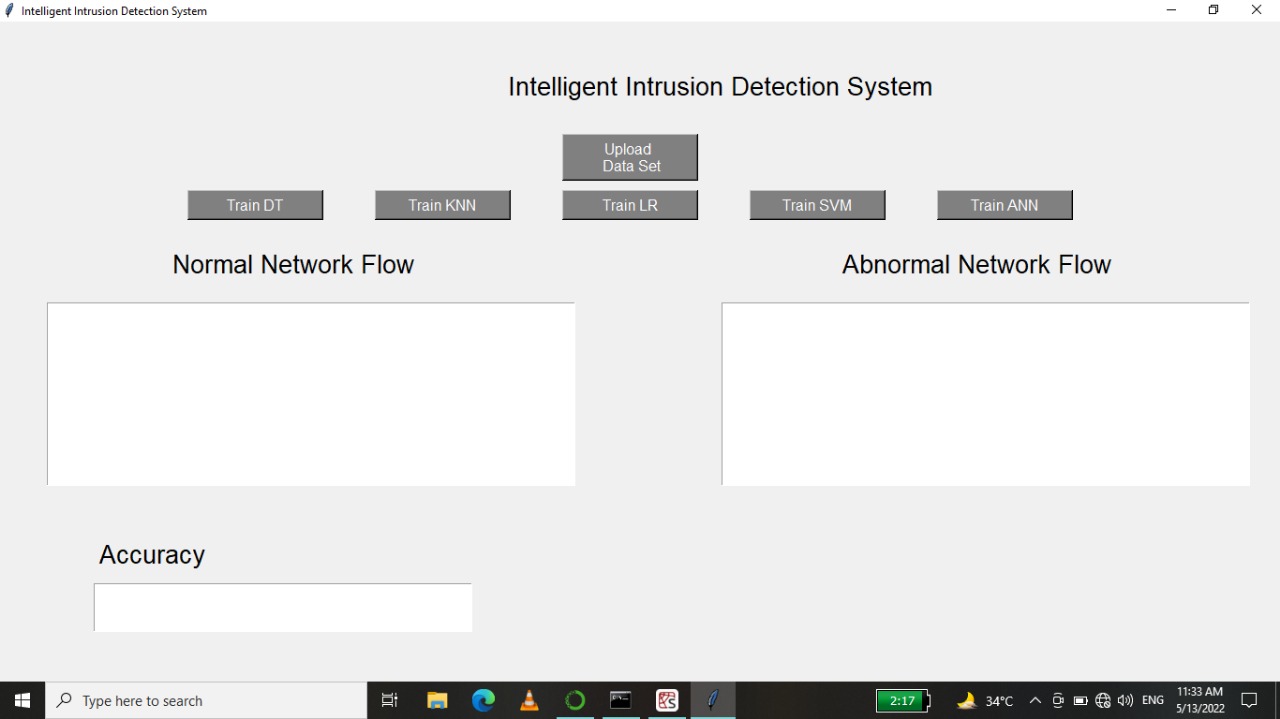


**Fig 5.1: Component Diagram**

**Explanation:**

The purpose of this project's Intelligent Intrusion Detection System is to check the network traffic and identify if the data is normal or abnormal. IIDS collects network behavior. It then collects network packets. Check for anomalies. Detect botnets.

**5.2 User Interface:**

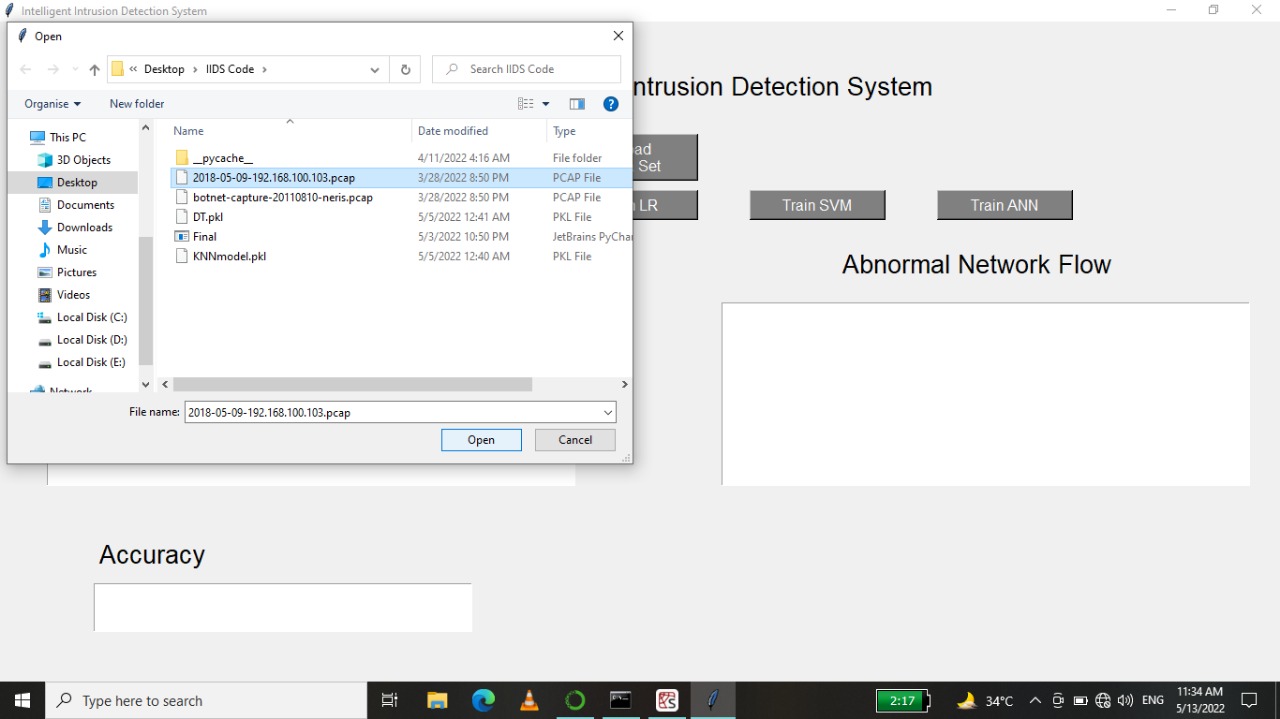


**Fig 5.2 User Interface**

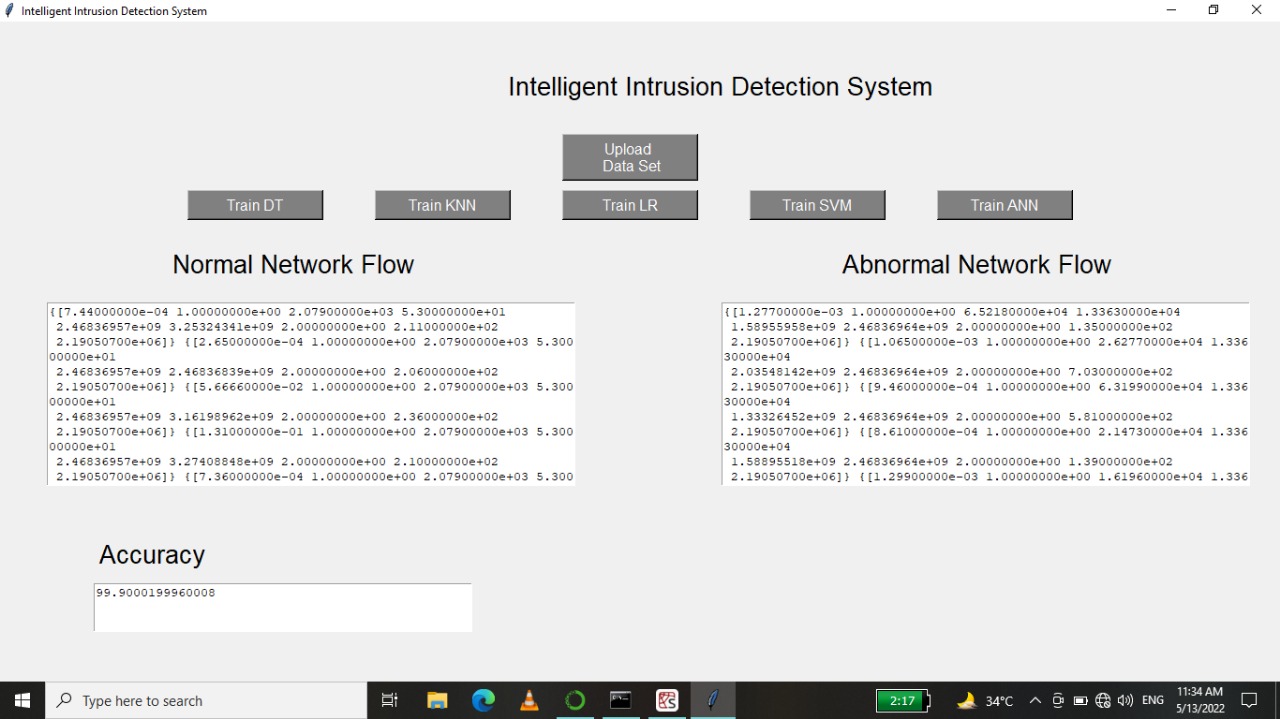
1. Upload data

2. Train Model

3. Show Normal and Abnormal Data

 **Fig 5.3: User Interface**

Here we will upload CTU IOT dataset.



**Fig 5.4: User Interface**

Show the Result Normal Network Flow and Abnormal Flow.

**Chapter 6: Testing and Evaluation**

**6.1. Verification:**

Verification is the process of checking that a software achieves its goal without any bugs. It  
is the process to ensure whether the product that is developed is right or not. It verifies whether the developed product fulfills the requirements that we have.  
  
**6.1.1. Functional Testing:**Functional Testing is a type of black box testing whereby each part of the  
system is tested against functional specification/requirements.  
  
**KNN Model**

|  |  |
| --- | --- |
| Tested By | Adil Riaz |
| Test type | Model testing |
| Test case number | 01 |
| Test case name | KNN Model |
| Test case description | This test case will Upload and train model. |
| **Procedural Steps** | **Procedural Steps** |
| 1 | User Will Upload data set |
| 2 | Then train the model |
| 3 | Show accuracy. |

**Table 6.1: KNN Model**

**DT model**

|  |  |
| --- | --- |
| Tested By | Adil Riaz |
| Test type | Model testing |
| Test case number | 01 |
| Test case name | DT Model |
| Test case description | This test case will Upload and train model. |
| **Procedural Steps** | **Procedural Steps** |
| 1 | User Will Upload data set |
| 2 | Then train the model |
| 3 | Show accuracy. |

**Table 6.2: DT Model**

**LR Model**

|  |  |
| --- | --- |
| Tested By | Mohib younis |
| Test type | Model testing |
| Test case number | 01 |
| Test case name | LR Model |
| Test case description | This test case will Upload and train model. |
| **Procedural Steps** | **Procedural Steps** |
| 1 | User Will Upload data set |
| 2 | Then train the model |
| 3 | Show accuracy. |

**Table 6.3: LR Model**

**ANN Model**

|  |  |
| --- | --- |
| Tested By | Talal Ahmed |
| Test type | Model testing |
| Test case number | 01 |
| Test case name | ANN Model |
| Test case description | This test case will Upload and train model. |
| **Procedural Steps** | **Procedural Steps** |
| 1 | User Will Upload data set |
| 2 | Then train the model |
| 3 | Show accuracy. |

**Table 6.4: ANN Model**

**SVM Model**

|  |  |
| --- | --- |
| Tested By | Talal Ahmed |
| Test type | Model testing |
| Test case number | 01 |
| Test case name | SVM Model |
| Test case description | This test case will Upload and train model. |
| **Procedural Steps** | **Procedural Steps** |
| 1 | User Will Upload data set |
| 2 | Then train the model |
| 3 | Show accuracy. |

**Table 6.5: SVM Model**

**6.2 Unit Testing:**Unit testing is a confusing part of the software development process. Unit testing involves  
individually testing unit of code separately to make sure that it works on its own, independent of the other units. Unit testing is essentially a set of paths, test performed to examine the several different paths through the modules. Unit testing is remarkably done by programmers with the help of Unit framework. Unit testing is usually an automated process and performed within the programmers IDE. Unit testing is an action used to validate that separate units of source code remains working properly. In our project we test the object detection, scheduling and creating project modules individually for better results.

**6.3 Integration Testing:  
Integration Testing** is defined as a type of testing where software modules are integrated  
logically and tested as a group. A typical software project consists of multiple software  
modules, coded by different programmers. The purpose of this level of testing is to expose  
defects in the interaction between these software modules when they are integrated. In our  
project, we integrate scheduling and creating project modules and they show results  
successfully.

# 

# Chapter 7: Conclusion and Future Work

This chapter concludes the project and highlights future work.

**7.1 Conclusion:**

Project includes four machine learnig models named KNN (K-nearest Neighbour), DT (Decision Tree), LR (Log Regression) and SVM (Straight Vector Machine) and it contains one deep learning model ANN (Artificial Neural Network).

CTU-IOT dataset is uploaded to train and test the five models.

After training every model generate pickle file containing trained data.

Models then tests the data after training and displays the normal and abnormal data on gui.

At the end it displays the accuracy of the model used to distinguish data for normal and abnormal flow.

# 7.2 Future Work

The system required some improvements which will help the system to work in more efficient way. The system that we have designed only read pcap file data. In future we tend to make our system read multiple types of datasets file.

Our system only classifies between normal and abnormal network flow. In future we want our system to classify different types of malware.