الجمهورية الشعبية الديمقراطية الجزائرية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
المدرسة العليا للإعلام الآلي - 08 ماي 1945 – بسيدي بلعباس
Ecole Supérieure en Informatique
-08 Mai 1945- Sidi Bel Abbes

Mémoire de Fin d'étude

En Vue de l'obtention du diplôme d'**ingénieur d'état**

Filière : **Informatique**

Spécialité : **Ingénierie des Systèmes Informatiques (ISI)**

Thème

Firewalls Logs Analysis Platform and Access Lists Vulnerability Detection System.

Présenté par :

- M. El MOGHERBI Mohammed Fayçal

Soutenu le : **03/07/2022**

Devant le jury composé de :

M. KHALDI Belkacem	Encadreur
M. AZZA Mohammed	Examinateur
M. KHALDI Miloud	Examinateur

Année Universitaire : 2021/2022

Acknowledgement

First of all, I want to thank God for giving me the courage and patience to carry out this work.
The following individuals deserve a special note of gratitude for their assistance and help with this master's thesis:
firstly, I want to express my gratitude to my family and friends, especially my parents, who have supported me though the good and the bad on this great journey and throughout my whole life. Words cannot express how much support they have given me and continue to provide me.
To all my framers in this thesis including, **DR. KHALDI Belkacem** who is one of the best and most kind professors i came across, who guided me in the best way to establish this thesis, **Mr. BENHARATS Abd El Rezzak**, **Mr. SAADALLAH Hamdane**, **Mr. CHOUAL Anouar** and **Mr. BENKHALFA Amine** for integrating me in the best conditions and for their kindness, generosity and all the help they gave me during my internship.
To our school's directors of studies, Drs. Amar Bensaber Djamel and Benslimane Sidi Mohammed, thank you for all of your hard work throughout the course of our five years of in-depth study.

El Mogherbi Mohammed Fayçal

Abstract

In today's business environments, information is becoming increasingly easily accessible through a large network of interconnected systems. Therefore, protecting this information and delivering it to the right recipient becomes a critical need for business security.

Firewall logs analysis reveals a lot of information about the security threat attempts at the periphery of the network and on the nature of traffic coming in and going out of the firewall. The analyzed firewall logs information, provides real-time information to the Administrators on the security threat attempts and so that they can swiftly initiate remediation action.

Firewall log monitoring plays an important role in business risk assessment. Analyzing firewall traffic logs is vital to understand network usage. Firewall Analyzer, a firewall monitoring tool, offers many features that help in collecting, analyzing and reporting on firewall logs.

Keywords : Network, log analysis, log monitoring, log collectors, machine learning, anomaly detection.

Résumé

Dans les milieux d'affaires d'aujourd'hui, l'information devient de plus en plus facilement accessible grâce à un vaste réseau de systèmes interconnectés. Par conséquent, la protection de ces informations et leurs transmission au bon destinataire devient un besoin critique pour la sécurité de l'entreprise.

L'analyse des journaux de pare-feu révèle beaucoup d'informations sur les tentatives de menace à la sécurité et sur la nature du trafic entrant et sortant du pare-feu. Ces informations une fois enregistre et analysé , fournit des informations en temps réel aux administrateurs sur les tentatives de menace à la sécurité et leur permet de lancer rapidement des mesures correctives.

La surveillance des journaux des pare-feu joue un rôle important dans l'évaluation des risques commerciaux. L'analyse des journaux du trafic est essentielle pour comprendre l'utilisation du réseau. L' analysateur des pare-feu, un outil de surveillance qui offre de nombreuses fonctionnalités qui aident à collecter, analyser et produire des rapports sur les journaux de pare-feu.

Mots clés : Réseau, analyse des journaux, surveillance des journaux, collecteurs de journaux, apprentissage automatique, détection d'anomalies.

Contents

List of Figures

List of Tables

Chapter 1

General Introduction

1.1 motivation

Rapid advancements in the internet and communication areas have resulted in a massive expansion of network size and data. As a result, numerous new attacks are being developed, making it difficult for network security to detect breaches correctly. Furthermore, intruders with the intent of launching various assaults within the network cannot be overlooked.
As result, nowadays any network security manager needs a dashboard for the real time monitoring of generated logs from the network infrastructure, with advanced analytic and visualisation tools plus the power of identifying illegal infiltration situations by using intelligent models, which will help to add a layer of safety to these networks by preserving the confidentiality, integrity, and availability.
This is where artificial intelligence meets network traffic logs analysis to introduce a particular type of systems known as intrusion detection system, Its goal is to identify computer network assaults automatically and effectively, and ensures a system's confidentiality and integrity.

1.2 Objectives

The main Objectives of this work are:

- Establish a strong base in the subject background.
- Summarize all of the related works in the field.
- Create a preferment log collector that can handle the huge traffic.
- Create a model that can properly detect anomalies with high accuracy in near real time.
- Create a web application for real time monitoring, visualisation, statistic consulting and anomaly alerting.

1.3 Limitations

The main limitations faced in this work are:

- The firewall logs are human friendly and not machine friendly, they are unstructured and different based on log type.
- The difficulty to handle traffic sent from the firewall to the log collector in near real time.
- The need of huge storage amount with a fast database, to store and retrieve the logs.
- The lack of recent public data sets on anomaly detection.
- The lack of calculation power to train the model with huge data sets.
- Time is a great limitation in this work.

1.4 Plan Of The Thesis

Background:
In this part, we'll go over all of the principles and essentials of these three domains in order to provide the groundwork for the methodologies that will be employed in the next parts. It contains three chapters which are :

- Logs analysis
- Artificial intelligence
- Network intrusion detection system

State Of The Art
In this part, we'll use the understanding ground established in the previous part to discuss about the multiple solutions for our problem and compare between them.

Our Solution

In this part i spoke about 3 main things:

- Solution design and all describing diagrams.
- Used technologies in this work and how to use them.
- Artificial intelligence model (data processing, creation, training and evaluation).
- Web application creation and demonstration.

- Collaborative work and organization of the project.

Conclusion

A conclusion of the overall study on the intrusion detection systems and log analysis.

Part I

Background

Chapter 2

Logs Analysis

2.1 Introduction

Logs generated from security systems, network devices (Firewalls) and other software are the key information to monitor the happening in these systems, they are the best way to give a meaningful representation of the system behavior. In the last decades with the massive growth in the volume of generated logs, logs analysis is becoming an imperative way to ensure the reliability and the security of any system since they are the only source available that stores critical information (software run time, access lists history, bug. . .).

2.2 Log File Definition

Log files are the primary data source for network observability. A log file is a computer-generated data file that contains information about usage patterns, activities, and operations within an operating system, application, server or another device (Logic 2022).
Log file is a historical record that took place at a certain time and might have metadata that contextualizes it containing everything and anything that happens within a system, including events such as transactions, errors and intrusions (Ltd 2021).

2.3 Types Of Logs

2.3.1 Event Logs

An event log is a high-level log that records information about network traffic and usage, such as login attempts, failed password attempts, and application events (Logic 2022).

2.3.2 Server Logs

Server logs are in the text format containing activities about a specific server in a specific period (Logic 2022).

2.3.3 System Logs

Or SYSLOG are records of the actions happening in an operating system such as warnings, errors, startup, shutdown and other changes (Logic 2022).

2.3.4 Resource Logs

Resource logs provide information about the consumption of the physical resources of a device and their connectivity and capacity limits (Logic 2022).

2.3.5 Threat Logs

Mostly generated from firewalls containing all the traffic that matches with a predefined security rule (Logic 2022).

2.4 Log Analysis Definition

Log analysis is the process of collecting, reviewing, interpreting and understand computer-generated records called logs. It is the delicate art of reviewing and interpreting these messages to gain insight into the inner workings of the system (Logic 2022).

2.5 Open Source Log Collectors

For any Intrusion detection system we need real time shipping and filtering of the network logs from the firewall to our IDS with the capacity of managing huge amount of data with minimal latency for this we need a log collector. Some log collectors are faster and others are easier to use or less resource-intensive as a result of this variety the choice of Log Collector strongly depends on the architecture and the needs of the system.

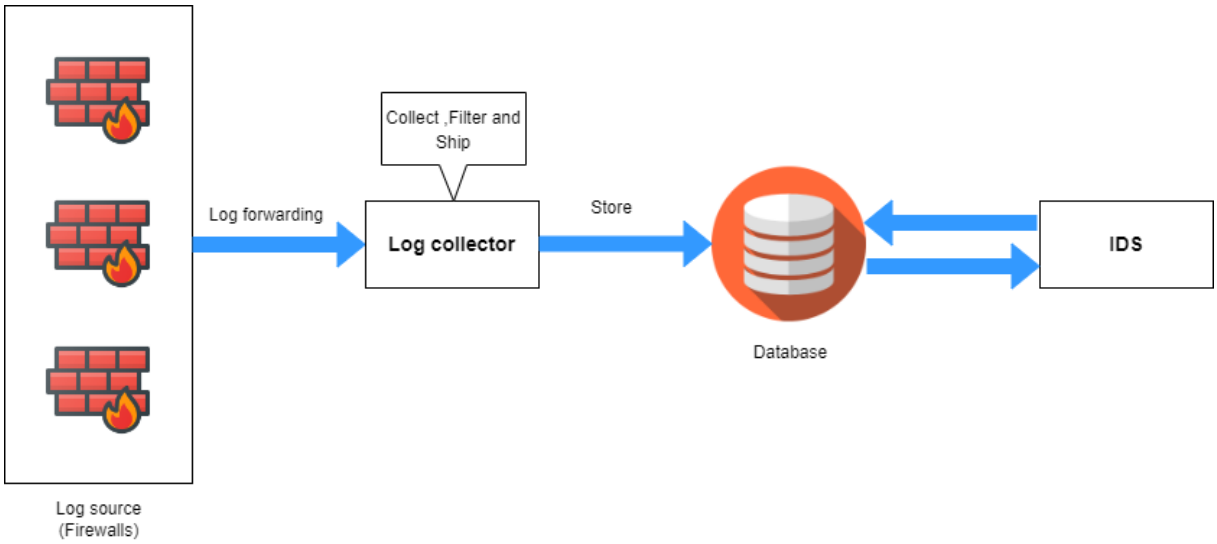


Figure 2.1: Log Collector Architecture

2.5.1 Rsyslog

Advantages :
Ultra-fast for processing logs, flexible configurations, consumes less resources. It can take entries from multiple data sources, transform them and send the output to multiple destinations (MySQL, PostgreSQL, Oracle...) (Logic 2022).

Disadvantages :
A huge lack of documentation and configuration difficulty.

Use case :
In the case where you need ultimate performance due to massive number of logs.

2.5.2 Fluentd

Advantages :
Able to collect data basically from any production system, ship it into the desired structure, create a custom pipeline and feed it to your preferred analytic platform, whether it's MongoDB or Elasticsearch. In addition, it consumes less resources (Logic 2022).

Disadvantages :
Less performant and it does not support unstructured data.

Use case :
In the case where the sources are custom applications, so working with fluentd's libraries would be easier.

2.5.3 Logagent

Advantages :
Very easy to use, simple to maintain and the setup does not take more than a few minutes. In addition to the simplicity of use Logagent can take entries from multiple data sources (Logic 2022).

Disadvantages :
New to the market and has less documentation.

Use case :
In the case where the ease of setup is the most important factor.

2.5.4 Logstash

Advantages :
Full of plugins that ensure the collection of large quantities of data from several platforms, Configuration relatively easy and a very rich documentation (Logic 2022).

Disadvantages :
Average performance and resource-intensive.

Use case :
In the case where you need a log collector, at the same time a pipeline to filter the logs with an easy configuration.

2.5.5 Filebeat

Advantages :
Simple to use and it consumes very few resources.

Disadvantages :
Limited number of output plugins.

Use case :
Generally used with Logstash and not alone due to its lack of outputs.

	Logstash	Filebeat	Logagent	Fluentd	Rsyslog
Resource consumption	high	ultra low	low	low	medium
Number of inputs (log sources)	ultra high	medium	medium	high	medium
Outputs	ultra high	ultra low	low	medium	medium
Filters	high	medium	medium	low	medium
Shipping Speed	medium	medium	medium	medium	high
Configuration Difficulty	low	medium	ultra low	high	high

Tableau 2.1: Comparative Table Between The Different Log Collectors.

2.6 Best Open Source Databases For Logging

In log analysis we are facing a huge amount of unstructured data this mean massive amount ofwhich lead as to use NoSQL databases over SQL

2.6.1 Cassandra

Originally developed by Facebook, this NoSQL database is now managed by the Apache Foundation. It's used by many organizations with large, active datasets, including Netflix, Twitter, Urban Airship, Constant Contact, Reddit, Cisco and Digg. the advantage of this database is the high availability, fault tolerance and performance with tunable consistency and elastic scalability (bitnine 2017).

2.6.2 MongoDB

MongoDB was designed to support humongous databases. It's a NoSQL database with document-oriented storage, full index support, replication and high availability (bitnine 2017).

2.6.3 CouchDB

Designed for the Web, CouchDB stores data in JSON documents that you can access via the Web or query using JavaScript. It offers distributed scaling with fault-tolerant storage (bitnine 2017).

2.6.4 OrientDB

This NoSQL database can store up to 150,000 documents per second and can load graphs in just milliseconds. It combines the flexibility of document databases with the power of graph databases, while supporting features such as ACID transactions, fast indexes (bitnine 2017).

2.6.5 Elasticsearch

ElasticSearch is a highly scalable document-oriented Database, enterprise-level and open source search engine that is based on Apache Lucene and runs in real time. Rather than following traditional text search setup, ElasticSearch provides an extended ability to make searches using query DSLs and APIs. It's very suited for the logging process, especially for searching in huge number of logs. It is known for its high performance and easy scalability (bitnine 2017).

2.7 Challenges In Log Analysis

2.7.1 Unstructured Data

Since these logs can be generated from different sources (network devices, security devices, servers, storage, etc.) and there is no defined or universal form for these files, they usually are in an unstructured or semi-structured format which differs from an operating system to another. As result centralizing log collection and processing from different sources engage a lot of big challenges.

2.7.2 Instability

(Zhang et al. 2019) acknowledged the problem of log instability. They identified a few reasons for it, such as the evolution of logging statements by modification of source code and processing of noise in log data. There is no fixed rule for a set of a distinct set of logs or several logs to be generated for any task.

2.7.3 Volume

In the last decade device intelligence, calculation speed and need for security increased exponentially. As a result the number of logs generated from each device increased also (millions/second) which caused a big data problem in collecting, processing and storing this information.

2.8 conclusion

Analysing log files for the extraction of meaningful information or the prevention of security flaws has become an obligation nowadays, although with the massive growth in the number of the generated logs the process of analysing theme is in the obligation of being automated.

Chapter 3

Artificial Intelligence

3.1 Introduction

With the appearance of big data and the calculation power of machines nowadays the need of giving machines some autonomy and intelligence is essential to exploit this data in the best ways which led to the reincarnation of machine learning methods.

3.2 Definition

Machine Learning is concerned with computer programs that automatically improve their performance through experience.
Develop systems that can automatically adapt and customize themselves to individual problems.
Discover new knowledge from large databases, mimic human and replace certain monotonous tasks which require some intelligence.

3.3 Basic steps used in Machine Learning

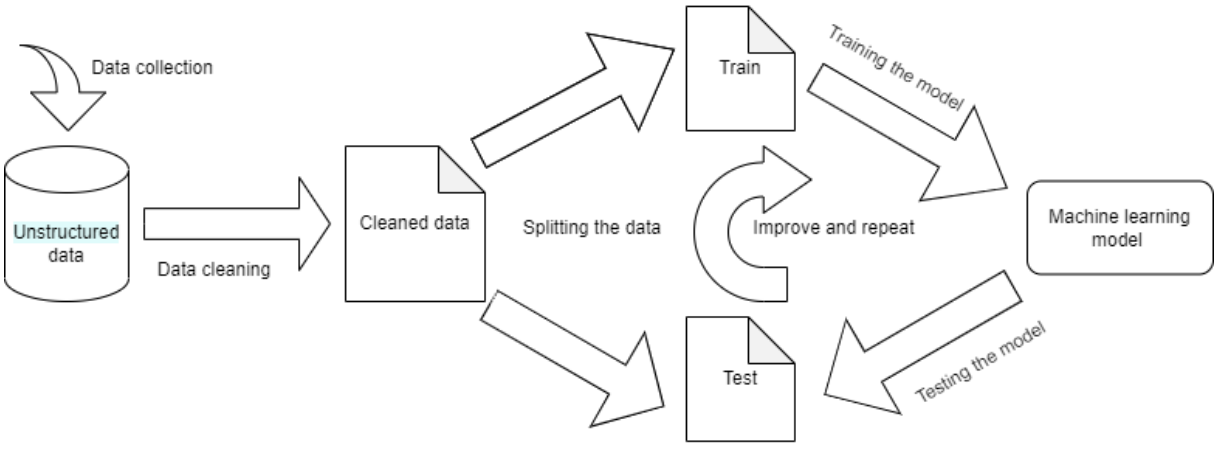


Figure 3.1: The Basic Steps Used In Machine Learning

3.3.1 Data Collection

The step of gathering data from different sources (databases, text files, cameras, hospitals, firewalls, etc.). It is the foundation of the future learning.

3.3.2 Data Cleaning

Any amelioration or modification based on analytical or statistical methods to improve the quality of data.

3.3.3 Model Training

This step is mainly about choosing both the right algorithm and form of data that will be split into two main parts. The first is the training data which will be fed to the model for training (learning patterns and making sense of linear or non linear data), the second is the test data which will be used as a reference for defining the actual performance of the model.

3.3.4 Model Evaluation

In this step we use the test data to define the accuracy (the precision in the choice of the algorithm based on the outcome) of our model. For this we have many methods:

Classification Accuracy:
The ratio is the number of correct predictions divided by the total number of input samples. It only works when there are an equal number of samples in each class (Mishra 2018).

$$\texttt{Accuracy} = \frac{\texttt{Number of correct predictions}}{\texttt{Total number of predictions made}} \tag{3.1}$$

Mean Absolute Error:
Mean Absolute Error is the average of the difference between the Original Values and the Predicted Values. It tells us how much the forecasts differed from the actual result. They don't, however, provide us any indication of the error's direction, i.e. whether we're under- or over-predicting the data. It is expressed mathematically as (Mishra 2018) :

$$\texttt{MeanAbsoluteError} = \frac{1}{N}\sum_{n=1}^{\infty} |Ytrue - Ypredicted| \tag{3.2}$$

Mean Squared Error:
The main difference between Mean Squared Error and Mean Absolute Error is that MSE

takes the average of the square of the difference between the actual and projected values. The benefit of MSE is that it is simpler to compute the gradient, whereas Mean Absolute Error necessitates the use of complex linear programming methods (Mishra 2018).

$$\texttt{MeanSquaredError} = \frac{1}{N}\sum_{n=1}^{\infty}(Ytrue - Ypredicted)^2 \tag{3.3}$$

Confusion Matrix:
The Confusion Matrix, as the name implies, produces a matrix as an output and uses the following four terms to describe the model's overall performance:

- True Positives : The cases in which we predicted YES and the actual output was also YES.
- True Negatives : The cases in which we predicted NO and the actual output was NO.
- False Positives : The cases in which we predicted YES and the actual output was NO.
- False Negatives : The cases in which we predicted NO and the actual output was YES.

(Mishra 2018).
The accuracy for the matrix can be calculated as:

$$\texttt{Accuracy} = \frac{TruePositive + TrueNegative}{Totalsamples} \tag{3.4}$$

F1 Score:
The Harmonic Mean of accuracy and recall is the F1 Score. F1 Score has a range of [0, 1]. It informs you how exact and robust your classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances) (Mishra 2018).

$$\texttt{F1} = 2 * \frac{1}{\frac{1}{Precision} + \frac{1}{recall}} \tag{3.5}$$

- Precision : It is the number of correct positive results divided by the number of positive results predicted by the classifier.

$$\texttt{Precision} = \frac{TruePositive}{TruePositive + FalsePositive} \tag{3.6}$$

- Recall : It is the number of correct positive results divided by the number of all samples that should have been identified as positive.

$$\texttt{Recall} = \frac{TruePositive}{TruePositive + FalseNegative} \tag{3.7}$$

3.4 Types Of Machine Learning Algorithms

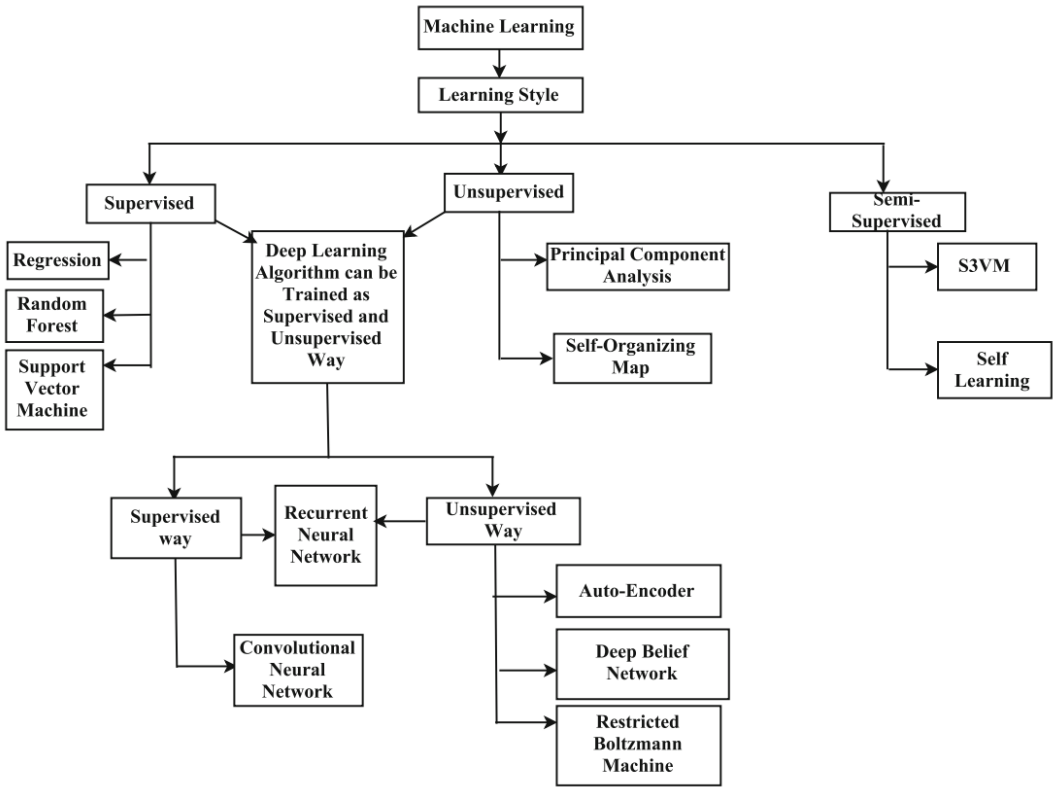


Figure 3.2: Overview Of Machine Learning Approaches Types (Sultana et al. 2017)

3.4.1 Supervised

SVM, Random Forest, Regression, and Knn are all examples of supervised learning algorithms, and even deep learning (Neural networks) may be taught in this manner. This means that these algorithms can learn patterns and representations from labelled data in order to obtain the information needed to predict fresh unlabeled data (Sultana et al. 2017).

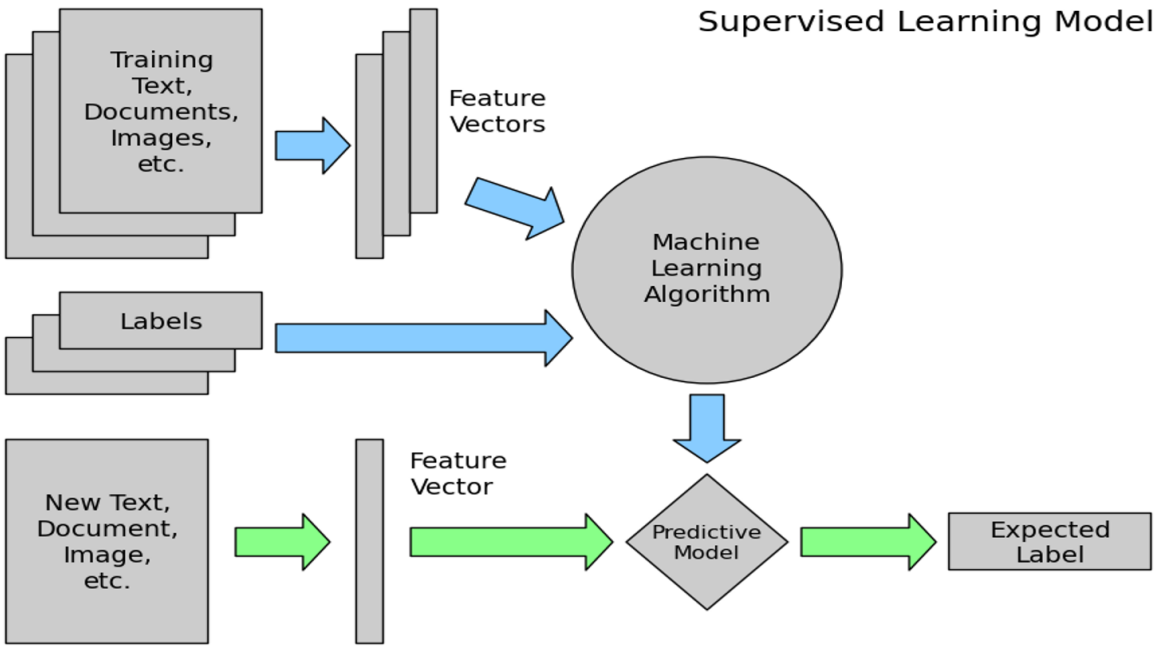


Figure 3.3: Supervised Learning Model

3.4.2 Unsupervised

Unsupervised learning techniques are used in algorithms like PCA, SOM, and k-means, and even deep learning (Neural networks) may be taught this way. This indicates that these algorithms are capable of learning representations and structure from unlabeled data. The purpose of an unsupervised learning algorithm is to anticipate unknown data by modeling the data's essential structure or distribution to predict unknown data (Sultana et al. 2017).

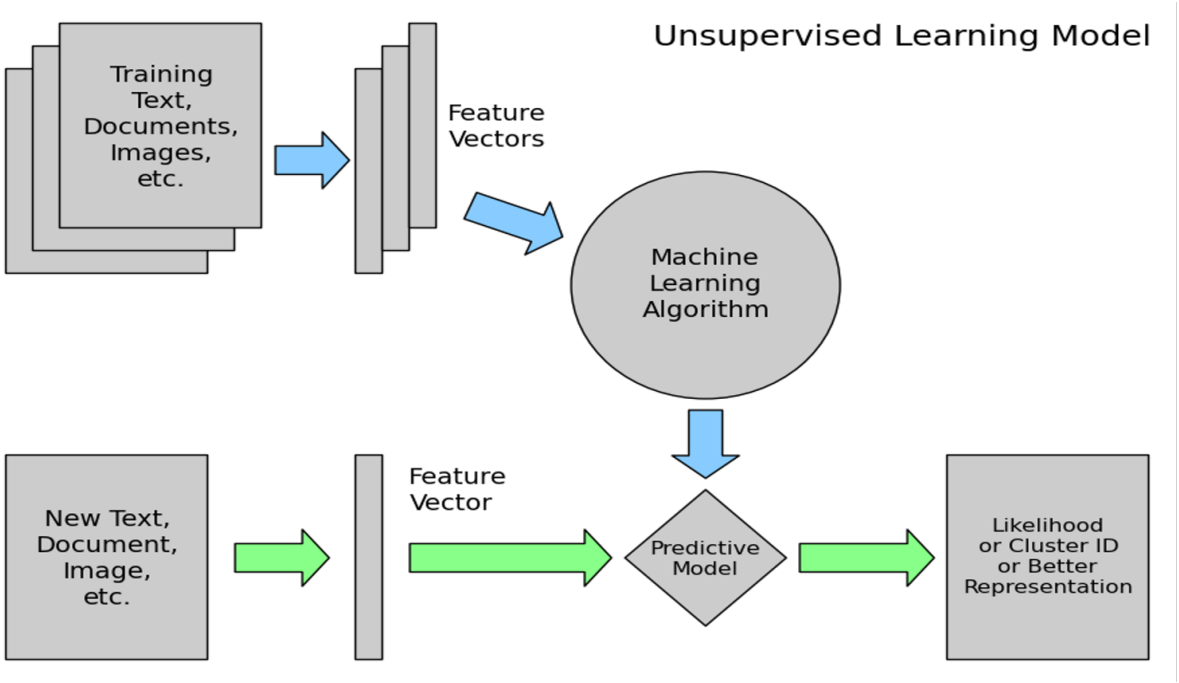


Figure 3.4: Unsupervised Learning Model

3.4.3 semi-supervised

Semi-supervised learning is a sort of supervised learning that incorporates the use of unlabeled data in the training process. A little quantity of labeled data and a huge amount of unlabeled data make up the training set. It's appropriate for situations when significant volumes of tagged data aren't accessible, such as picture archives with only a few identified photos (e.g., a person) and the most of them unlabeled (Sultana et al. 2017).

3.5 Background

3.5.1 Feature Extraction

Definition : Feature extraction is a process used in machine/Deep learning to ease the pattern recognition task for the model by building a new variables from the original features which must have to be informative (minimize information loss). When there is a large amount of data it tend to be redundant and complex, so for that This step can be used to reduce the features of the information and make it easier for the model to learn useful knowledge from a less complex but a better presented data (Sultana et al. 2017).

Feature extraction techniques :

- **PCA (Principal Component Analysis) :**
  Consists of an orthogonal transformation that converts samples of linearly uncorrelated features into samples of correlated variables. Principle components are new characteristics that are smaller or equal to the beginning variables. Because

PCA is an unsupervised approach, it does not incorporate data label information. When data is typically dispersed, principle components are self-contained. PCA is a straightforward non-parametric approach for extracting the most important information from a group of redundant or noisy data. This is the fundamental rationale for its use. (Khalid et al. 2014)

- **Word embedding :**
  Used in the natural language processing, word embedding is one of the most popular representation of document vocabulary. It is on of the best ways to get the best representation in term of the context for a word in a document,both in the semantic and syntactic similarity, also by taking in consideration the relation with other words. In the word embedding every word is transformed from its normal (letters) representation to a vectorized form.

- **Convolution layers :**
  Used widely in the field of image processing, the role of a convolution layer is to detect local features at different positions from the input feature maps, it is a filter in matrix form that extract the most important information from an image (curves, colors, edges, lines, etc.) (Sultana et al. 2017).

3.5.2 Machine Learning Algorithms

- **PCA (Principal Component Analysis) :**
  PCA is an unsupervised method to understand global properties of a dataset consisting of vectors. (Shashank Gupta 2018)

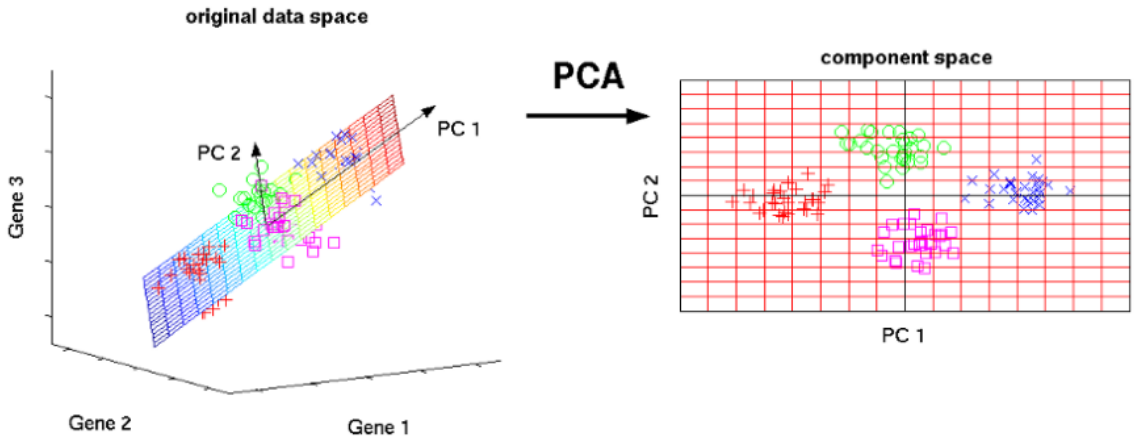


Figure 3.5: Principal Component Analysis (Shashank Gupta 2018)

- **K Means Clustering :**
  It is an unsupervised algorithm that takes a set of unlabeled data points as input and iteratively clusters them into K clusters depending on their distance.

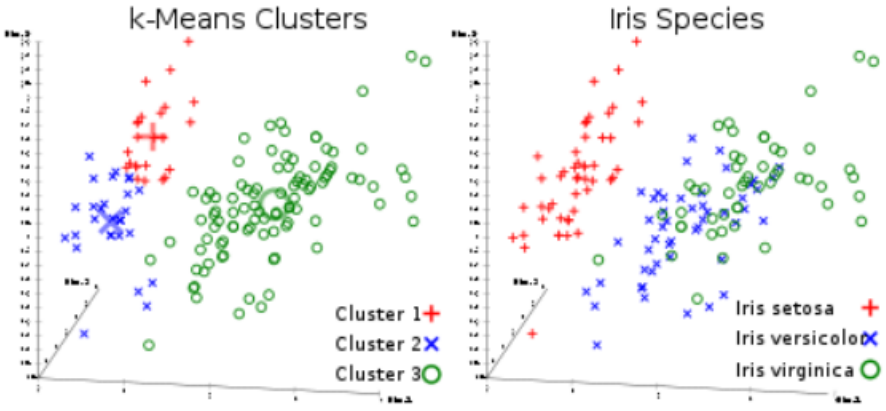


Figure 3.6: K Means Clustering Example On Real Data Set (Shashank Gupta 2018)

- **SVM (Support Vector Machines) :**
  SVM is a supervised method and a linear model; it may be thought of as a regressor, similar to linear regression, with the exception that it uses kernels on data to feature engineering and has a distinct margin-based loss function.

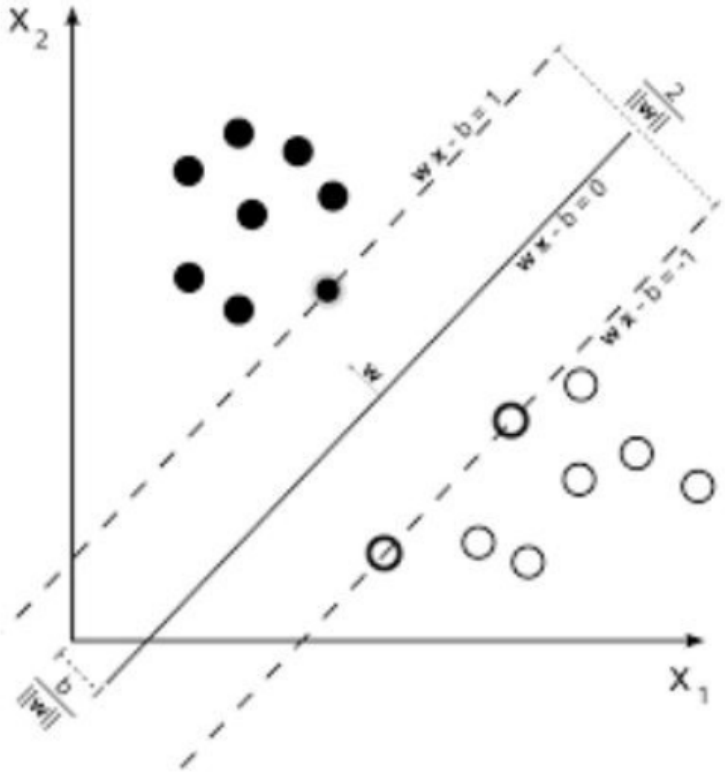


Figure 3.7: Support Vector Machines (Shashank Gupta 2018)

- **Linear Regression :**
  Linear regression is a supervised statistical procedure for numerical values that may determine the relationship between a dependent variable Y and the value of one or more independent variables X.

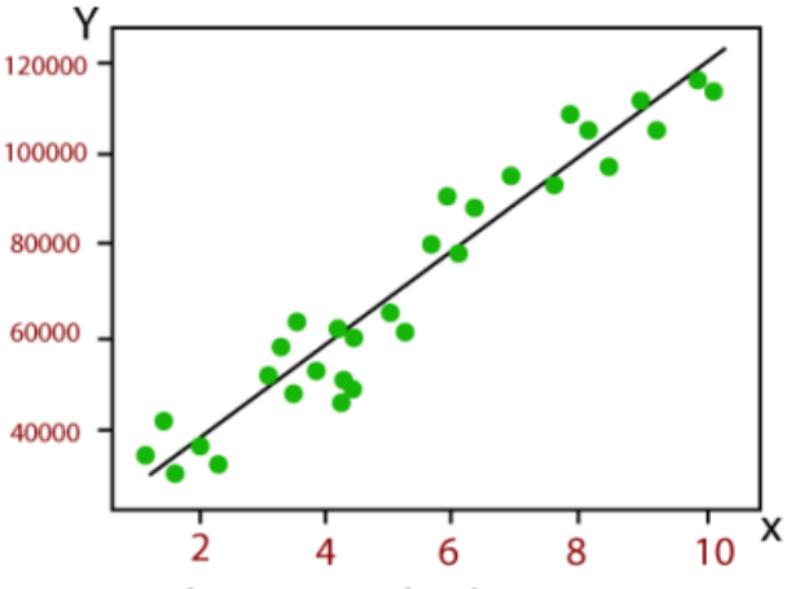


Figure 3.8: Linear Regression Visualization

- **Logistic Regression :**
  Logistic Regression is supervised algorithm used for classification, not regression. It is constrained Linear Regression with a non-linearity (sigmoid function is used mostly or you can use tanh too) application after weights are applied, hence restricting the outputs close to +/- classes (which is 1 and 0 in case of sigmoid). Cross-Entropy Loss functions are optimized using Gradient Descent.

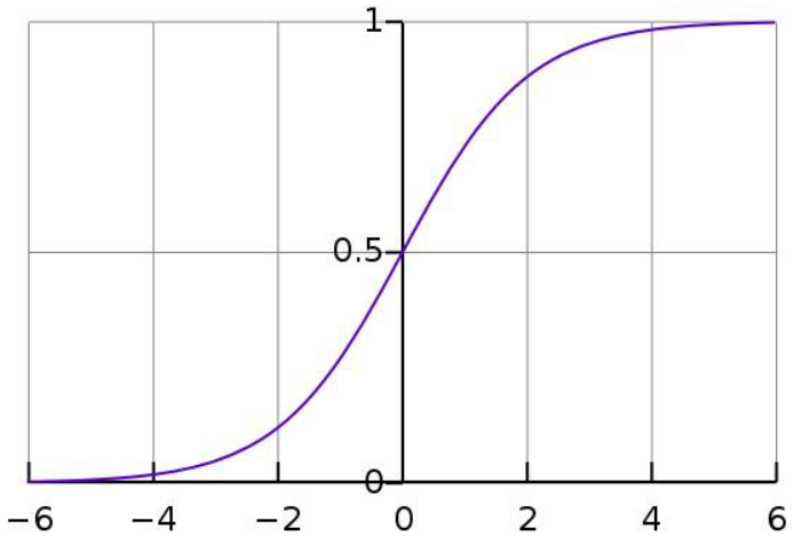


Figure 3.9: Logistic Regression Model (Shashank Gupta 2018)

- **Decision Trees :**
  Supervised classification method based on the principal of "divide and conquer", it uses recursive method to calculate the probability of every leaf class.

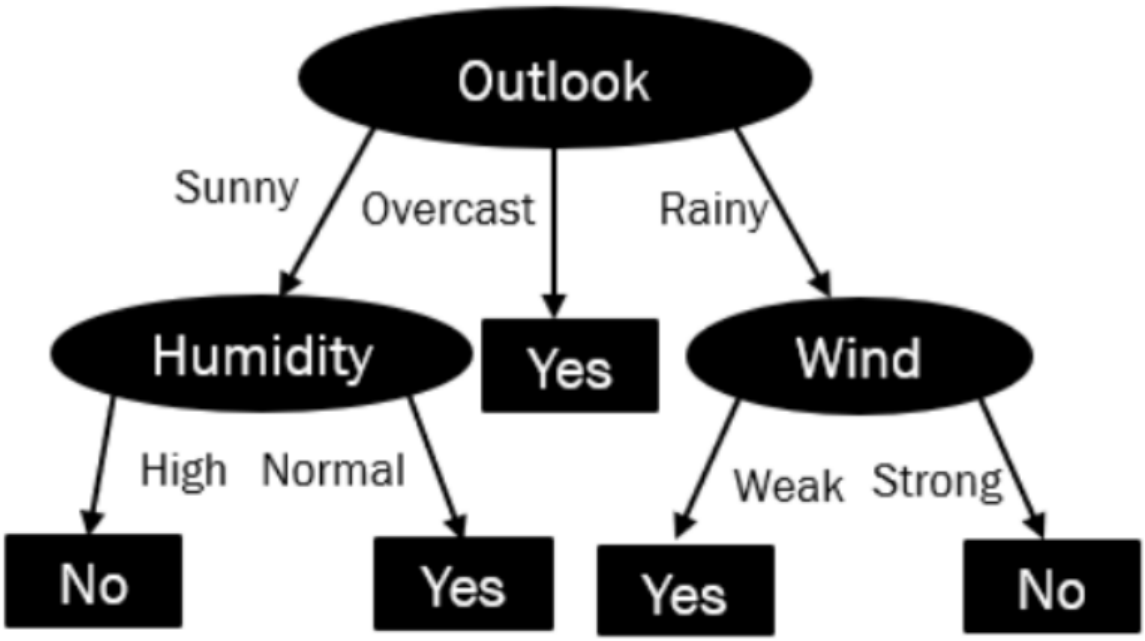


Figure 3.10: Decision tree model

3.5.3 Deep learning algorithms

- **Feed forward Neural Networks :**
  These are basically multilayered Logistic Regression classifiers. Many layers of weights separated by non-linearities (sigmoid, tanh, relu + softmax, etc). Multi-Layered Perceptrons is another name for them. As autoencoders, FFNNs may be utilized for classification and unsupervised feature learning. (Shashank Gupta 2018)

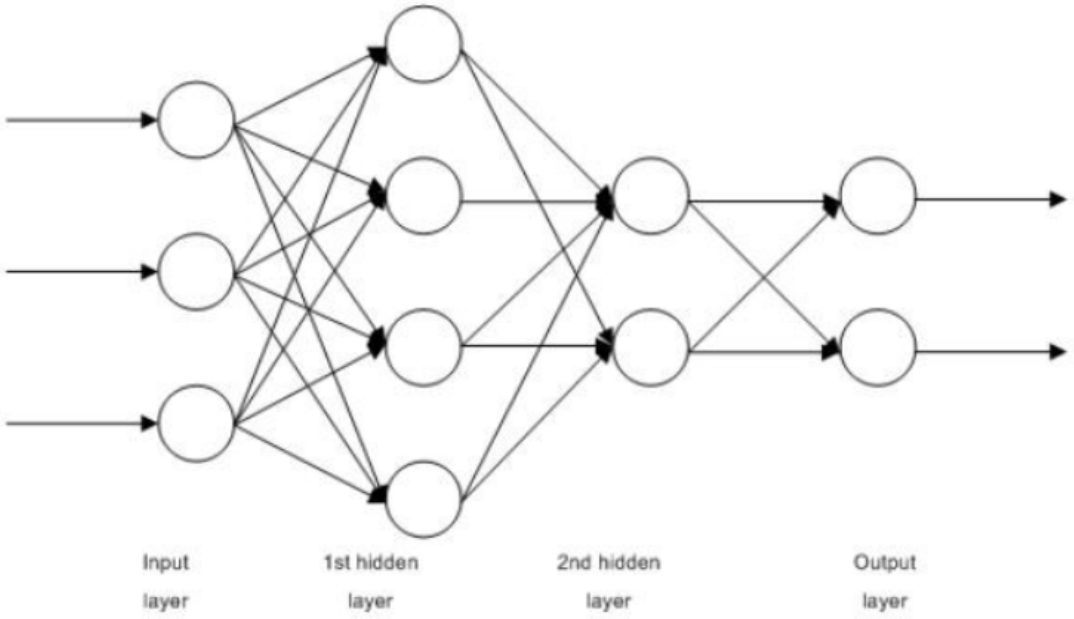


Figure 3.11: Feed forward Neural Networks Model

- **Convolutional Neural Networks:**
  Filters like layers used almost in every computer vision project such as Image classification, Object Detection or even segmentation of images, they act as hierarchical feature extractors.(Shashank Gupta 2018)

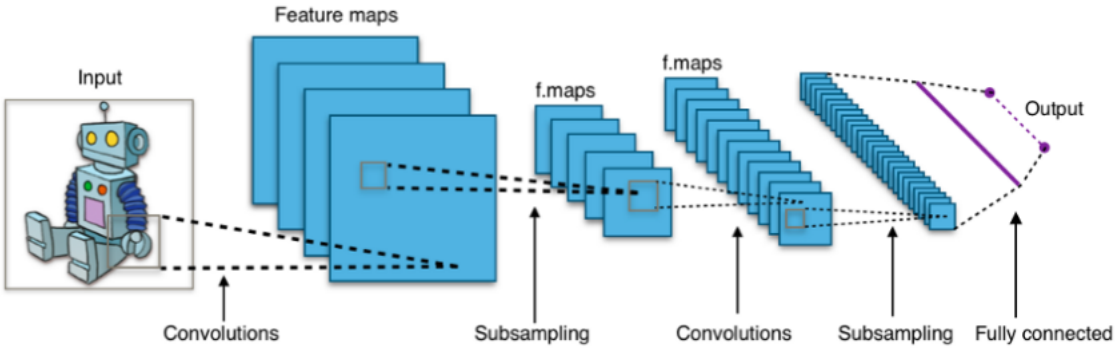


Figure 3.12: Convolutional Neural Networks Model (Shashank Gupta 2018)

- **Recurrent Neural Networks:**
  RNNs model sequences by iteratively applying the same set of weights on the aggregator state and input at time t. (Given a sequence has inputs at times 0..t..T, and have a hidden state at each time t which is output from t-1 step of RNN). In most sequence modeling applications, pure RNNs are no longer employed, but their equivalents, such as LSTMs and GRUs, are state-of-the-art.(Shashank Gupta 2018)

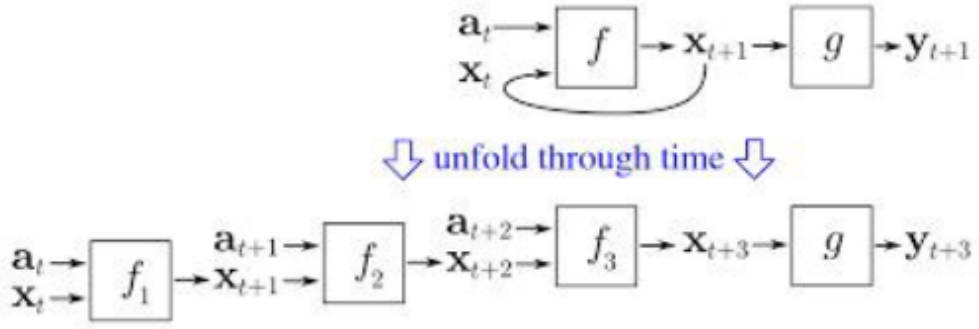


Figure 3.13: Recurrent Neural Networks Model (Shashank Gupta 2018)

- **LSTM (Long Short Term Memory):**
  It's a unique type of recurrent neural network that can learn long-term data relationships. This is possible because the model's recurring module is made up of four levels that interact with one another.

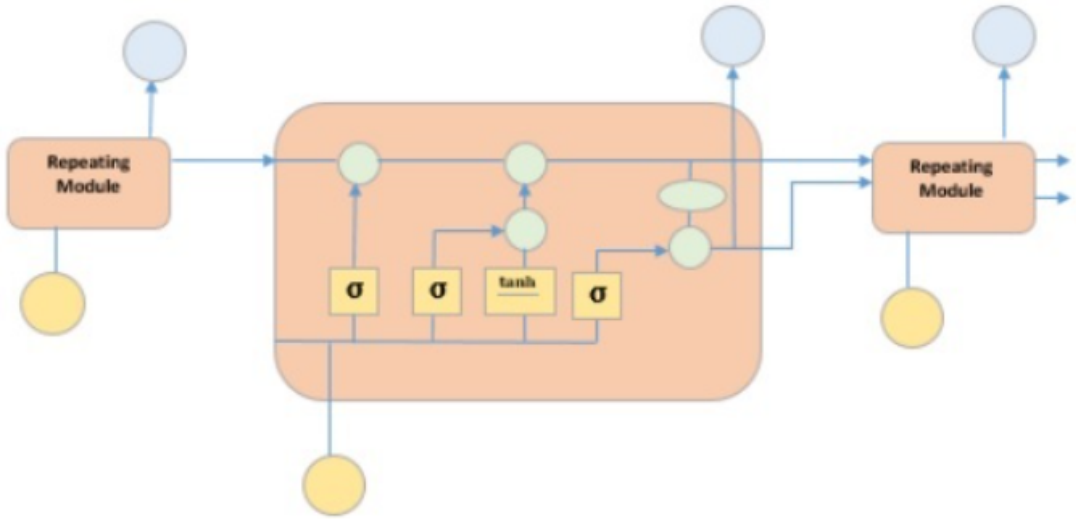


Figure 3.14: LSTM Model (Point 2020)

Four neural network layers are depicted in yellow boxes, point wise operators in green circles, input in yellow circles, and cell state in blue circles in the diagram above. An LSTM module contains a cell state and three gates, giving it the ability to learn, unlearn, or retain information from each of the units selectively. By permitting only a few linear interactions, the cell state in LSTM allows information to travel across the units without being changed.Each unit contains an input, output, and a forget gate that adds or removes data from the cell state. The forget gate utilizes a sigmoid function to determine whether information from the previous cell state should be ignored. The input gate uses a point-wise multiplication operation of'sigmoid' and

'tanh' to regulate the information flow to the current cell state. Finally, the output gate determines which data should be sent to the next concealed state (Point 2020).

- **Auto Encoders:**
  An autoencoder is used to learn a representation (encoding) for a set of data for the purpose of dimensionality reduction. It can learn to reconstruct its inputs when trained with a set of examples in an unsupervised way. The layers then act as feature detectors on inputs. (Sultana et al. 2017)

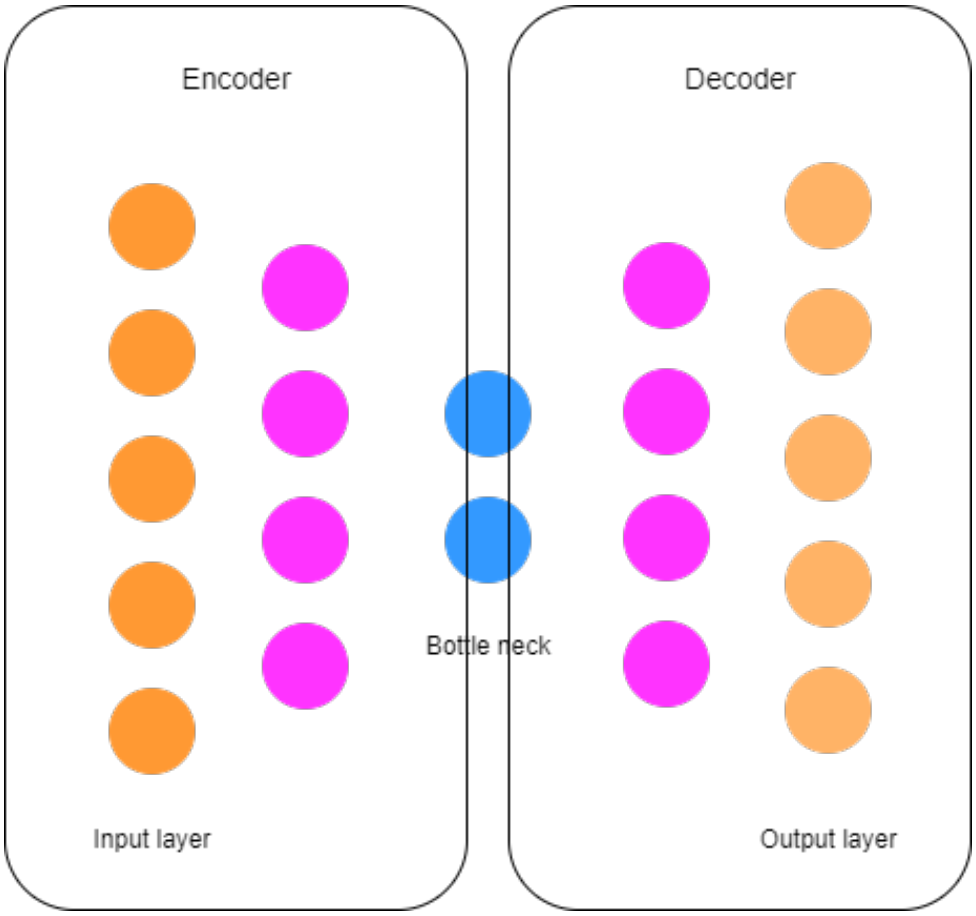


Figure 3.15: Autoencoder model

3.6 conclusion

Machine/deep learning already have major impact on our living way with his specialized applications in various domains such as medical, business, agriculture and networking.

Chapter 4

Network Intrusion Detection System

4.1 Introduction

Network security is becoming such a growing field due to the explosion of network traffic, which means the appearance of new and more malware. Due to this explosion the task of log analysis and anomaly detection in the network is becoming more and more heavy, so a great deal of attention has been given to the automation of this task using intelligent models.

4.2 Definitions

4.2.1 Anomaly

A deviation from the common rule or an unusual behaviour that is strange enough to be noticeable (He et al. 2021).

4.2.2 Anomalous Behavior

Strange behaviour over time that doesn't match with the expected one that can be identified as intrusion, failure or malware on network log data (He et al. 2021).

4.2.3 Anomaly Detection

Anomaly detection is the task of identifying system anomalous patterns that do not conform to expected behaviors on log data. Typical anomalies often indicate possible error, fault or failure in software systems (He et al. 2021).

4.2.4 Network Intrusion Detection System

An Intrusion Detection System (IDS) is developed in a network to detect threats from monitoring packets transmitted though or logs generated from this packets. IDSs detect anomalous and malicious activities from inside and outside intruders. Large network traffic volumes, highly unequal data distribution, and unstructured data sets are among difficulties that an IDS must cope with. An IDS's main job is to keep an eye on information sources like computers and networks for unauthorized access attempts. IDSs collect data

from different systems and network sources and analyse the data for possible threats. (Sultana et al. 2017)

4.3 Types Of Anomalies

4.3.1 Point Anomaly

A single data point that deviates from the common rule or expected output is referred to as an anomaly point (Ahmed et al. 2015).

4.3.2 Contextual Anomaly

A contextual or conditional anomaly occurs when a data object acts abnormally in a certain context (Ahmed et al. 2015).

4.3.3 Collective anomaly

When a collection of similar data instances behave anomalously with respect to the entire dataset, the group of data instances is termed a collective anomaly (Ahmed et al. 2015).

4.4 Intrusion Detection Datasets And Issues

Due to security and privacy issues Intrusion detection datasets are difficult to find publicly, in the other hand they can't be generalized to every company since the network infrastructure differs from one company to an other.

4.4.1 KDD

This is one of the most widely used datasets in the field. Due to its ease of deployment, the KDD datasets were developed using a Solaris-based operating system to collect a wide range of data. However, we can see significant differences in today's operating systems which barely resemble Solaris. Solaris has practically little market share in this day and age of Ubuntu, Windows, and MAC. TCPdump, the traffic collector used in KDD datasets, is prone to being overwhelmed and dropping packets under high traffic loads. More critically, there is substantial ambiguity regarding the attack distributions of these datasets. There are 24 training and 14 test attacks in the KDD datasets, according to the description. However, it is reported that the training data contain 22 attacks and the test data 17. This inconsistency has a significant impact on the class distribution of attacks.(Ahmed et al. 2015)

4.4.2 Contemporary Network Attacks Evaluation Dataset: ADFA-LD12

The ADFA-LD12 dataset is a publicly accessible dataset that represents contemporary attack structure and methods. The dataset was created using the Ubuntu Linux operating system. The attacks were carefully chosen by the dataset's creators, who concentrated

on the techniques employed by modern penetration testers and hackers. There was also a trade-off between the targeted system's vulnerability and the realism required, with the ADFA-intended LD12's target server completely patched to provide a realistic target. Surprisingly, the performance of intrusion detection systems on the KDD datasets differs from the ADFA-LD12 dataset, owing to the fact that the KDD dataset does not represent current attacks. (Ahmed et al. 2015)

4.4.3 NSL-KDD

It's a dataset that's been proposed as a way of addressing some of the KDD dataset's flaws.(Ahmed et al. 2015)

4.5 Internet Traffic Archive

It is funded by ACM SIGCOMM and serves as a repository for facilitating universal access to traces of internet network activity. However, it is heavily anonymized, lacks crucial packet information, is not labeled, and does not support multiple-attack situations.(Ahmed et al. 2015)

4.5.1 PREDICT

Stands for Protected Repository for the Defense of Infrastructure Against cyber threats. It is a networking and information security research community centered in the United States that produces security-relevant network operational data. This dataset provides frequently updated network data essential to cyber security research to developers and assessors.(Ahmed et al. 2015)

4.6 Intrusion Detection Techniques

4.6.1 Signature Based Detection

The detection range of a signature-based NIDS is restricted to known harmful threats. Through signature specification, a mix of packet header and packet content inspection criteria are applied to the detection system from aberrant traffic flows (Sultana et al. 2017).

4.6.2 Anomaly Based Detection

For signature-based NIDS, anomaly detection algorithms are meant to automatically recognize threats that are unknown and unpredictable. Anomaly-based intrusion detection approaches include technologies like machine learning (Sultana et al. 2017).

technique	Alarm Rate	Speed	Flexibility	Reliability	Scalability	Robustness
Signature	Low	High	Low	High	Low	Low
Anomaly	High	Low	High	Moderate	High	High

Tableau 4.1: Comparison Between Detection Method (Sultana et al. 2017)

4.7 Signature Vs Anomaly Based Detection

4.8 Types Of Networking Attacks

4.8.1 Denial Of Service (DoS)

DoS attacks are when a hacker makes memory resources too busy to handle legitimate networking requests, denying users access to a computer (for example, apache, smurf, Neptune, ping of death, back, mail bomb, UDP storm, and so on) (**Intrusion˙Detection˙System˙Using˙Se**

4.8.2 Remote To User Attacks (R2L)

A remote to user attack is when a user sends packets to a machine that the user does not have access to over the internet in order to expose the machine's vulnerabilities and exploit privileges that a local user would have on the computer, such as xlock, guest, xnsnoop, phf, sendmail dictionary, and so on (**Intrusion˙Detection˙System˙Using˙Self˙Organizing˙Maps**).

4.8.3 User To Root Attacks (U2R)

These attacks are exploitations in which the hacker logs in as a regular user and attempts to exploit system weaknesses in order to get super user privileges, such as perl or xterm (**Intrusion˙Detection˙System˙Using˙Self˙Organizing˙Maps**).

4.8.4 Probing

Probing is a type of attack in which a hacker checks a computer or a network device for flaws or vulnerabilities that may be exploited later to breach the system. devil, saint, portsweep, mscan, nmap, and other data mining tools employ this method (**Intrusion˙Detection˙Syste**

4.9 Intrusion Detection Methods

4.9.1 Rule-based

Rule-based anomaly detection techniques (expert system) are simple to use but they are difficult to implement. The basic idea is to learn the normal behavior of a system and anything not encompassed within it considered anomalous. These techniques consider both single and multi-label learning algorithms. in general it means defining a set of

rules or conditions which present the normal behaviour and any thing but this rules is anomalous (Sultana et al. 2017).

4.9.2 Supervised Learning

- **Support vector machine (SVM)** They are widely used in NIDS research due to its powerful classification power and practicality in computation. They are suitable for high dimensional data, but selecting a reasonable kernel function is critical (Sultana et al. 2017).

- **Recurrent neural network (RNN)** They are widely used in in the field of anomaly detection especially LSTM's because of their ability to detect patterns in time series data and in most of the cases the anomaly is Collective over the time and not in a single data point.

4.9.3 Unsupervised learning

- **AutoEncoders** They are generative models that have the capacity to reconstruct data. once trained on normal network data they will achieve the ability to reconstruct it with minimal error. Given an anomalous data they will reconstruct it with a significant error and that's how the anomaly is detected based on the error variance between the input and output of the autoencoder.

- **Self-organizing maps** A neural network model for processing and displaying high-dimensional data is the Self-Organizing Map. It belongs to the competitive learning network category. The SOM is a two-dimensional array of neurons that maps a high-dimensional input data space to a normal two-dimensional array of neurons. It's a competitive network whose purpose is to convert an arbitrary-dimension input data set into a one- or two-dimensional topological map.Its goal is to uncover the input data set's underlying structure, such as a feature map, by creating a topology preserving map that depicts the points' neighborhood relations (**Intrusion˙Detection˙System˙Using˙Self˙Organizing˙Maps**).

- **Clustering** it does not require pre-labeled data to extract rules for grouping similar data instances. So, by having a set of data points they are grouped in clusters with the most normal network data is considered as normal and the others are considered as anomalies.

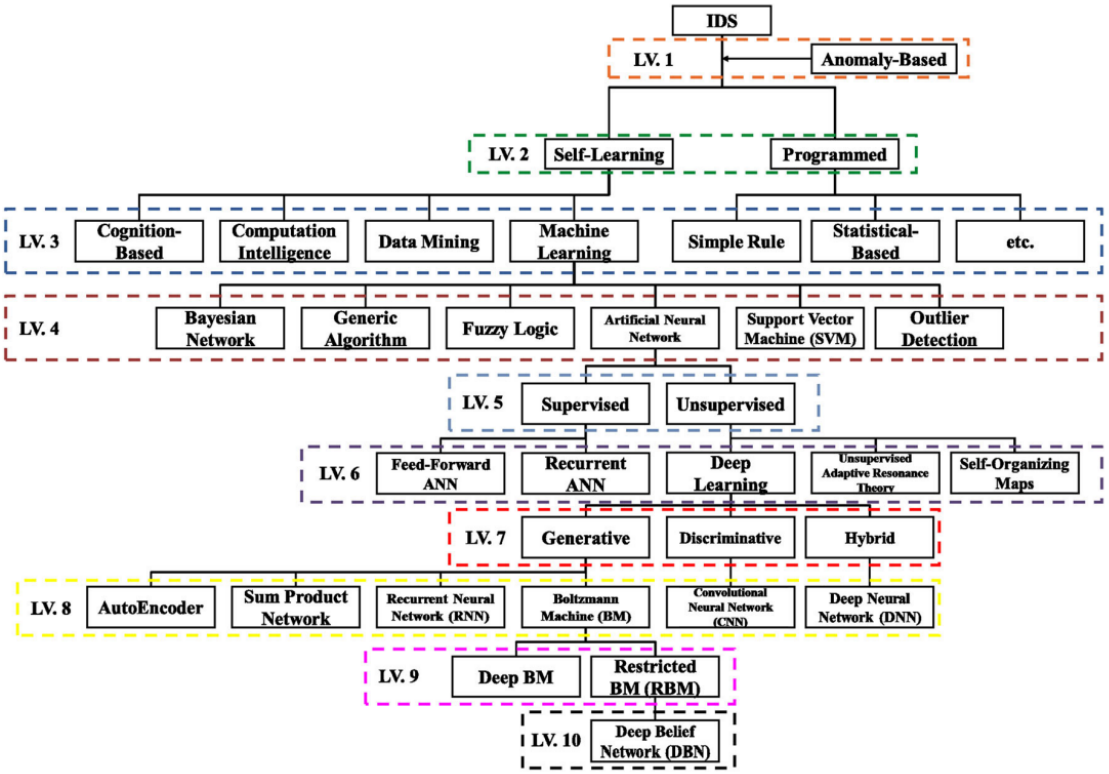


Figure 4.1: Classification Of Anomaly Detection (Kwon et al. 2017)

4.10 Conclusion

An intrusion detection system is such an important software in any company nowadays because of the layer of protection that he adds to the network and security to the information which are vital aspects which grow by the company growth.

Part II

State Of The Art

Chapter 5

Comparative Study Between The Related Works

5.1 SVM Based Network Intrusion Detection For The UNSW-NB15 Dataset

5.1.1 Problem Description

Over the past decade, the KDDCUP99 dataset and its refined version NSL-KDD dataset are the most widely used in NIDS, some works indicate that these datasets can not reflect the output performance of NIDSs well due to the following problems: Firstly, modern low footprint attacks are missing. Secondly, since these datasets were established twenty years ago, there is much difference between the normal flow defined by the benchmark datasets and the present normal flow. In response to these challenges, the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) created the UNSW-NB15 dataset in 2015, which generates a hybrid of the real modern normal behaviors and the current comprehensive attack environment. Compared to the KDDCUP99 and NSL-KDD datasets, the UNSW-NB15 dataset includes more modern abnormal types and normal behaviors captured over time. Besides, the data includes in-depth features of network traffic.
In this solution they used SVM model with UNSW-NB15 dataset for both binary and multi class classification. (Jing et al. 2019)

5.1.2 Model Architecture

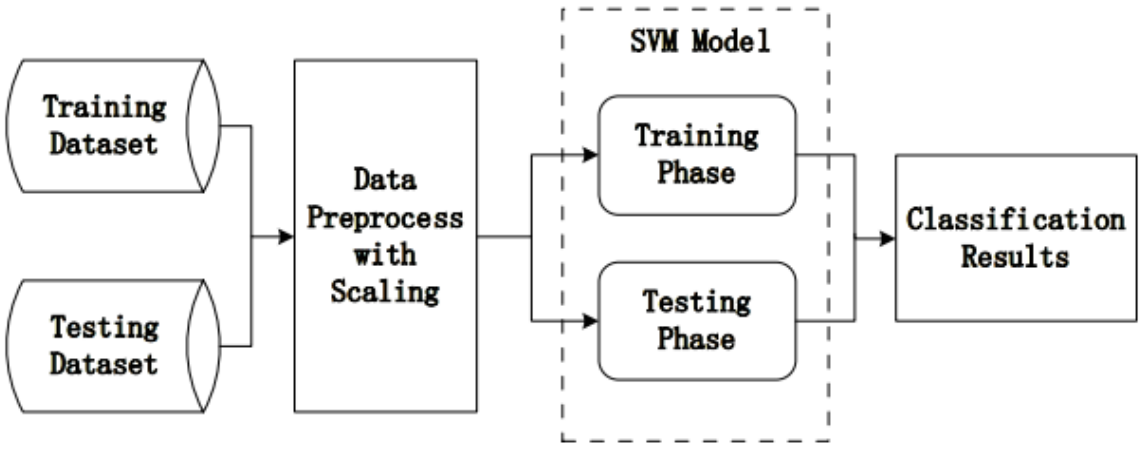


Figure 5.1: Intrusion Detection With The Proposed SVM Method. (Jing et al. 2019)

As a supervised learning algorithm, support vector machine (SVM) is often used as a classifier for intrusion detection, especially in high-dimensional space. In their paper, a SVM model is proposed for intrusion detection. Particularly, they use a nonlinear log function scaling method instead of the traditional linear normalization in the data preprocessing stage, which is more suitable for the UNSW-NB15 dataset.
The normalization function used is : X' = LOG10(X)

5.1.3 Experiment

5.1.3.1 Binary-classification results

Type	Detaction Rate
Normal	72.6
Anomaly	97
AVG	86

Tableau 5.1: Performance Accuracy Table

5.1.3.2 Multi-classification Results

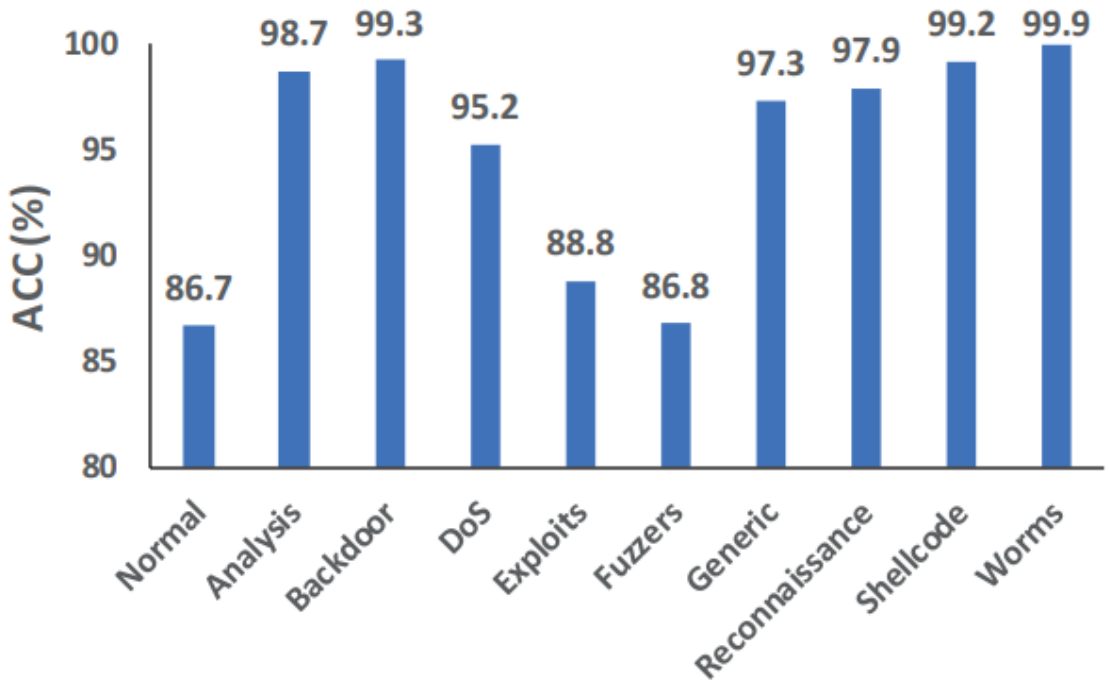


Figure 5.2: Accuracy Of The Ten Classes In Testing Dataset. (Jing et al. 2019)

5.2 Intrusion Detection System Using PCA with Random Forest Approach

(Waskle et al. 2020) proposed an effective network anomaly traffic detection algorithm. They converted the network traffic anomaly detection problem to a classification problem, and proposed an random forest model with PCA method to solve it.

5.2.1 Problem Description

The suggested system includes two methods: one is principal component analysis, and the other is random forest. The principle component analysis is used to reduce the dataset's dimension; this approach improves the dataset's quality by ensuring that the dataset has the proper characteristics. Following that, the intruders will be detected using the random forest method.

Improvement Algorithm for the Base Classifier:

- initialization of the active property of the data set by marking All condition attributes.
- Determine the modulus for each condition attribute in the primary and secondary sets.
- This stage calculates the compatibility of all conditional attributes using equation (5.1). If there are other characteristics with comparable compatibility, use equation (5.2).
- Select the most complete compatibility for splitting as the split node and remove the active tag to divide the sample.
- Continue picking the active attribute for splitting until you reach the leaf node's active quality.
- At last, the base classifier is generated.

5.2.2 Model Architecture

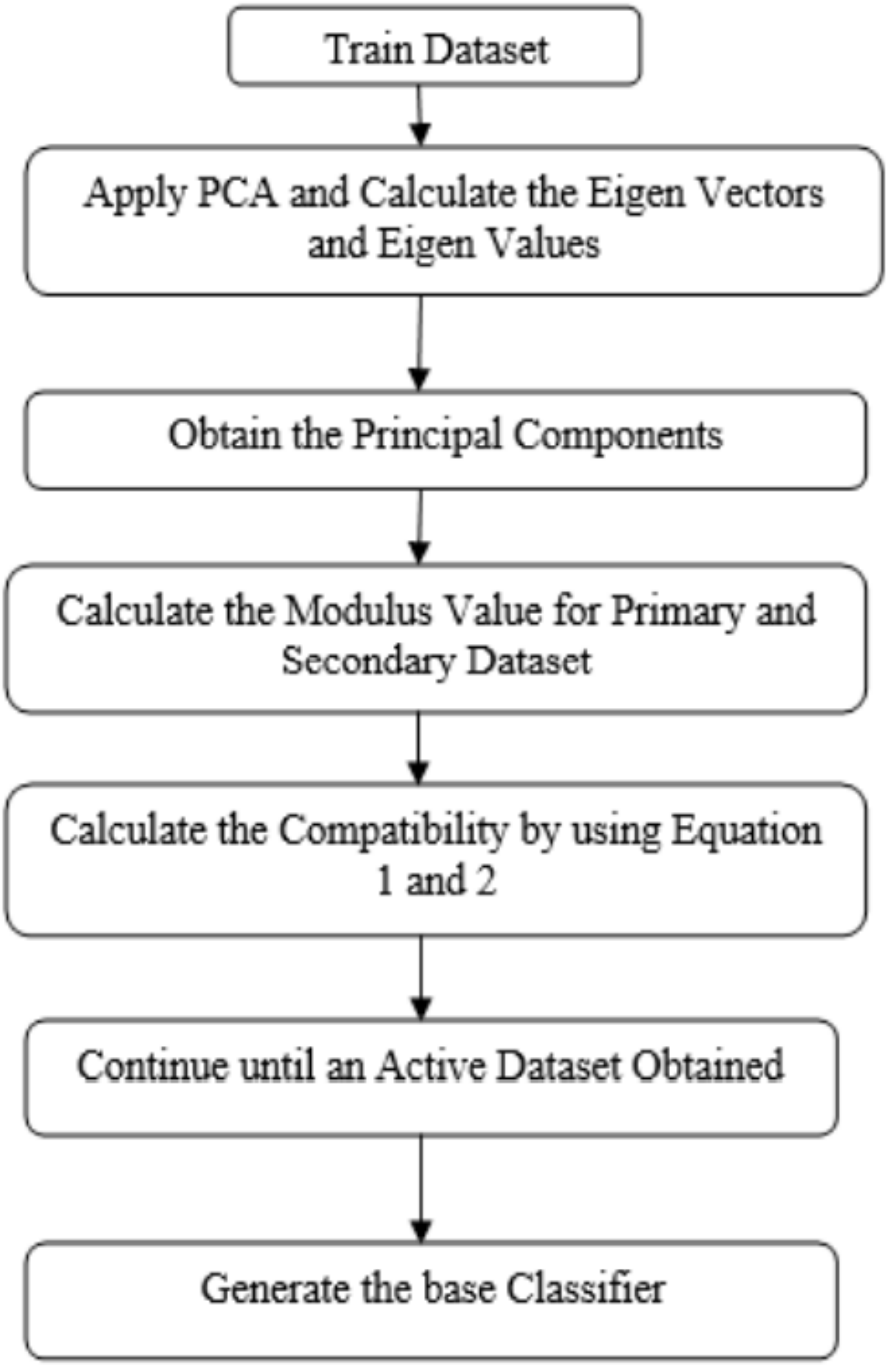


Figure 5.3: Flowchart For The Proposed Approach (Waskle et al. 2020)

The equations :

$$\texttt{compatibility(X -> D)} = \frac{|primarySet| - |secondarySet|}{|X|} \tag{5.1}$$

$$\texttt{compatibility(X -> D)} = \frac{|primarySet|}{|X|} \tag{5.2}$$

5.2.3 Experiment

For the test purpose they used the KDD dataset. The performances were like this:

Accuracy rate	Error rate
96.78	0.21

Tableau 5.2: Performance Table (Waskle et al. 2020)

5.3 Intelligent Intrusion Detection System Using Clustered Self Organized Map

(Almi'ani et al. 2018) proposed a neural network-based technique that is ideal for Intrusion Detection Systems (IDS). "Self-Organizing Maps" is the name of the algorithm (SOM). The neural networks method is a promising methodology that has been utilized to solve a variety of classification issues. The neural network component will implement the neural method, which is based on the concept that each user is distinct and leaves a distinct imprint on a computer system when they use it. The system administrator or security officer can be notified if a user's footprint does not match his or her reference footprint as a result of typical system activity.

5.3.1 Problem Description

The SOM is usually used in field such as clustering, data reduction, data compression or patterns recognition but in this solution they used it for network intrusion detection to detect anomaly traffic in networks by classifying traffic into categories.

5.3.2 Model Architecture

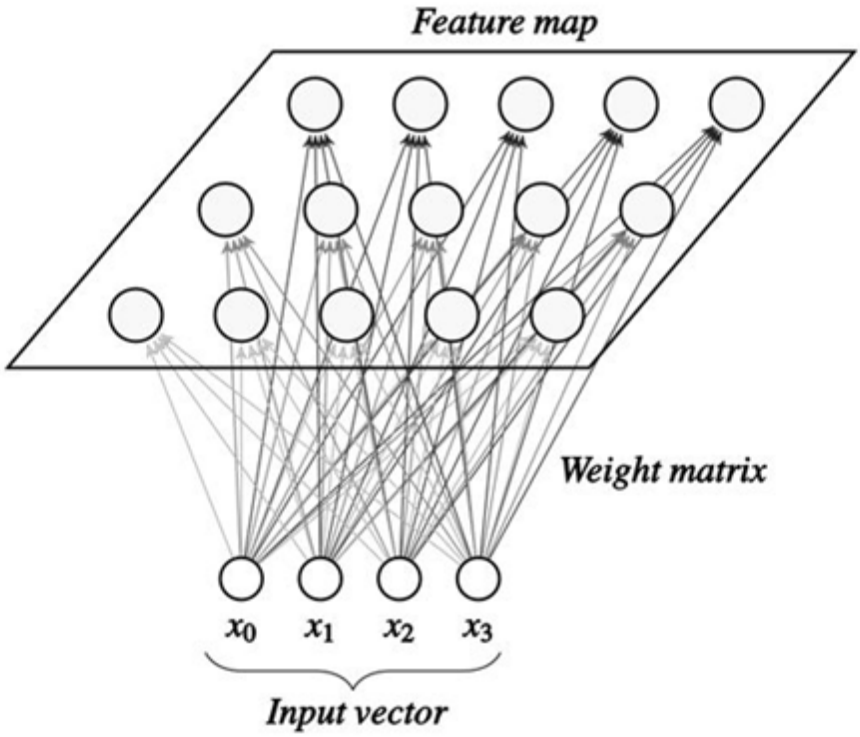


Figure 5.4: The self-organizing (Kohonen) Map (Almi'ani et al. 2018)

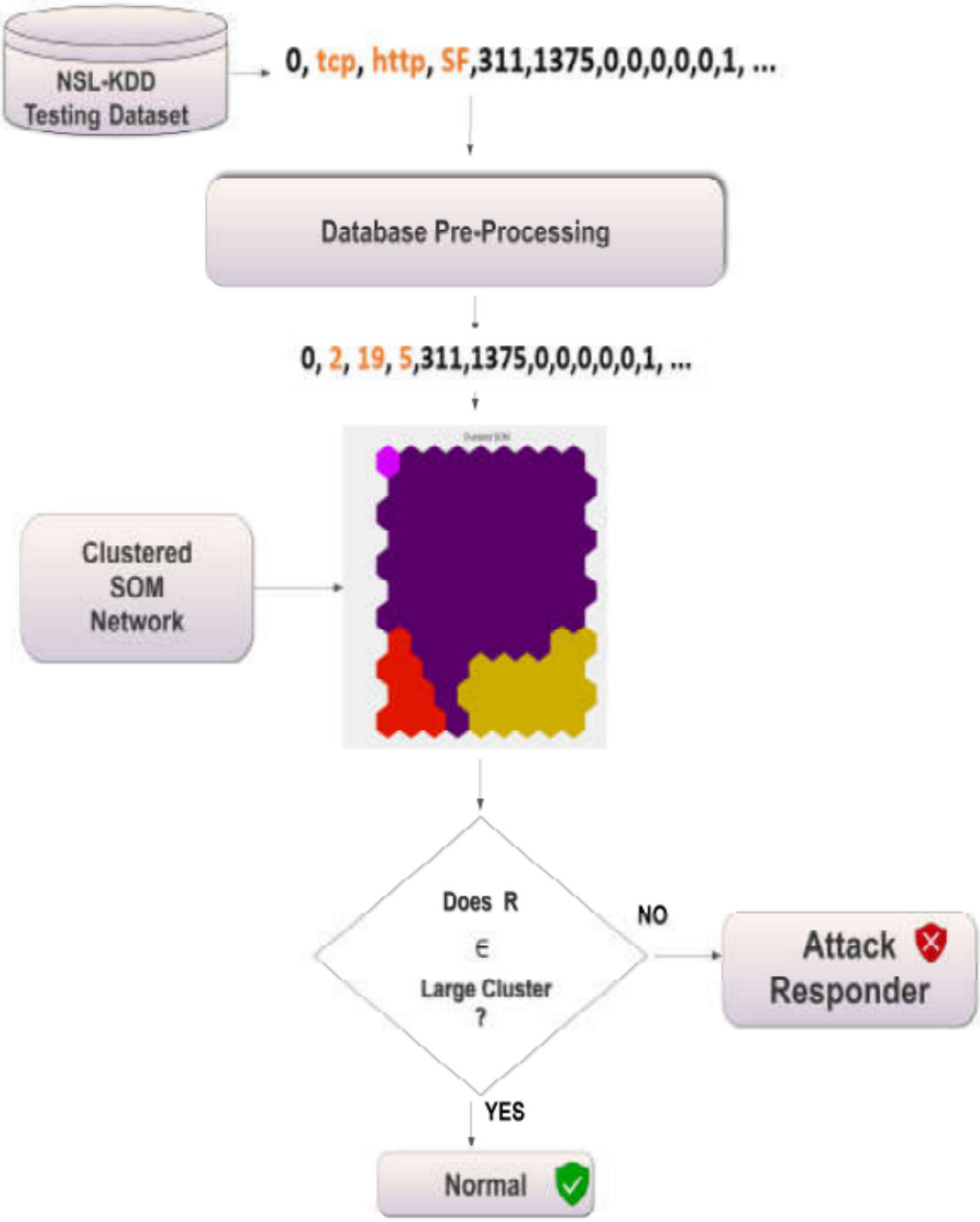


Figure 5.5: Explanatory Example Of Detection Process Using SOM (Almi'ani et al. 2018)

The SOM has a single feed forward network topology, with each input layer source node coupled to all output neurons. In most cases, the number of input dimensions is greater than the number of output dimensions. The Kohonen layer neurons in the SOM are arranged into a grid , see figure 5.4, and are in a different space from the input space. The technique looks for clusters in which two contiguous clusters in the grid have codebook vectors in the input space that are close to each other. Another way to look at it is that in the input data set, relevant data is organized into clusters in the grid. The training

employs competitive learning, which means that the weight vector of the neuron with the most comparable weight vector to the input vector is modified toward the input vector. The neuron is said to be the 'winning neuron' or the Best Matching Unit (BMU). The weights of the neurons close to the winning neuron are also adjusted but the magnitude of the change depends on the physical distance from the winning neuron and it is also decreased with the time.
The aim of training process is not to make a supervision, rather, it aims to make the topology of SOM network covers the input space. Finally, we'll have a clustered SOM, which is the outcome of using a clustering method on SOM neurons, where neurons with similar weight vectors are clustered together. The k-means algorithm is used in their suggested system for this purpose. As a result, the pattern of SOM neurons activation is enlarged in terms of clusters, i.e., one cluster is activated for each input connection record, while the remaining clusters stay inactive (Almi'ani et al. 2018).

5.3.3 Result

For this example they used the NSL-KDD dataset for both training and testing and here is the results:

Parameter	Value
Specificity	72
Sensitivity (TPR)	96.66
Accuracy	83.46
Precision	75

Tableau 5.3: Performance Accuracy Table (Almi'ani et al. 2018)

5.4 Hybrid Intrusion Detection System Using K-means And K-nearest Neighbors Algorithms

5.4.1 Problem Description

(Aung et al. 2018) proposed a hybrid solution using K-means and k-nearest neighbors algorithms for the network intrusion task. K-means is used for clustering the data and k-nearest neighbors is used to find to closer cluster to a data point (Normal cluster or abnormal).

5.4.2 Model Architecture

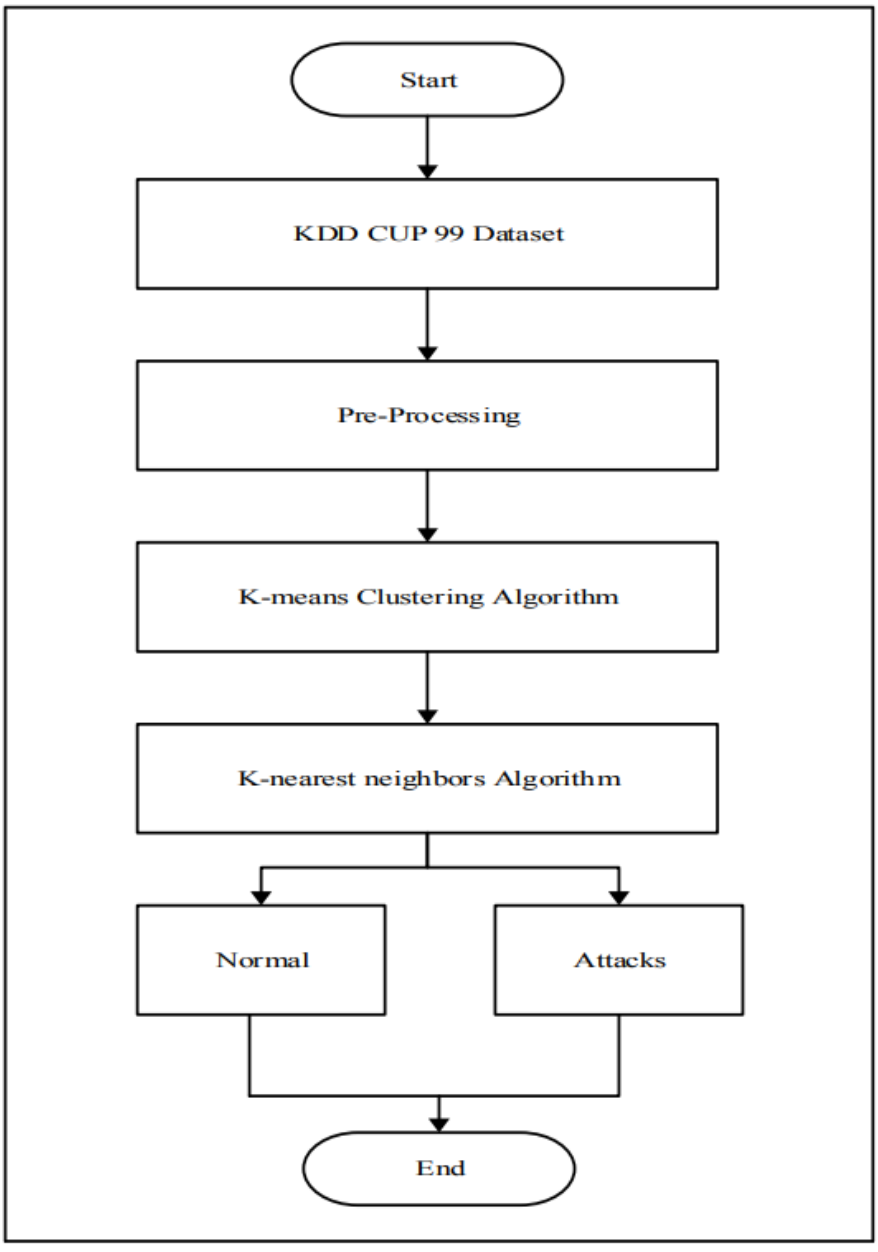


Figure 5.6: Hybrid K-means And KNN Intrusion Detection System (Aung et al. 2018)

This section consists of the conversation of the two algorithms of data mining classification approaches. These are K-means and K-nearest neighbors algorithms. The system design of intrusion detection system can be seen in figure 5.6.

5.4.2.1 K-means Algorithm

Starting with k arbitrary cluster centers in space, the K-means clustering method divides the set of given objects into k subgroups based on a distance measure. Cluster centers are updated repeatedly depending on an objective function that is optimized. This approach is one of the most prominent clustering algorithms, and it is extensively used since it is simple to construct and has a linear time complexity. The main purpose of using the K-means clustering approach is to divide a collection of similar-behaving normal and attack data into many divisions known as K-th cluster centroids. To put it another way, K-Means calculates a fixed number of K, the optimum cluster centroid for data with similar characteristics. In this work, they predefined K=5, representing Cluster 1, Cluster 2, Cluster 3, Cluster 4 and Cluster 5. (Normal, Dos, Prob, R2L and U2R) (Aung et al. 2018).

5.4.2.2 K-nearest Neighbors Algorithm

The outcome of KNN classification is a class membership. A majority of an item's neighbors vote to classify it, with the object being allocated to the most common class among its K closest neighbors (k is a positive integer, typically small). If k = 1, the item is simply assigned to that single nearest neighbor's class. The membership in this case is to one of the five clusters.

5.4.3 Experiment

They employed the KDDCup 99 dataset in this experiment; the test covers just around 10 percent of the dataset, which has 494021 Connection records with a total of 41 attributes, 7 symbolic fields, and 34 numeric values.

dataset	k-means	KNN	Correctly Classified Instances	Correct Instances percentages	Incorrectly Classified Instances	Incorrect Instances percentage
10% P1	Y	Y	108838	99.9936	7	0.0064
10% P2	Y	Y	23491	99.8427	37	0.1573
10% P3	Y	Y	280797	99.9996	1	0.0004
10% P4	Y	Y	78629	99.8375	128	0.1625
10% P5	Y	Y	2054	98.1366	39	1.8634
Total	Y	Y	493809		212	

Figure 5.7: Testing Results For 10 Fold Cross Validation (Aung et al. 2018)

5.5 Network Intrusion Detection Using Sequential LSTM Autoencoders

(Mirza et al. 2018) Using LSTM-autoencoders, they built an online sequential unsupervised framework for network intrusion detection. Because it operates on both fixed and variable length network data sequences, the proposed framework is dynamic and scalable. The suggested framework performs well as a feature extractor and also makes use of previous data to make accurate conclusions.

5.5.1 Problem Description

In this solution they trained multiple autoencoders with multiples hyper parameters and evaluated them on intrusion detection evaluation data set (ISCX IDS 2012). Network payloads are captured for seven days. There are around 1.8 millions of connections for FTP, SMTP, HTTP, SSH, IMAP, and POP3. Around five percent of connections are labelled as an anomaly. These anomalies come from a diverse set of multi-stage attacks. Some connections do not have packet payloads at the source or/and destination ports. Since packet payloads are used as input in their systems, they disregard the connections without payloads at both ports, regarding anomaly occurrence rate must remain almost the same after this operation. The model will be trained to reconstruct the normal data, for normal data sequences, the value of reconstruction error is less than the reconstruction error for anomalous data sequence. As a result, in order to classify the data as an anomaly, we assign a threshold value T . The value of T is critical, as it is directly related to the accuracy of the system (Mirza et al. 2018).

5.5.2 Model Architecture

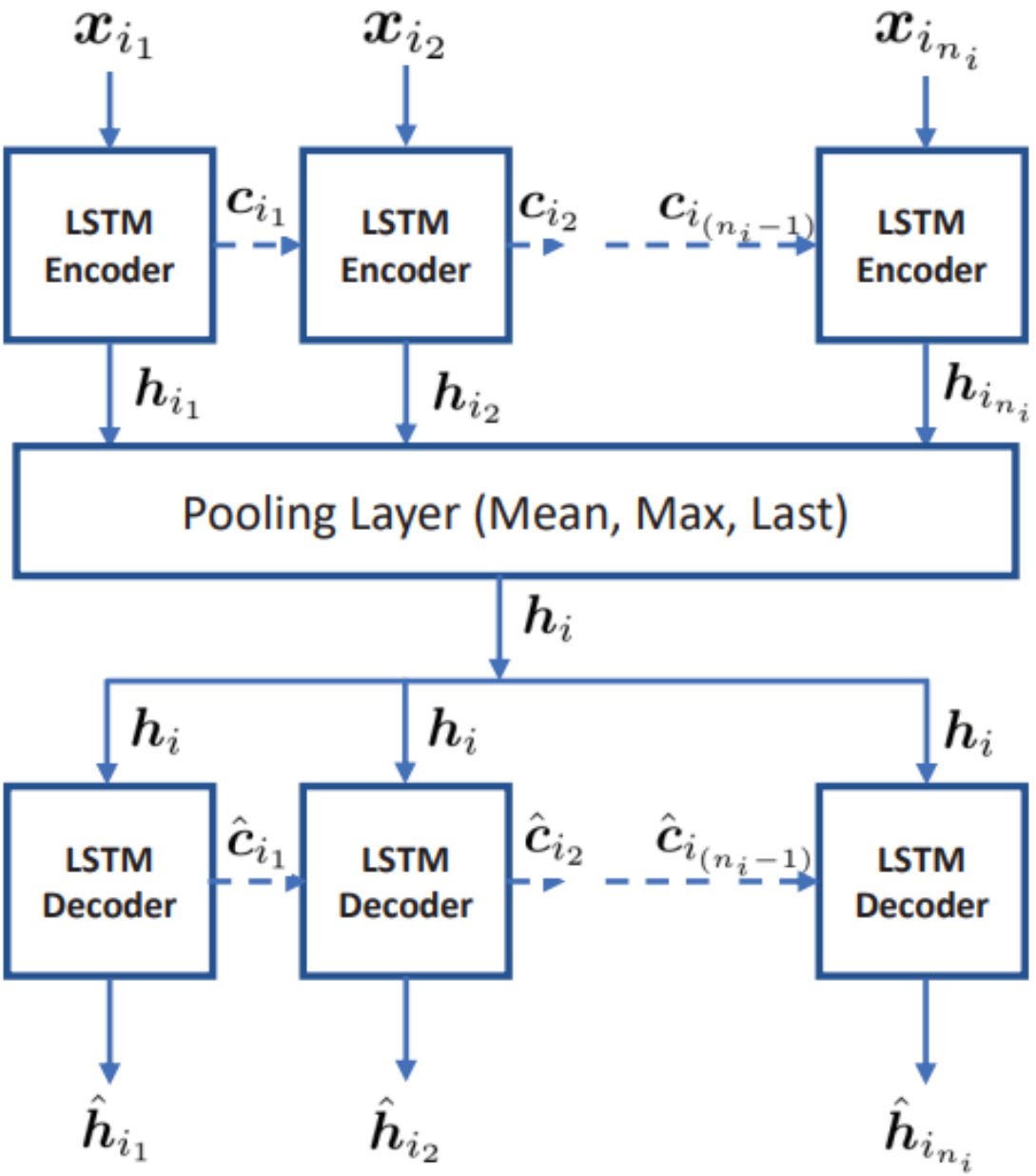


Figure 5.8: Detailed Description Of The LSTM Sequential Autoencoder Model (Mirza et al. 2018)

They randomly split the data set into training and test sets with percentage 90 and 10, respectively. Among the training set, 20k of connections are chosen randomly to be used in their experiments. For all the splits, anomaly occurrence rate remains the same. As their training method, Adam is employed with default parameters. The objective function is mean squared error. Batch size is chosen as 64. All the LSTM autoencoders have a unit size of 64 at both encoder and decoder layers. Sigmoid activation is used at the output layer.

5.5.3 Model Evaluation

For all the experiments, 20k of data is split into training and test sets with size 16000 and 4000, respectively. The training set is further split into training and validation sets with 80/20 ratio. First, we compare various systems with the proposed LSTM autoencoders. They also added more layers in the proposed algorithm, 2 layers each in encoder and decoder part, and call it Deep Auto-LSTM networks. For the case of the RNNs, the LSTM network, the GRU network, and the bidirectional LSTM networks all with one layer is applied separately with linear regression at the end. In addition, feed-forward neural networks with one layer are trained with the SGD algorithm. they used the validation set to obtain the receiver operating characteristics (ROC) and choose the threshold T corresponding to the best AUC score. Then, the test set is evaluated with the fixed threshold T and f1 scores are evaluated.

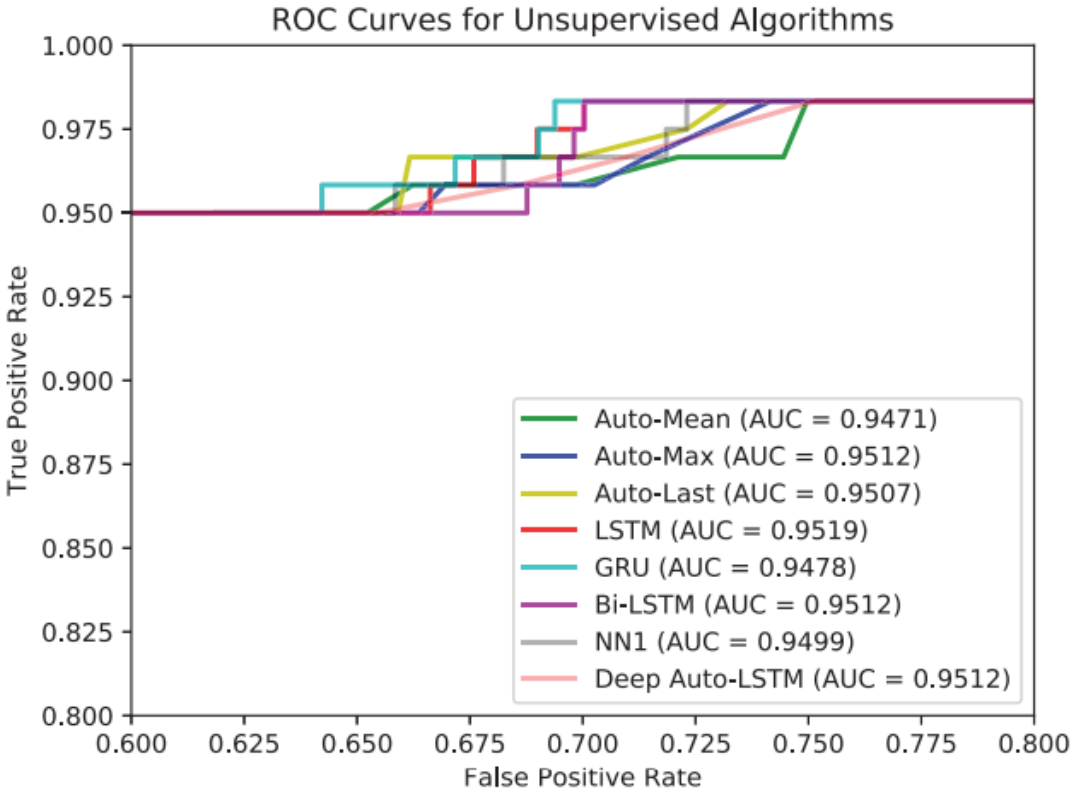


Figure 5.9: ROC Curves For The Proposed Sequential LSTM Autoencoders (Mirza et al. 2018)

Algorithms	Threshold	f1-score	AUC Score
LSTM	0.008	0.8409	0.9519
GRU	0.006	0.8102	0.9478
Bi-LSTM	0.008	0.8461	0.9512
NN1 - 1 layer	0.008	0.8352	0.9499
LSTM Autoencoder with Last Pooling	0.076	0.8409	0.9507
LSTM Autoencoder with Max Pooling	0.076	0.8538	0.9512
LSTM Autoencoder with Mean Pooling	0.079	0.8072	0.9471
Deep Auto LSTM	0.076	0.8538	0.9512

Tableau 5.4: Performance Metrics Table (Mirza et al. 2018)

5.6 Summary of the related work

Study	Algorithm	Method	Dataset	Acc	Observation
(Mirza et al. 2018)	Deep learning	LSTM (RNN) autoencoder	ISCX IDS 2012	95	Very good acc and It can detect more unknown intrusions then mentioned in the dataset
(Aung et al. 2018)	Hybrid	K-means + KNN	KDD CUP 99	98	Very good acc and It can detect more unknown intrusions then mentioned in the dataset
(Almi'ani et al. 2018)	Deep learning	Self Organizing Maps	NSL-KDD	83	Simple to use but time consuming algorithm
(Waskle et al. 2020)	Machine learning	PCA + Random forest	KDD	96	Limited to the dataset anomalies
(Jing et al. 2019)	Machine learning	SVM	UNSW-NB15	86	Limited to the dataset anomalies and low performance

Tableau 5.5: Comparative table between the related works.

5.7 Results discussion

In this survey we discuss 5 solutions of the network intrusion detection systems. (Jing et al. 2019) proposed an SVM model which gave a good score, but it is less preferment than (Waskle et al. 2020)'s model even though they both used machine learning algorithm but trained it in different datasets which lead us to conclude that PCA + random forest is a better solution than SVM, also that UNSW-NB15 dataset is better than the variations of the KDD datasets since it's richer and newer.
(Almi'ani et al. 2018) proposed a solution using self organizing map which gave promising results but the solution is weak compared to the others and much time consuming. In the other hand, (Aung et al. 2018) and (Mirza et al. 2018) both proposed different architectures (LSTM autoencoder, K-means + KNN) with very high accuracy rate with the ability of recognizing unknown anomalies in the network since both are trained in an unsupervised manner.
The best solution is to use either LSTM autoencoder or K-means + KNN for the model with the UNSW-NB15 dataset.

Chapter 6

Conclusion

Nowadays, every field is concerned with the network security more than ever, which led to the massive attention for the creation of various intelligent systems or solutions for the prevention of the assaults.
Every year new methods are proposed in this concern using machine learning and deep learning for network analysis, anomaly detection and flaws prevention, which, with the advancement of the artificial intelligence methods, achieved a great results and can go even beyond by detecting new types of attacks never seen before.
In this paper we covered the basics of this fields along side with the old and new aspects and proposed solutions to find the best ways to secure any network information and achieve a more safe data transfer and privacy.

Part III

Our Solution

Chapter 7

Solution design

7.1 General Architecture

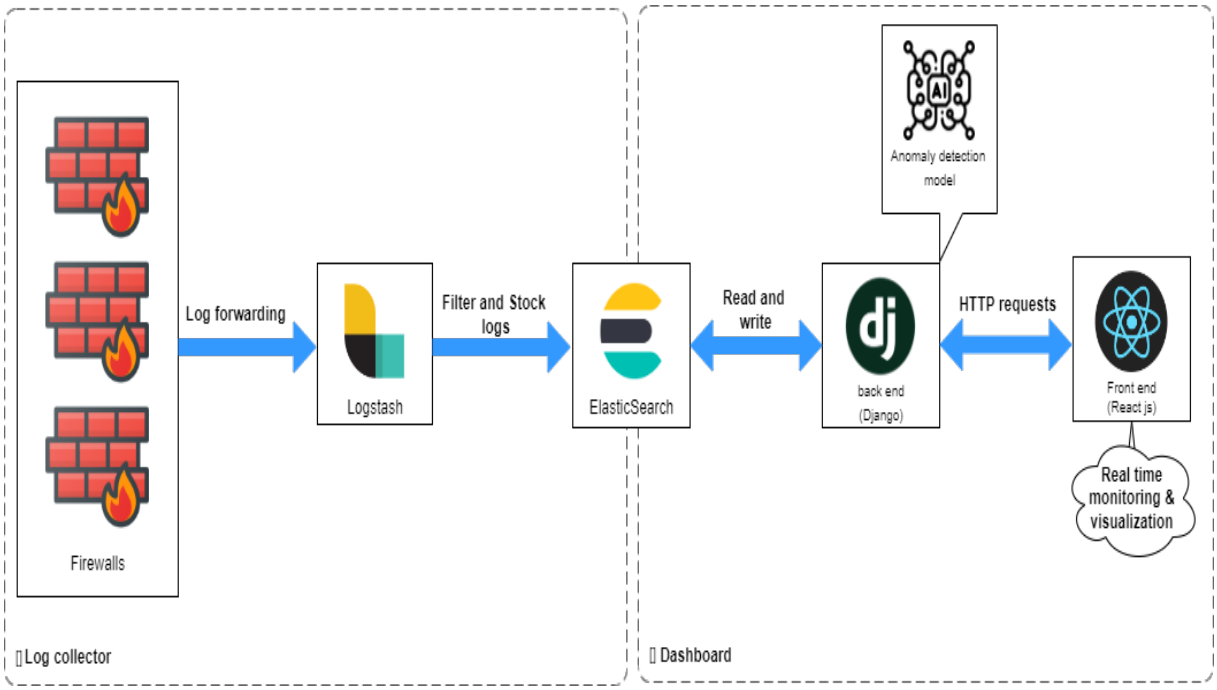


Figure 7.1: General Architecture

In this architecture we have two main sections. The first is consisted of several firewalls that forward network logs to the logstash instance, which filter and structure this logs and store them in the elasticsearch database.The second is a web application (django + reactjs) that read this logs, perform analysis and visualise them all this with an intelligent model integrated in the back end part to perform anomaly detection over logs.

7.2 Use Case Diagram

In the next diagram we can see a summary of all the functionalities that an admin can do in the system from real time monitoring and filtering logs and visualizing them in statistical ways to anomaly detection.

Figure 7.2: Use Case Diagram

7.3 Component Diagram

We have both the Django component and Logstash component will use the database component to retrieve and insert data. The user device will use the web server to interact with Django.

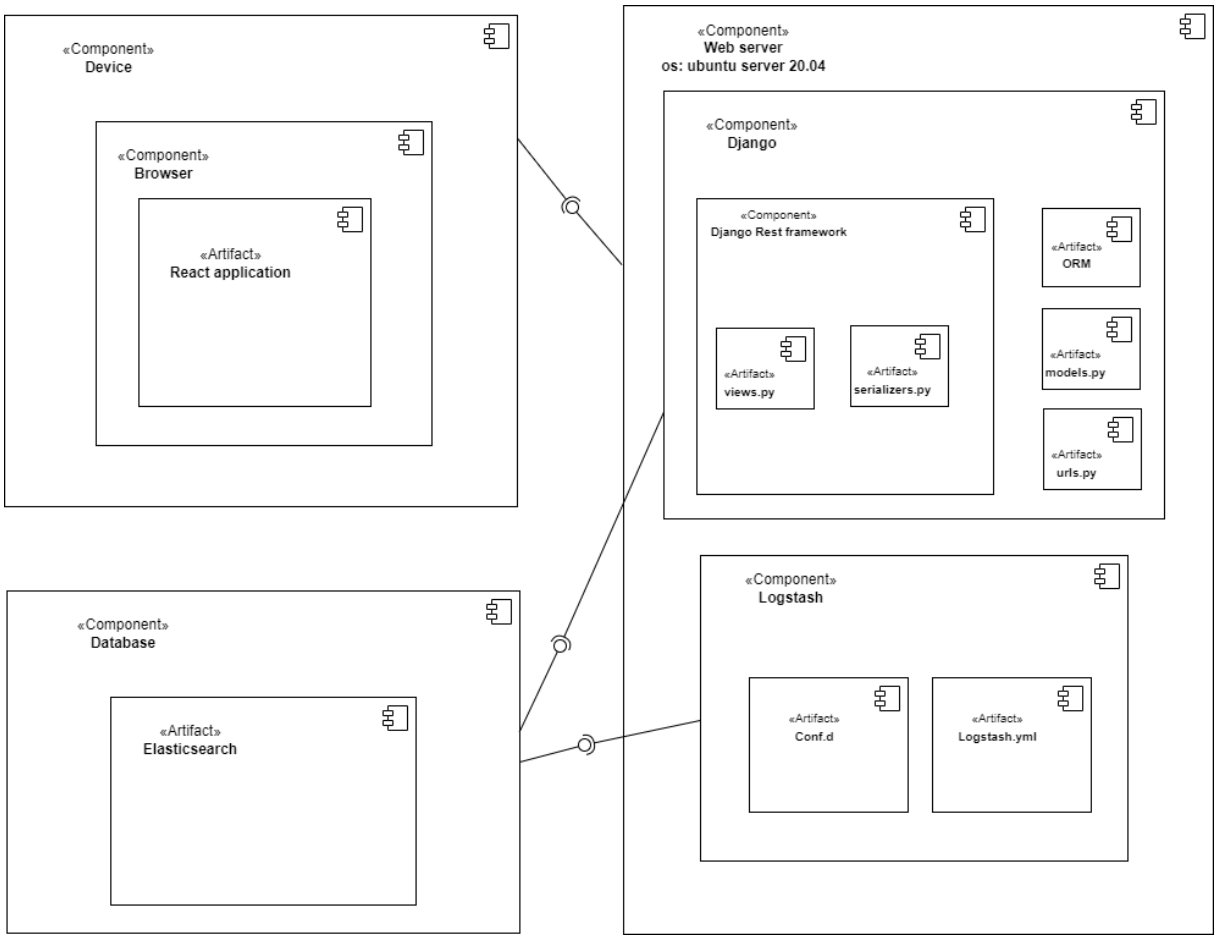


Figure 7.3: Component Diagram

7.4 Deployment Diagram

For the deployment we have one ubuntu server containing 3 virtual machines each on containing :

- Django application up and running containing both Wsgi app (for synchronous communication) and Asgi app (for asynchronous communication) connected to the Elasticsearch database using the django-elasticsearch-dsl package.
- Elk stack (Elasticsearch and logstash) enabled and its services activated.
- The build of the react application running over a node js server.

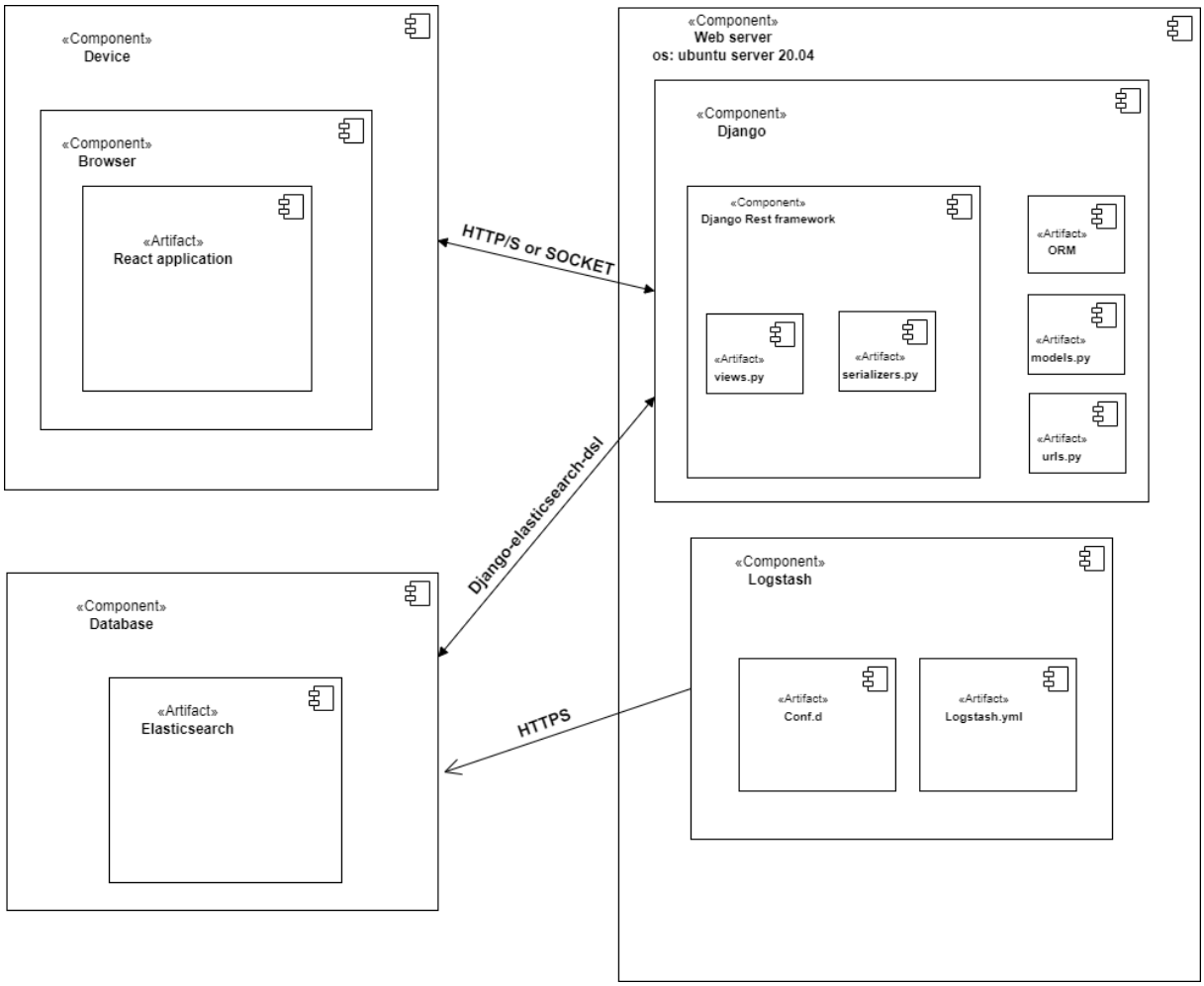


Figure 7.4: Deployment Diagram

7.5 Class Diagram

This diagram is used to explain how the database layer is composed, as we can see that each log is used in calculating the flows for the flow matrix and in calculating statistics, and can apply anomaly detection function on. The history contain multiple flow matrix which by it turn contain multiple flow that are calculated by grouping and aggregating the logs.

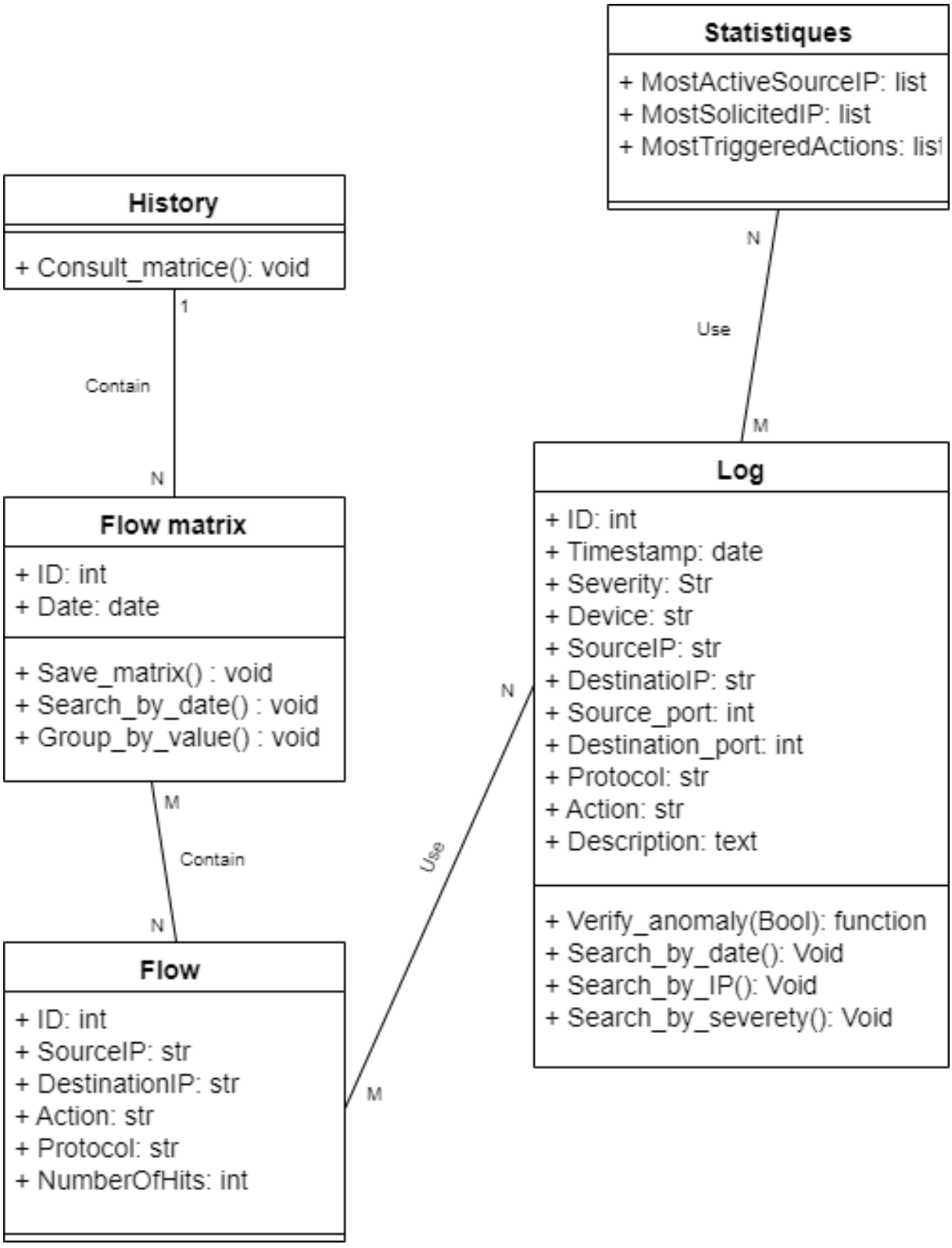


Figure 7.5: Class Diagram

Chapter 8

Technologies

In this chapter we will list and discuss all the technologies used in our solution implementation :

8.1 Artificial Intelligence Technologies

8.1.1 Environment

- **Python :**

In just a few years, Python has risen to prominence as a key language for data-processing applications. It's an object-oriented programming language that allows you to express complicated ideas in only a few lines of code. As a result, prototyping is substantially facilitated, which explains its widespread usage and popularity in machine learning (Sakshi Gupta 2022).

8.1.2 Libraries

- **TensorFlow :**

TensorFlow is a Python library for differentiable programming that is free and open-source. Beginners and pros alike will benefit from the library's collection of tools and information that make developing DL and ML models and neural networks simple. TensorFlow's architecture and foundation are adaptable, allowing it to run on a variety of platforms, including CPU and GPU. It is, however, at its finest when used with a tensor processing unit (TPU) (Sakshi Gupta 2022).

- **Keras :**

Keras is a Python open-source framework for creating and analyzing neural networks in deep learning and machine learning models. It can be used with Theano and TensorFlow, allowing you to start training neural networks with just a few lines of code. Keras is a beginner- and user-friendly library since it is modular, adaptable, and extendable. It also includes goals, layers, optimizers, and activation functions, making it a fully functional model for developing neural networks (Sakshi Gupta 2022).

- **Scikit-Learn :**

Scikit-learn is a free Python package that is generally considered a straight extension of SciPy. It is based on NumPy and SciPy. It was created primarily for data modeling and the development of both supervised and unsupervised machine learning algorithms (Sakshi Gupta 2022).

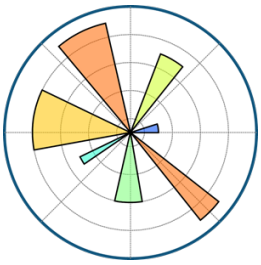
- **Pandas :**

Pandas is a Python framework for data research and analysis that allows programmers to create intuitive and seamless high-level data structures. Pandas, which is based on NumPy, is in charge of creating data sets and points for machine learning. Pandas combines one-dimensional (series) and two-dimensional (DataFrame) data structures, which allow it to be utilized in a wide range of fields, from science and statistics to finance and engineering (Sakshi Gupta 2022).

- **Numpy :**

NumPy is a well-known open-source numerical Python package. On arrays and matrices, it can execute a range of mathematical operations. It's one of the most widely used scientific computer libraries, and scientists frequently use it to analyze data. Its capacity to analyze multidimensional arrays, as well as handle linear algebra and Fourier transformation, makes it excellent for machine learning and AI applications (Sakshi Gupta 2022).

- **Matplotlib :**



Matplotlib is a data visualization framework for creating graphs and plots. It's a SciPy extension that can handle NumPy data structures as well as Pandas' complicated data models. Matplotlib can create high-quality and publish-ready diagrams, graphs, plots, histograms, error charts, scatter plots, and bar charts, despite its expertise being confined to 2D charting (Sakshi Gupta 2022).

8.2 Log Collector Technologies

- **Logstash :**

Logstash is the data collection pipeline tool. It gathers data inputs and sends them to Elasticsearch for indexing. It collects data from a variety of sources and makes it available for future use. Logstash can bring together data from a variety of sources and normalize it for use in your intended destinations. It enables you to cleanse and democratize all of your data in preparation for analytics and use case visualization (Taylor 2022).

- **Elasticsearch :**

Elasticsearch is a NoSQL (non-relational) database. It is created with RESTful APIs and is based on the Lucene search engine. It has a straightforward deployment process, high dependability, and is simple to administer. It also provides powerful searches for detailed analysis and centrally saves all data. It is useful for conducting a rapid document search. You may also use Elasticsearch to store, search, and analyze large amounts of data. It's mostly utilized as the backend engine for apps that fulfill search needs (Taylor 2022).

8.3 Web Platform Technologies

8.3.1 Back end

Django :

Django is the most widely used open-source Python web framework for developing dynamic websites. The Django framework's main goal is to make it easier to create complicated and massive database-driven websites. Django's growth was aided by the massive number of Python libraries available. The MVT (Model-View-Template) architectural design pattern is used in this web framework. For a newcomer, model building in Django may be challenging, but once you figure it out, you're set to go. The Django framework is used by several prominent apps such as Instagram, YouTube, and Reddit (Khatri 2022).

Django Project Structure :
The process of creating a Django project is pretty simple. On your Linux or Windows machine, just open your terminal or command prompt, navigate to the location where you want to start a project, and type the following command:
django-admin startproject myproject
It will make a folder called 'myproject' with the following structure:
manage.py/myproject settings.py urls.py wsgi.py myproject/ init.py
When you create a Django project, it will come with some basic files, as previously mentioned. These are the files:

- **manage.py:** This file allows you to use the command line to communicate with your project.
- **init.py:** It's a Python module. This file is called whenever we import a package or a module from a package.
- **settings.py:** This file has all of the website's settings. It comprises details such as new application registration, static file location, database configuration settings, and so on.
- **urls.py:** This file contains all of the project's links as well as the functions to call.
- **wsgi.py:** This file facilitates communication between your Django project and the webserver. It also aids in the deployment of your project using WSGI.
- **models.py:** In this file we implement the models that are the real deal that makes it work for the back-end. In Django, the models are the data carriers, and they're linked to the database, which stores all of the data that comes in from a user or is predefined once. Django makes use of ORM (Object-Relational-Mapping),

which allows programmers to construct a database using the Python programming language.

- **views.py:** This file contains classes that deal with the logic. It fetches the data from the Model and sends it to the front end so that a user can interact with it.

Django Rest Framework :

The Django Rest Framework allows you to develop RESTful APIs : A simple method of transferring data between an interface and a database. It isolates the user interface from the data storage and sends a.json file between the user and the database (Khatri 2022).

Why Django ? :

- It can quickly construct apps, and the build quality of Django-based applications is outstanding (Khatri 2022).
- It has a number of built-in features that set it apart from other Python frameworks, like admin authentication, site mapping, content administration, and much more (Khatri 2022).
- Security and the ease of Database management.
- The possibility of integrating an artificial intelligence model.
- It is a Python framework, so it has many libraries in inheritance (Khatri 2022).

8.3.2 Front End

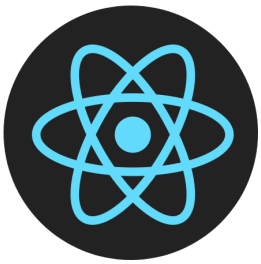
8.3.2.1 Environment

- **Java Script :**

JavaScript is a text-based programming language that allows you to construct interactive web pages on both the client and server sides. Where HTML and CSS provide structure and design to web pages, JavaScript adds interactive components that keep users engaged (Deshpande 2022).

8.3.2.2 Libraries

- **React JS :**

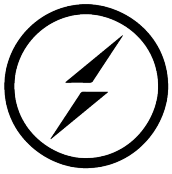


React is a UI development library based on JavaScript. It is administered by Facebook and an open-source developer community. Despite the fact that React is more of a library than a language, it is frequently utilized in web development. The library debuted in May 2013 and has since grown to become one of the most widely used frontend libraries for web development (Deshpande 2022).

- **Bootstrap :**

Bootstrap is an open-source CSS framework for creating mobile-friendly websites. CSS design templates for JavaScript typography, forms, buttons, navigation, and other interface elements are included (Deshpande 2022).

- **Web Socket :**



WebSockets is a next-generation bidirectional communication mechanism for web applications that operates over a single socket and is offered in HTML 5 compatible browsers through a JavaScript interface (Deshpande 2022).

- **Chart JS :**

Chart.js is a free JavaScript toolkit that allows you to create HTML charts. It is one of the most basic JavaScript visualization packages, it comes with a lot of predefined charts which describe and visualize the best your data. (Deshpande 2022).

Chapter 9

Log Collector

9.1 Why Logstash?

We utilize Logstash because it offers a collection of plugins that can be quickly attached to different targets in order to collect logs from them. Additionally, developers may easily edit, truncate, or change data streams using Logstash's very powerful template language.

9.2 Why Elasticsearch?

With the help of Elasticsearch, you can store, search, and analyze massive amounts of data fast and in close to real-time, with results arriving in milliseconds. Because it searches an index rather than the text itself, it can produce quick search results.

9.3 Installation

The log collector consist of two main sections. The first is a Logstash instance that is linked to the second which is Elasticsearch database. In this installation we had a ubuntu 20.04 server operating system.

9.3.1 Elasticsearch installation

Firstly we need to make sure that we have the latest version of Java, then we need to get the GPG key to get the official versions of Elasticsearch

```
$ curl -fsSL https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
```

after that we need to get the Elasticsearch package

```
$ sudo apt install elasticsearch
```

once downloaded we need to enable and start the Elasticsearch instance

```
$ sudo systemctl start elasticsearch
```

```
$ sudo systemctl enable elasticsearch
```

now we can verify if Elasticsearch is running by executing :

```
$ curl -X GET "localhost:9200"
```

```
Output
{
  "name" : "Elasticsearch",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "qqhFHPigQ9e2lk-a7AvLNQ",
  "version" : {
    "number" : "7.7.1",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "ef48eb35cf30adf4db14086e8aabd07ef6fb113f",
    "build_date" : "2020-03-26T06:34:37.794943Z",
    "build_snapshot" : false,
    "lucene_version" : "8.5.1",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

Figure 9.1: Elasticsearch Cluster

9.3.2 Logstash Installation

Since we have already installed the GPG key we can download directly the Logstash package

```
$ sudo apt install logstash
```

once the installation is finished we need to enable the Logstash instance

```
$ sudo systemctl start logstash
$ sudo systemctl enable logstash
```

if the installation doesn't return any error then we installed Logstash successfully.

9.4 Configuration

9.4.1 Elasticsearch Configuration

We need to do some modification in the "/etc/elasticsearch/elasticsearch.yml" file, which contains all the configuration of the database. Since we are not using multiple clusters,

then the configuration will be only for the the network part.
Elasticsearch listens to all incoming traffic on port 9200. we want to limit outside access to your Elasticsearch instance to prevent outsiders from reading our data

```
. . .
# ---------------------------------- Network -----------------------------------
#
# Set the bind address to a specific IP (IPv4 or IPv6):
#
network.host: localhost
. . .
```

Figure 9.2: Elasticsearch Configuration File

since both the Logstash instance and the Django are in the same server with Elasticsearch, so we gave access to only localhost other wise we can soecify the ip address.
After that we need to allow the cors headers for the localhost applications by adding the next lines in the elasticsearch.yml file :
http.cors.enabled: true
http.cors.allow-origin: /https?://(localhost)?(127.0.0.1)?(:[0-9]+)?/
http.cors.allow-methods: OPTIONS, HEAD, GET, POST, PUT, DELETE
http.cors.allow-headers: Authorization,
X-Requested-With,X-Auth-Token,Content-Type, Content-Length

9.4.2 Logstash Configuration

In Logstash there is a configuration file where you define the pipeline which is the reason why we used Logstash

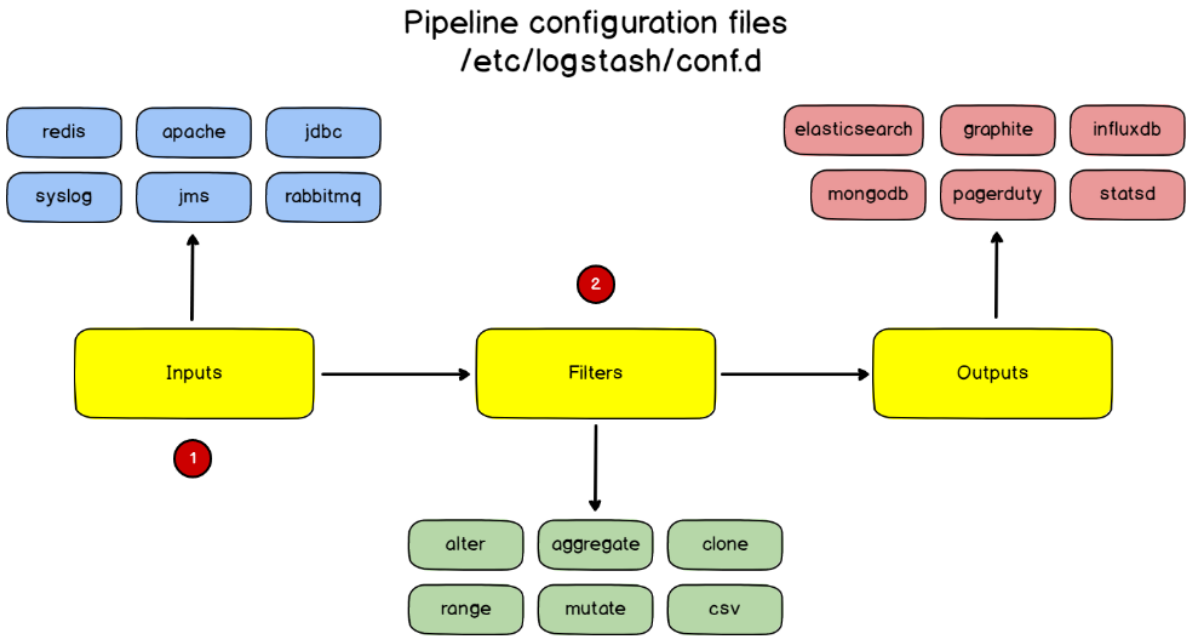


Figure 9.3: Logstash Pipeline

The pipeline contains 3 plugins :

- **Input plugin :** Where to listen to data from.
- **Filter plugin :** The transformation to apply on the data to structure it.
- **Output plugin :** Where to send the data after filtering.

Since we are listening to the firewall logs which are transferred over the UDP protocol, the input plugin configuration will be like this:

Figure 9.4: Logstash Input Plugin

In the other hand we want to store the logs in the Elasticsearch database so our output plugin will look like this:

Figure 9.5: Logstash Output Plugin

After this we will face the problem of parsing this incoming logs

Parsing logs problems:

- The logs are in unstructured format, they are human friendly and not for machines to understand.
- There is multiple form of logs because it depends on the firewall os and the message itself.
- Not all the logs contain the same informations.
- Using a lot of parsing can affect the performance of the pipeline.

The goal of parsing is to achieve this result:

```
From this:
 Teardown udp connection 3038845860 for Lan-Wan:10.30.1.231/63408 to SH-AEB-LAN:10.114.106.13/53 duration 0:02:11 bytes 126
....
to this:
 Receive_Time                                          3/22/22 1:44:59 PM
Severity                                                   Informational
Event_Type_ID                                                     302016
Event_Name                                                   Teardown UDP
Device                                                   SH-AVL-AEB-ASA-01
Source                                                       10.30.1.231
Source User Identity                                                 NaN
Source_Service                                                 udp/63408
Destination                                                10.114.106.13
Destination FQDN                                                     NaN
Destination_Service                                            udp/53
Action                                                       teardown
Risk Rating                                                          NaN
Description            Teardown udp connection 3038845860 for Lan-Wan...
Event_ID                                                  391946475631
Name: 1, dtype: object
 ........
```

Figure 9.6: Filtering Expected outcome

For this the solution is to use regular expressions (grok filter plugin), and fit them to parse all different types of logs. Since the firewall we are using is cisco-asa, we will create a patterns directory and then import it in our filter plugin :

patterns/patterns.txt

```
#== Cisco ASA ==
CISCO_TAGGED_SYSLOG ^<%{POSINT:syslog_pri}>%{CISCOTIMESTAMP:timestamp}( %{SYSLOGHOST:sysloghost})?: %%{CISC
CISCOTIMESTAMP %{MONTH} +%{MONTHDAY}(?: %{YEAR})? %{TIME}
CISCOTAG [A-Z0-9]+-%{INT}-(?:[A-Z0-9_]+)

# Common Particles
CISCO_ACTION Built|Teardown|Deny|Denied|denied|requested|permitted|denied by ACL|discarded|est-allowed|Drop
CISCO_REASON Duplicate TCP SYN|Failed to locate egress interface|Invalid transport field|No matching connec
CISCO_DIRECTION Inbound|inbound|Outbound|outbound

# ASA-6-106015
CISCOFW106015 %{CISCO_ACTION:action} %{WORD:protocol} \(%{DATA:policy_id}\) from %{IP:src_ip}/%{INT:src_por

# ASA-6-302013, ASA-6-302014, ASA-6-302015, ASA-6-302016
CISCOFW302013_302014_302015_302016 %{CISCO_ACTION:action}(?: %{CISCO_DIRECTION:direction})? %{WORD:protocol

# ASA-7-710001, ASA-7-710002, ASA-7-710003, ASA-7-710005, ASA-7-710006
CISCOFW710001_710002_710003_710005_710006 %{WORD:protocol} (?:request|access) %{CISCO_ACTION:action} from %

#== End Cisco ASA ==
```

Figure 9.7: Grok Patterns For Cisco ASA Firewall Logs

```
filter {
    # Extract fields from the each of the detailed message types
    # The patterns provided below are included in Logstash since 1.2.0
    grok {
      patterns_dir => "./patterns/"
      match => [
        "message", "%{CISCOFW106001}",
        "message", "%{CISCOFW106006_106007_106010}",
        "message", "%{CISCOFW106014}",
        "message", "%{CISCOFW106015}",
        "message", "%{CISCOFW106021}",
        "message", "%{CISCOFW106023}",
        "message", "%{CISCOFW106100}",
        "message", "%{CISCOFW110002}",
        "message", "%{CISCOFW302010}",
        "message", "%{CISCOFW302013_302014_302015_302016}",
        "message", "%{CISCOFW302020_302021}",
        "message", "%{CISCOFW305011}",
        "message", "%{CISCOFW313001_313004_313008}",
        "message", "%{CISCOFW313005}",
        "message", "%{CISCOFW402117}",
        "message", "%{CISCOFW402119}",
        "message", "%{CISCOFW419001}",
        "message", "%{CISCOFW419002}",
        "message", "%{CISCOFW500004}",
        "message", "%{CISCOFW602303_602304}",
        "message", "%{CISCOFW710001_710002_710003_710005_710006}",
        "message", "%{CISCOFW713172}",
        "message", "%{CISCOFW733100}"
      ]
    }
}
```

Figure 9.8: Logstash Filter Plugin Configuration

Chapter 10

Machine Learning Model

In this chapter we will speak in detail about the deep learning model for anomaly detection.

10.1 Data

The logs stored in the Elasticsearch database generated from the firewall are in this form:

Figure 10.1: Log Format

10.1.1 Problem

The problem here is that the data collected from the company is not labeled, and there is less abnormal records than the normal ones (imbalanced dataset) since the attacks are a rare event in the network compared to the normal traffic.

10.1.2 Solution

I used a public dataset called "UNSW-B15" (The raw network packets of the UNSW-NB 15 dataset was created by the IXIA PerfectStorm tool in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) for generating a hybrid of real modern normal activities and synthetic contemporary attack behaviours) and did some data augmentation on it to fit the company firewall logs.

10.2 Feature Selection

For the best training i only used the common features in the UNSW-B15 and the company's logs, that i estimated had an impact of the decision making, whether it's an attack or not. This features are:

- Receive Time
- Protocole
- Source Port
- Destination Port
- Duration
- Source Bytes
- Destination Bytes
- Source Packets
- Destination Packets
- Source to destination time to live
- Destination to source time to live

I added some other helping features that can add information to our data about the attacks:

- smeansz: Mean of the row packet size transmitted by the src.
- dmeansz: Mean of the row packet size transmitted by the dst.
- is-sm-ips-ports: If source and destination IP addresses and ports are equals.

- ct-srv-src: No of connections that contain the same service and source address in last 100 connections.
- ct-srv-dst: No of connections that contain the same service and destination address in last 100 connections.
- ct-dst-ltm: No of connections of the same destination address in last 100 connections.
- ct-src-ltm: No of connections of the same source address in last 100 connections.
- ct-src-dport-ltm: No of connections of the same source address and the destination port in last 100 connections.
- ct-dst-sport-ltm: No of connections of the same destination address and the source port in last 100 connections.
- ct-dst-src-ltm: No of connections of the same source and destination address in last 100 connections

10.3 Preprocessing

At this step we have two types of data (numerical, categorical).

10.3.1 Numerical Values

First of all we need to remove the outliers, any value that is beyond the percentile of 95 and greater than 10 times the median is considered as outlier.

```
for feature in df_numeric.columns:
    if df_numeric[feature].max()>10*df_numeric[feature].median() and df_numeric[feature].max()>10 :
        df[feature] = np.where(df[feature]<df[feature].quantile(0.95), df[feature],
df[feature].quantile(0.95))
```

Figure 10.2: Outliers Removal Code

After that we need to standardize the numerical values to feed them into the model.

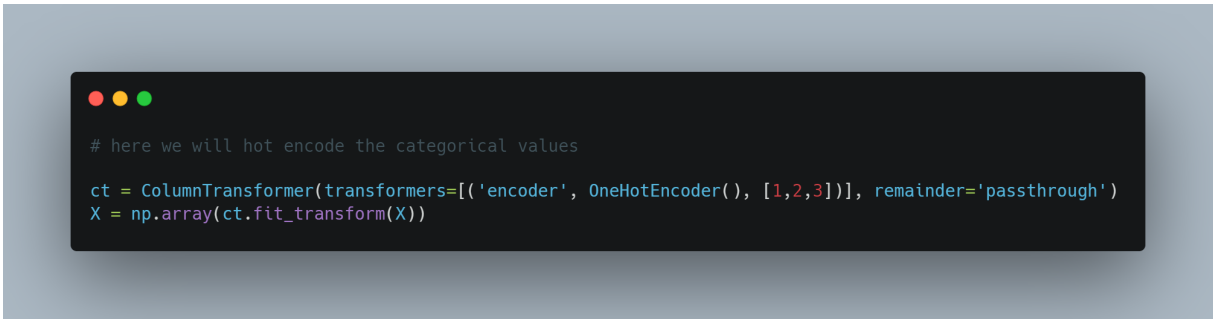
10.3.2 Categorical Values

For the categorical values we will take only the top 6 categories of each column and replace the rest (less appearing values) with "other" to prevent over-fitting.

```
for feature in df_cat.columns:
    # here we replace the less occurence values with Ohters
    if df_cat[feature].nunique()>6:
        df[feature] = np.where(df[feature].isin(df[feature].value_counts().head().index), df[feature],
'Ohters')
```

Figure 10.3: Categorical Values Reduction Code

Next we need to one hot encode this values.



```
# here we will hot encode the categorical values

ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [1,2,3])], remainder='passthrough')
X = np.array(ct.fit_transform(X))
```

Figure 10.4: Categorical Values One-hot-encoding Code

10.4 Model Training And Evaluation

10.4.1 Architecture

Since the inputs are time series, i used an LSTM Autoencoder for anomaly detection. The input have the shape [10 (number of time steps), 50 (log vector size)]. For both the decoder and encoder i used 3 symmetrical LSTM layers without returning the sequences , and i used a repeat vector layer just after the bottle neck.

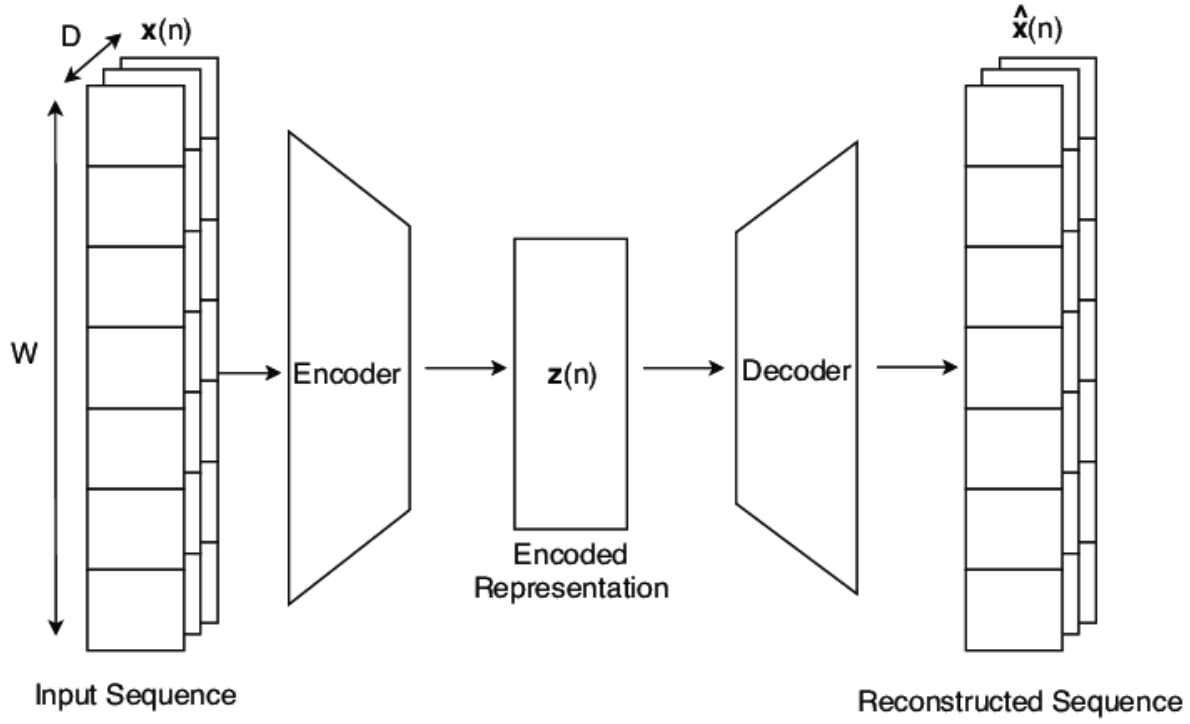


Figure 10.5: Model Architecture

10.4.2 Training

I trained the model in an unsupervised way on the normal data only. The model learns how to reconstruct normal traffic series with a minimal loss (mean squared error), when given an anomaly as input the reconstruction loss will increase. If we plot the reconstruction loss in the training part over the normal data and abnormal data we will have:

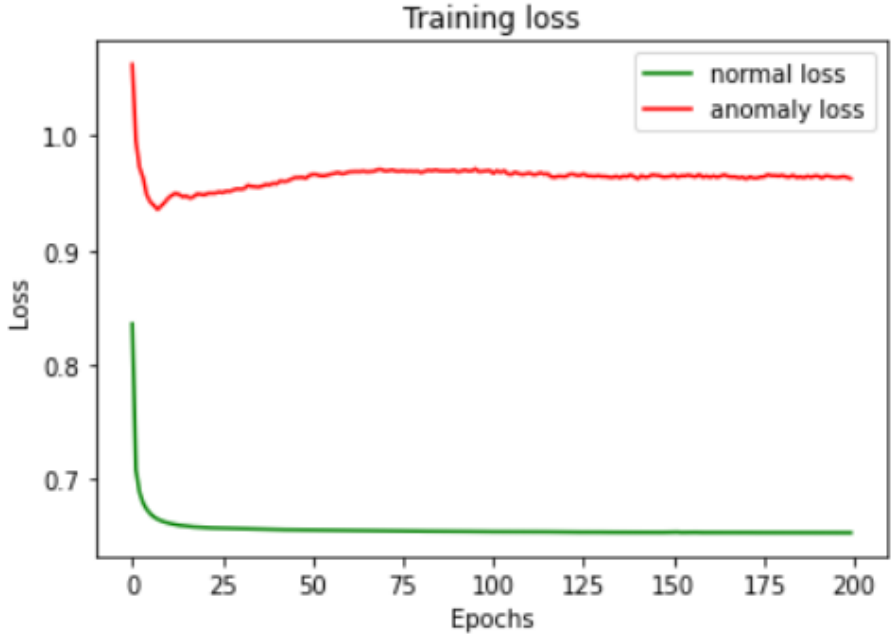


Figure 10.6: Loss Rate Over Normal an Abnormal Data

We can see that the normal data loss is under 0.7 and the abnormal data loss is beyond 0.9, As result, i have chosen 0.8 as the threshold value.

10.4.3 Evaluation

The power of an Autoencoder is in the fact that it can detect unknown attacks. Since it is trained on known normal traffic, it consider every thing different from normal traffic as abnormal. I evaluated the model over 175341 labelled log (normal and abnormal) containing 9 types of attacks (Reconnaissance, Backdoor, DoS, Exploits, Analysis, Fuzzers, Worms, Shellcode, Generic) and got 88 percent accuracy over testing data, and 96 percent over training data.

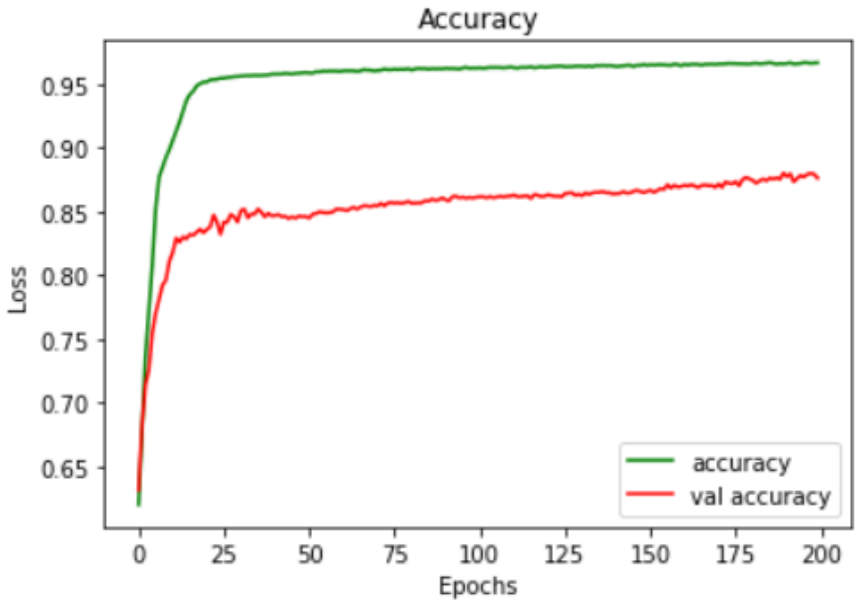


Figure 10.7: Accuracy Over Testing And Training Data

10.4.4 Exporting

```
# Save the model and its weights as h5 file
model.save("Anomaly_detector.h5")
model.save_weights("Anomaly_detector.weights.h5")
```

Figure 10.8: Exporting The Trained Model

Chapter 11

Web Application

In this chapter we discuss about the web application creation and design using the above technologies.

11.1 Project Creation

11.1.1 Django

The first step in starting a project while using Django is to create it. Django gives us a collection of tools for the administration during project creation, and the module that contains those capabilities is known as Django-admin. Firstly we need to create a project using the command **django-admin startproject projectname**. After that we need to create an app that contains our api's logic and AI model, this can be achieved by the command **django-admin startapp appname**. After that step we will have two main folders, the firs contain the project settings where we define all the configurations such as linking the database to our back end. The second contain our application files used in api definition, routing and testing, etc.

11.1.1.1 APIs

- **GET flow matrix:** returns a matrices containing the aggregated calculation for every flow.
- **GET statistics:** returns the statistics of the sources and destination IPs.
- **Real time monitor:** based on django channels (socket connection) for real time surveillance.
- **Get filters:** takes as input the filter (date range, IP address or severity) and returns a list of logs.
- **GET and POST history:** used to save or get a saved flow matrix or a list of saved flow matrices.
- **POST verify anomaly:** takes a list of logs as input and returns if an anomaly has occurred and where.

11.1.2 React Js

Firstly we need to create a react app using the NPM package by the command **npm create-react-app appName**. Once the app created we will have an src folder, this is where we can create our interfaces. For the connection with the back end i used axios package for HTTP requests and w3cwebsocket package for the real time monitor.

11.1.2.1 Interfaces

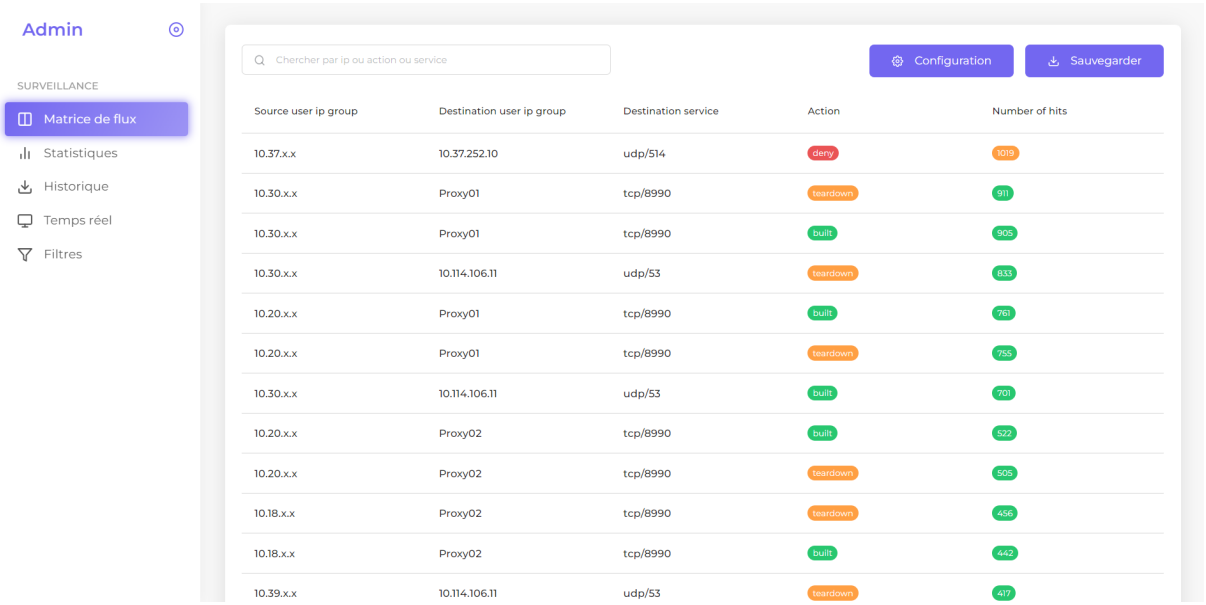


Figure 11.1: Flow Matrix Page

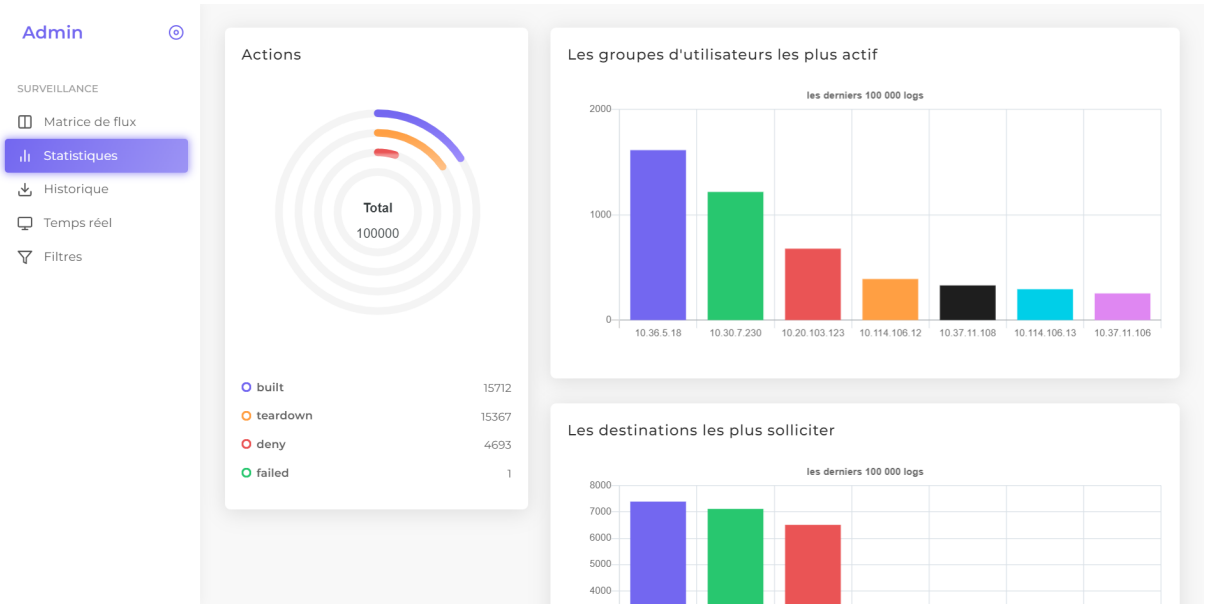


Figure 11.2: Statistics Page

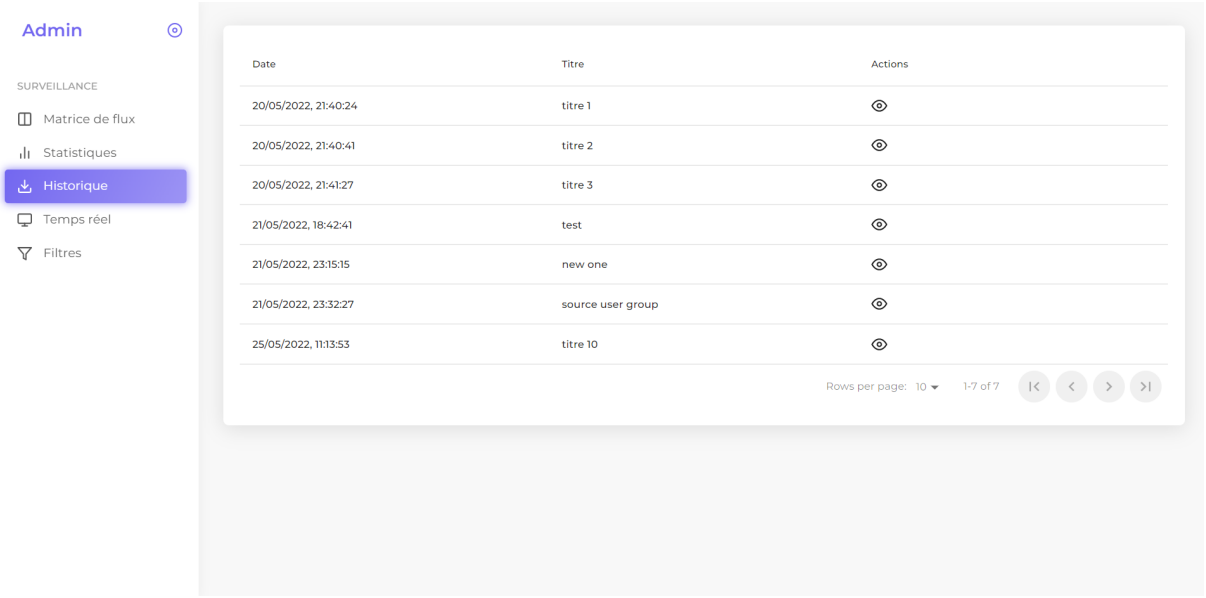


Figure 11.3: History Page

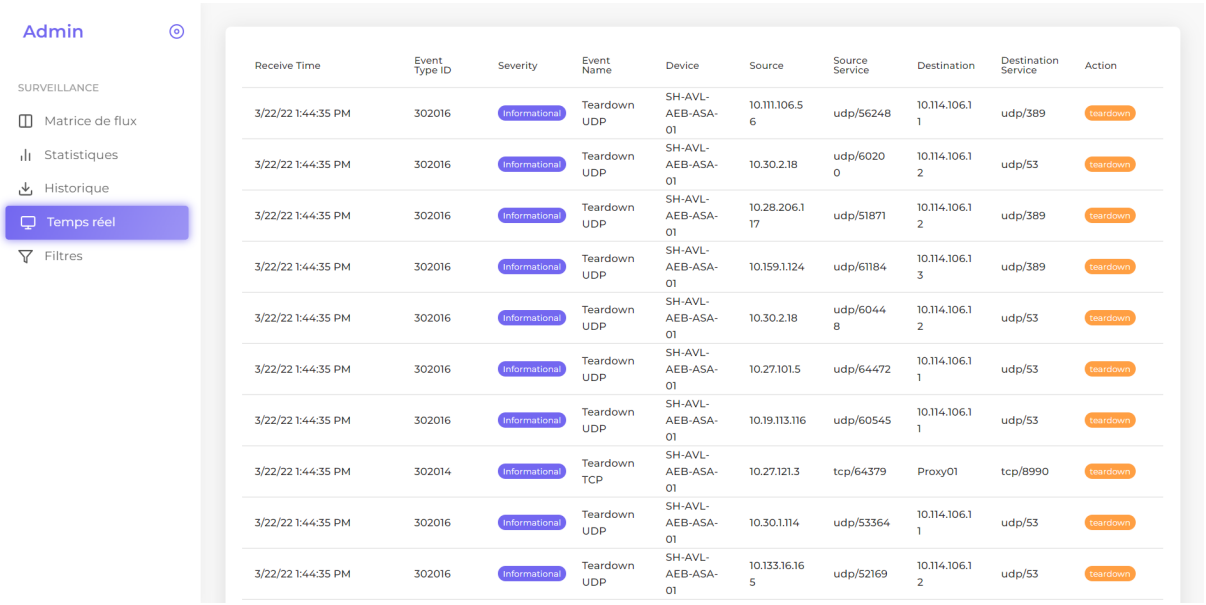


Figure 11.4: Real Time Monitoring Page

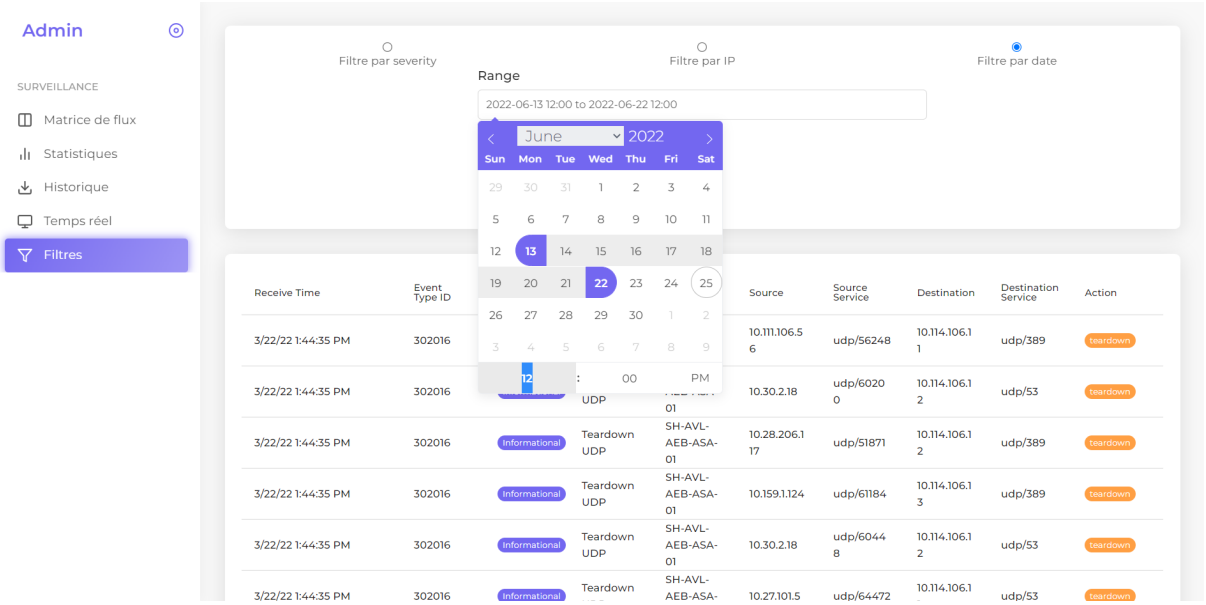


Figure 11.5: Filters Page

Part IV

Project Management And Conclusion

Chapter 12

Collaborative Work

12.1 Introduction

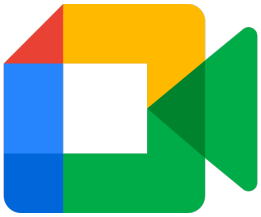
To make this project see the light and be successful, communication is an essential aspect. It help to share, collaborate and exchange knowledge and opinions to achieve the best possible outcome.

12.2 Tools

- **Git :**

Git is a technology that helps programmers monitor changes in any size of file by coordinating their efforts. Speed, data integrity, and support for distributed and non-linear processes (thousands of simultaneous branches operating on many platforms) are some of its objectives.

- **Google Meet :**



Google has created a video service called Google Meet (previously known as Hangouts Meet). One of two software options that take the place of Google Hangouts is this one. I conducted remote internet meetings and communications using Google.

- **Trello :**

Trello is an online project management tool, launched in September 2011 and inspired by Toyota's Kanban method. It is based on an organization of projects in boards listing cards, each representing tasks. I used it to organize the task of this project and define deadlines.

12.3 Project Monitoring

In terms of project monitoring, we spoke with the internship supervisor practically weekly and held meetings to assess the status of progress every two to three weeks. When we needed our university supervisor, he was always there to help, offer advice, and participate in online sessions with us.

Chapter 13

Project Management Methodology

13.1 Introduction

The use of project management approaches has grown in popularity during the last few years. Along with more conventional options like PMP or Prince 2, they also include agile approaches like SCRUM, Kanban, DSDM, and Extreme Programming.

The ability to standardize, structure, and organize work techniques is the main goal of project management approach. This promotes a consistent emphasis across all projects and enables us to replicate effective elements and learn from failures, leading to a process of continual progress. In other words, as it is applied, a technique is a wonderful instrument for increasing efficiency.
Benefits of using a project management methodology:

- Organizing project times
- Providing tools to estimate times and costs correctly
- Best management for project risks
- Improving the benefits over the costs relationship
- Better team work efficiency

13.2 Method Used

Since there are various methodologies, we have selected the Iterative and Incremental Project Management Methodology as the one that would work best with software development. With this methodology, we develop each phase, test it, and keep track of it before moving on to the next one, until the entire project is built.

Figure 13.1: Incremental Methodology

Chapter 14

Conclusion

In this project i solved on of the major problems in the world of networking, especially in the recent days with the appearance of IOT devices and the massive growth in the number of connected devices, network intrusion is more and more common these days, which resulted in such a great difficulty defining access lists particularly in big companies to prevent these unwanted intrusions. There is no need to go trough the long and exhausting process of verifying logs manually, this system can detect various types of attacks and tells you where it occurred with high detection rate. For my benefit, i have opened new perspectives and learned a lot by using new technologies and facing new challenges in many fields such as computer science, professional world and collaborative work.

14.1 Future Work

There is still a lot to do in this project and here is some possible future improvement:

- Fit the system to work with even more types of logs than the Cisco-asa firewall logs.
- Using multiple clusters for the log collector to go even beyond the actual performance.
- Using micro services for better availability and performance.
- Upgrading the model to detect even the attack type.
- Network architecture optimization model.

Bibliographie

Ahmed, Mohiuddin, Abdun Naser Mahmood, and Jiankun Hu (Nov. 2015). "A survey of network anomaly detection techniques". In: *Journal of Network and Computer Applications*.

Almi'ani, Muder et al. (2018). "Intelligent Intrusion Detection System Using Clustered Self Organized Map". In: *IEEE*.

Aung, Yi Yi and Myat Myat Min (2018). "Hybrid intrusion detection system using K-means and K-nearest neighbors algorithms". In: *IEEE*.

bitnine (2017). *TOP 10 Open Source Big Data Databases*. en. URL: https://bitnine.net/blog-useful-information/top-10-open-source-big-data-databases/?ckattempt=1 (visited on 06/05/2022).

Deshpande, Chinmayee (2022). *The Best Guide to Know What Is React*. en. URL: https://www.simplilearn.com/tutorials/reactjs-tutorial/what-is-reactjs (visited on 06/21/2022).

Gupta, Sakshi (2022). *15 Best Python Libraries for Machine and Deep Learning*. en. URL: https://www.springboard.com/blog/data-science/python-libraries-for-machine-learning/ (visited on 06/20/2022).

Gupta, Shashank (2018). *Ten Machine Learning Algorithms You Should Know to Become a Data Scientist*. en. URL: https://medium.com/towards-data-science/ten-machine-learning-algorithms-you-should-know-to-become-a-data-scientist-8dc93d8ca52e (visited on 05/14/2022).

He, Shilin et al. (July 2021). "A Survey on Automated Log Analysis for Reliability Engineering". In: *ACM Comput. Surv. 54, 6, Article 130*.

Jing, Dishan and Hai-Bao Chen (2019). "SVM Based Network Intrusion Detection for the UNSW-NB15 Dataset". In: *IEEE*.

Khalid, Samina et al. (Aug. 2014). "A Survey of Feature Selection and Feature Extraction Techniques in Machine Learning". In: *Science and Information Conference*.

Khatri, Vinay (2022). *What is Django?* en. URL: https://www.techgeekbuzz.com/blog/what-is-django-advantages-disadvantages/ (visited on 06/21/2022).

Kwon, Donghwoon et al. (Aug. 2017). "A survey of deep learning-based network anomaly detection". In: *Springer Science+Business Media, LLC 2017*.

Logic, Sumo (2022). *log-file*. en. URL: https://www.sumologic.com/glossary/log-file/ (visited on 03/16/2022).

Ltd, Humio (Sept. 2021). *Log Files Explained*. en. URL: https://www.humio.com/glossary/log-file/ (visited on 03/16/2022).

Mirza, Ali H. and Selin Cosan (May 2018). "Computer Network Intrusion Detection Using Sequential LSTM Neural Networks Autoencoders". In: *2018 26th Signal Processing and Communications Applications Conference (SIU)*.

Mishra, Aditya (2018). *Metrics to Evaluate your Machine Learning Algorithm*. en. URL: https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234 (visited on 03/25/2022).

Point, Tutorials (2020). *Time Series - LSTM Model*. en. URL: tutorialspoint.com/time_series/time_series_lstm_model.htm# (visited on 06/10/2022).

Sultana, Nasrin et al. (2017). "Survey on SDN based network intrusion detection system using machine learning approaches". In: *Springer Science+Business Media, LLC, part of Springer Nature 2018*, pp. 2–9.

Taylor, David (2022). *ELK Stack Tutorial: What is Kibana, Logstash and Elasticsearch?* en. URL: https://www.guru99.com/elk-stack-tutorial.html (visited on 06/20/2022).

Waskle, Mr Subhash, Mr Lokesh Parashar, and Mr Upendra Singh (2020). "Intrusion Detection System Using PCA with Random Forest Approach". In: *IEEE*.

Zhang, X. et al. (2019). "Robust log-based anomaly detection on unstable log data". In: pp. 807–817.