Component Design

For an Intrusion Detection System using a Neural Network

Version 1.0

Submitted in partial fulfillment of the requirements of the degree of MSE

Blake Knedler

CIS 895 – MSE Project

Kansas State University

Table of Contents

[1 Introduction 3](#_Toc469165750)

[2 Architecture 3](#_Toc469165751)

[3 Component Design 4](#_Toc469165752)

[3.1 Component Diagram 4](#_Toc469165753)

[3.2 Component Interface Specification 4](#_Toc469165754)

[4 State Diagram 6](#_Toc469165755)

[4.1 State Diagram Specification 6](#_Toc469165756)

[5 Class Design 7](#_Toc469165757)

[5.1 Class Design Diagram 7](#_Toc469165758)

[5.1.1 Network Traffic Reader 8](#_Toc469165759)

[5.1.2 Recorder 9](#_Toc469165760)

[5.1.3 Neural Network 13](#_Toc469165761)

[5.1.4 Data Reader 14](#_Toc469165762)

[5.1.5 PyIDS (UI) 16](#_Toc469165763)

[6 Sequence Design 20](#_Toc469165764)

[6.1 Operating Sequence Diagram 20](#_Toc469165765)

[6.2 Train Sequence Diagram 22](#_Toc469165766)

# Introduction

This document will provide the component design information for the PyIDS – a python interpretation of an intrusion detection system. The intrusion detection system is a single component itself but consists of several pieces that work together to perform the required functionality. This document will explain the detailed design of each of the components using the standard UML design language.

# Architecture

The Intrusion Detection System architecture is a very simple design. The architecture is a layered approach that is event driven. There are three main layers contained within the IDS. The three layers are the Network Traffic Reader, Neural Network, and Recorder. The Network Traffic Reader is a data reading layer of the system. It will take the data from the network card and package it in a way that is useful to the rest of the system. The Neural Network layer of the system is the brains of the system. It will take the data that is read in the Network Traffic Reader layer and make a decision based on backpropogation training or loaded synapse weights. The final layer is the Recorder. The Neural Network layer will communicate to this layer indicating any malicious packets it has received. It is the Recorder layer’s responsibility to log that information and notify the user. Since the system architecture is simple in nature, the Recorder also acts as the user interface. It will respond to the user when a start or train sequence is requested and notify the other layers of this information.

# Component Design

In this section, we will look at the different components of the system and focus on how they interact with each other through interfaces. We will also look at what interfaces the overall system has with external devices and users.

## Component Diagram

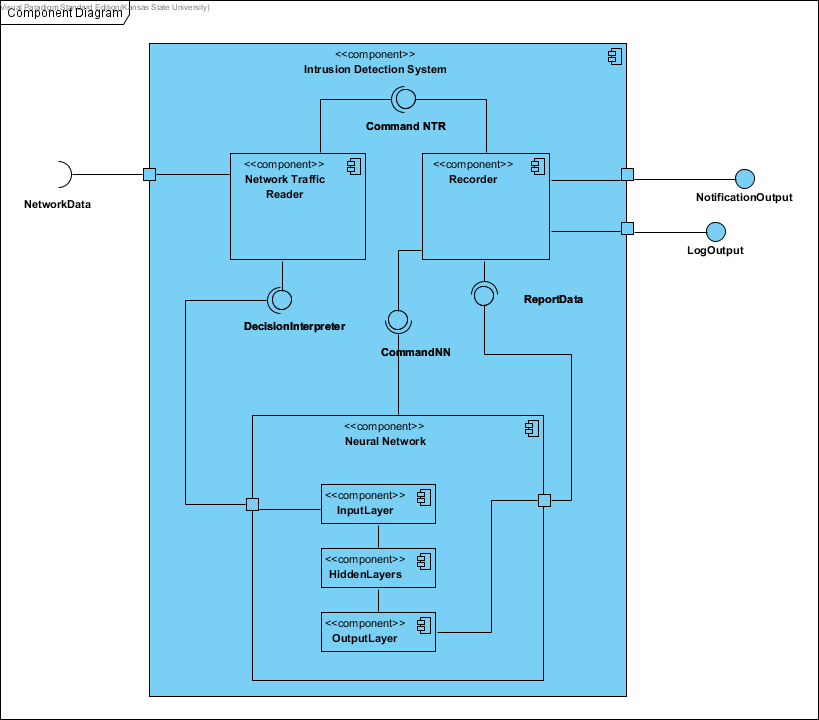


Figure 1. IDS System Component Design

## Component Interface Specification

Figure 1 shows the component diagram of the Intrusion Detection System. There are three main components contained in the overall component of the Intrusion Detection System. There are also three external interfaces to this system which are the NetworkData, which is the data traffic, the NotificationOutput and LogOutput which are notifications to the user about the data the system read. The Network Traffic Reader component is the component of the Intrusion Detection System that will read the network traffic. It will then pass this data via the DecisionInterpreter connection to the Neural Network component. This Neural Network component consists of three sub-components. These components are each of the layers of the Neural Network component. The InputLayer component will receive the data for the Neural Network component and pass the data on to the HiddenLayer and then to the OutputLayer. The OutputLayer then provides the connection of ReportData to the Recorder component. This connection will be how the Neural Network component passes any decisions of malicious data traffic to the recorder. Any non-malicious traffic can also be passed via this interface as well. The final component of this Intrusion Detection System is the Recorder component. This component is responsible for both logging and notifying the user of the data that it received.

# State Diagram

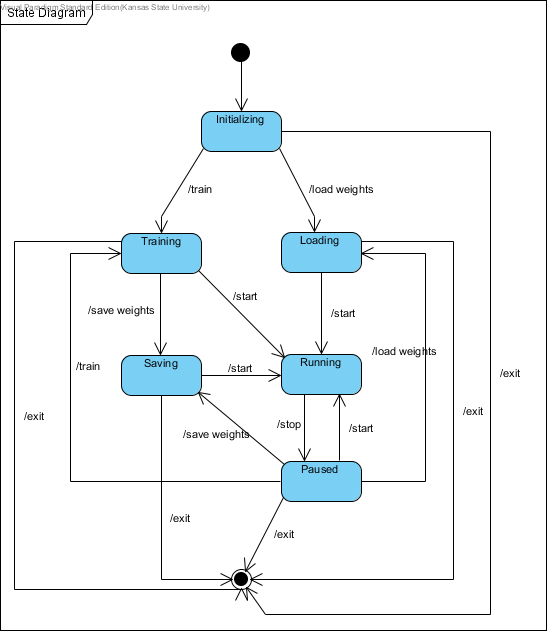


Figure 2. IDS State Diagram

## State Diagram Specification

There are six main states for the IDS system. These six states are initializing, training, loading, saving, running, and paused. The entry point into the system is initializing and the exit point can be reached from any state by exiting the application. The actions to take from one state to the next are noted in the diagram.

# Class Design

In this section, we will look briefly at a class design diagram of the three components of the system. Since this is a high level architecture document, this section will not completely detail each of these classes.

## Class Design Diagram

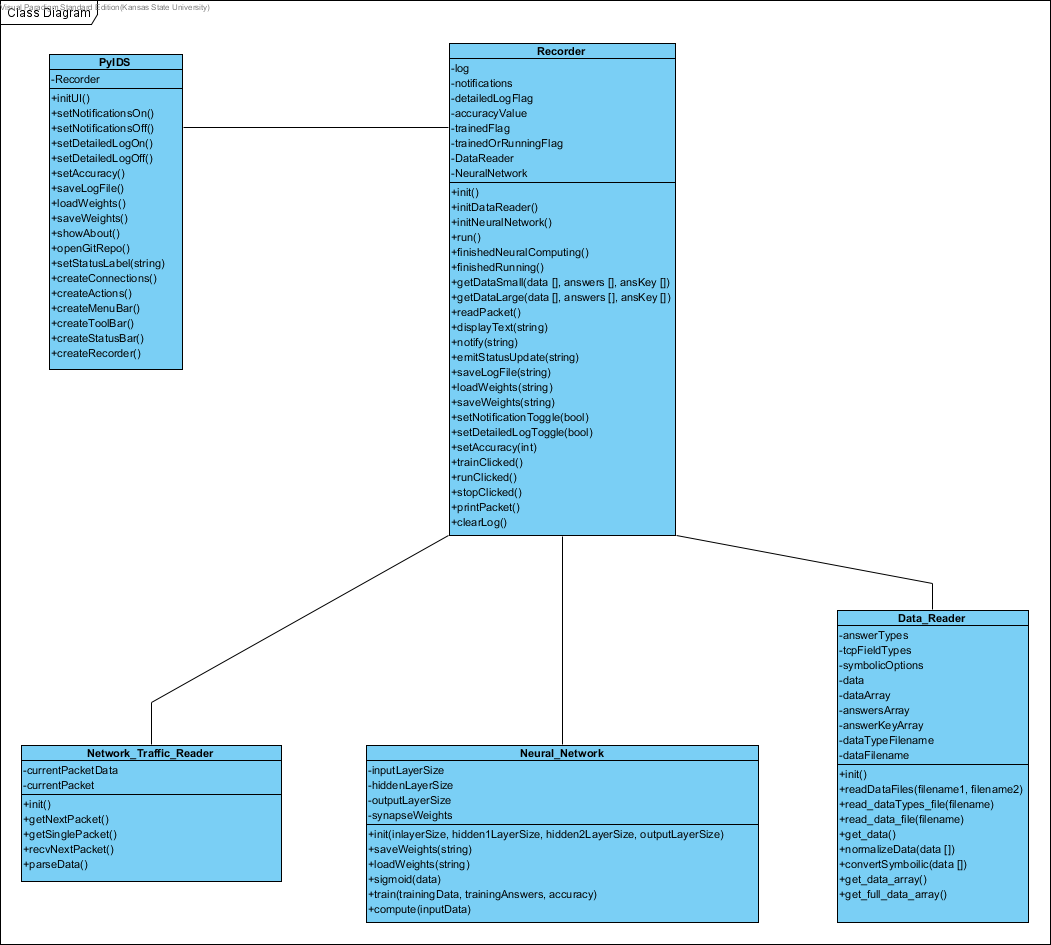


Figure 3. IDS System Class Diagrams

Figure 3 shows a class diagram of the three main components of the system. As mentioned previously, these three components are the Recorder, the Neural Network, and the Network Traffic Reader.

### Network Traffic Reader











### Recorder















































### Neural Network













### Data Reader



















### PyIDS (UI)





































# Sequence Design

In this section, we will look at a couple of the main operating sequences and how system communicates between the different internal components and also to any external user or device.

## Operating Sequence Diagram

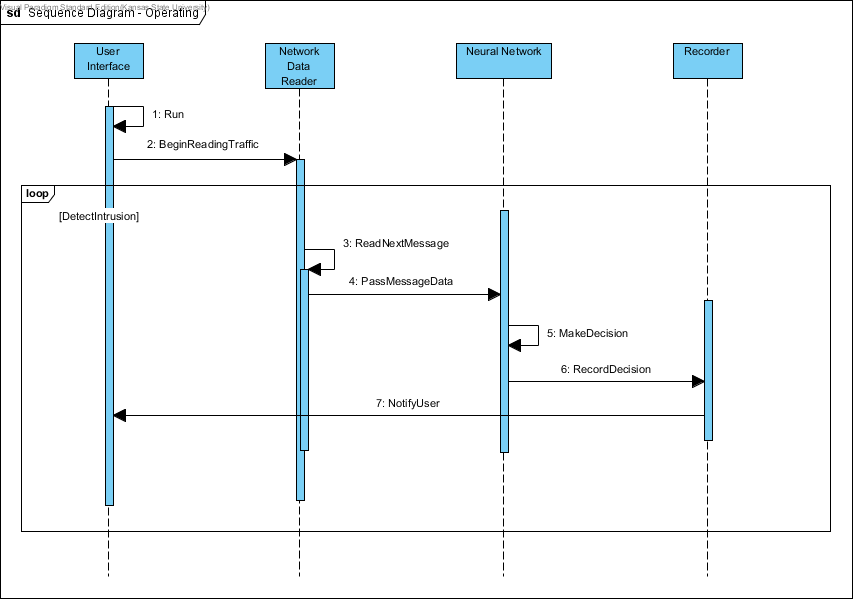


Figure 4. IDS System Operating Sequence Diagram

Figure 4 shows a sequence diagram of the main operating sequence for the Intrusion Detection System. The sequence begins with the user selecting to begin running the system. The user interface is separated in the diagram to help distinguish between it and the Recorder functionality. As mentioned previously, these two aspects will be handled by the Recorder. The User Interface notifies the Network Data Reader to begin reading network traffic. As each message is read, the Network Data Reader will pass the message data to the Neural Network. The Neural Network will then make a decision about the data packet and notify the Recorder of malicious packets. The Recorder will then notify the user by a notification system and a logging system. This process is an iterative process for each message read by the Network Data Reader.

**Name**: 1. Operating Sequence Diagram

**Description**: This use case will allow a *User* to start operating the *IDS* which will notify the *User* of any malicious packets.

**Actors**: *User*

**Stakeholders**: *User* – To start the system.

**Specializes**: None

**Includes**: None

**Extends**: None

**Triggers**: The *User* selects the start operation.

**Pre**-**condition**: *IDS* has been trained.

**Basic** **Flow**:

1. The *User* selects the start operating option of the GUI.
2. The GUI notifies the rest of the system to begin reading packets.
3. The packet reader sends packets to the Neural Network to make decisions.
4. The Neural Network notifies the Recorder of the decisions that are made.
5. The Recorder notifies the *User* when a malicious packet is found.

**Post**-**conditions**: The *IDS* is running.

**Exceptions**: None

**Constraints**: None

**Variants**: The *User* may stop the system which will temporary pause the system until “Start” is selected again.

**Comments**: Only malicious packets are notified to the *User*.

## Train Sequence Diagram

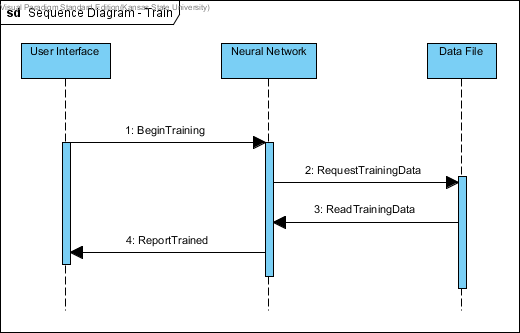


Figure 5. IDS System Train Sequence Diagram

Figure 5 shows a sequence diagram of the training scenario. This sequence begins with the user selecting to begin training. The User Interface notifies the Neural Network to begin training. The Neural Network will read the Data File containing the training data. It will then evaluate its performance of the training information and notify the user that it has now been trained and is ready to start.

**Name**: 2. Training Sequence Diagram

**Description**: This use case will train the *IDS.*

**Actors**: *User*

**Stakeholders**: *User* – To train the system.

**Specializes**: None

**Includes**: None

**Extends**: None

**Triggers**: The *User* selects the train operation.

**Pre**-**condition**: None

**Basic** **Flow**:

1. The *User* selects the train operating option of the GUI.
2. The GUI notifies the Neural Network to begin training.
3. The Neural Network read in the training data file.
4. The Neural Network begins training until the requested accuracy is met.
5. The *IDS* reports to the *User* that it has been trained.

**Post**-**conditions**: The *IDS* is trained.

**Exceptions**: None

**Constraints**: None

**Variants**: The *IDS* may not be able to reach the requested accuracy. In this case, the system will keep the last training state and report that accuracy.

**Comments**: None.