A

dissertation

on

Restructuring IDS: a new perspective

Submitted in partial fulfilment of the requirements for the award of degree of

Master of Technology

in

Computer Science and Engineering

Submitted by

Mridul Katta

(16535025)

Under the guidance of

Dr. Manoj Mishra

Professor, Dept. of Computer Science and Engineering

Dr. Durga Toshniwal

Associate Professor, Dept. of Computer Science and Engineering

INDAIN INSTITUTE OF TECHNOLOGY, ROORKEE

Department of Computer Science

May, 2018

AUTHOR'S DECLARATION

I declare that the work presented in this dissertation with title "Restructuring IDS : A new perspective" towards fulfilment of the requirement for the award of the degree of Master of Technology in Computer Science & Engineering submitted in the Department of Computer Science & Engineering, Indian Institute of Technology, Roorkee, India is an authentic record of my own work carried out during the period of May 2017 to May 2018 under the supervision of Dr. Manoj Misra, Professor, Department of Computer Science and Engineering, and Dr. Durga Toshniwal, Associate Professor, Department of Compute Science and Engineering, Indian Institute of Technology Roorkee, Roorkee, India. The content of this dissertation has not been submitted by me for the award of any other degree of this or any other institute.

Date: .....

Place: ROORKEE

MRIDUL KATTA
(16535025)
M.TECH (CSE)

CERTIFICATE

This is to certify that the statement made by the candidate is correct to the best of my knowledge and belief.

Date: .....

Place: .....

Sign: .....

Dr. Manoj Misra
(Professor)

Sign: .....

Dr. Durga Toshniwal
(Associate Professor)

IIT Roorkee

ACKNOWLEDGEMENTS

Dedicated to my family and friends, for standing by me through thick and thin, without whom I would not have gotten this far. I would like to express my sincere gratitude to my advisor Dr. Manoj Misra and Dr. Durga Toshniwal for the continuous support of my study and research, for his patience, motivation, enthusiasm and immense knowledge. His guidance helped me in all time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my study.

I am also grateful to the Department of Computer Science and Engineering, IIT Roorkee for providing valuable resources to aid my research.

MRIDUL KATTA

ABSTRACT

With Cloud and Internet among the essential pillars on which the world resides in this era. With the increasing advancements in technology, network speed and a need of greater, more reliable and feasible processing power, the dependency on these pillars is continuously growing, with growing concerns to security and privacy of the data. The former also stresses the need for faster servers and response time. With intrusions getting complex simple intrusion detection methods and, moreover, exclusive IDS are not an option anymore, there is a serious need of much more sophisticated methods; resulting in bulkier IDSs. The trade-off between resource utilisation & response time and security attracted a lot of researchers in recent years. The present scenario needs a resource-friendly IDS with no compromise with security, delivering good response time.

Table of Contents

List of Figures and Tables

Chapter 1 INTRODUCTION

With the ever-increasing network and its complex network management and network security is already an open area of research. Over the last decade, there has been a rapid increase in the dependency on the technology and networks, from news to shopping, from email to social feeds, all are being transmitted over the network to every day increasing receivers, leading to the enormous amount of data flowing within networks. Digitalization, accompanied by other factors like on-the-go need, instant share, instant access to file across the globe and on different devices, storage, etc. pushed for the need of what we call today as “Cloud”.

“Cloud”, more precisely, “Cloud Computing” refers to a network, with an on-demand and convenient access, of remote processing resources and servers, hosted for storing, managing and processing the data, over the internet or network, with least management or human supervision. Cloud computing is the answer to the present scenario needs from simple on-the-go access to heavy computations in research or medical fields, or just to store data by billions and billions of people across the world; by virtually collaborating various remote servers into a gigantic cost-effective processing resource. Thanks to Cloud Computing, now, the world has an access to all the divided processing power world has under one virtual roof. But as Juliet Marillier once said, “Nothing comes without a price” [1], with all the benefits of cloud it indulges some serious challenges towards security and privacy of all the data.

Recent years saw a great inclination of the world towards cloud and its computational efficiency. With advancements in the network, transmission speed and hardware, creamed with advanced machine learning and artificial intelligence algorithms, the complexity and the rate of intrusions have tremendously increased. Trivial means like authentication and their advancements like data encryption for intrusion detection are clearly not a choice anymore [20], and are broadly taken over by Intrusion Detection Systems (IDSs). Apart from IDS, firewalls are another apt and indeed one of the most common solutions to intrusions over the network. Firewalls are used to eavesdrop on every packet to match the same against a set of predefined rules and policies [1]. Naïvely speaking, the firewall can be coined as an initial or first level of the defence. Intrusion Detection System (IDS) and/or Intrusion Detection and Prevention System (IDPS) is the most widely accepted and invaluable resort for the threats to cloud systems. Besides detection as in IDS, IDPS can also defy or prevent the system against the attacks.

With intrusions evolving there is a need to restructure earlier IDSs, designed for a specific intrusion or signature, to cope up with prevalent and sundry intrusion categories. IDS running machine learning algorithm at its core aided with cloud's computational power has attracted a lot of researchers in recent years, and the systems are much more capable of adjusting to the present needs [21].

1.1 Intrusions

Attacks are oriented towards the vulnerabilities in the system, i.e., Integrity, confidentiality and availability, of the services and resources [2]. The intrusion, perhaps be intended to misuse stored data or to handicap the cloud system itself, i.e., flood the processing resources with huge number of packets from innocent host (zombie) [1], thus hindering the availability of the resources to authorized users [9] [2], and in the worst case could lead to Denial of Service (DOS), loss of availability of resources for intended users. This subsection describes some major intrusions/attacks [1] [9].

Insider attack: It covers attacks from within the organisation like frauds, attempts for unauthorised privileges and disclosure or changing information.

Flooding attack: This attack uses a couple of host machines as zombies and tries to flood the victim server with an immense number of packets, over the network, eventually, preoccupying all the resources and leaving none for the actual users. With VMs, the clouds are an easy target for flooding attacks, which may cause Denial-of-Service (DoS).

If the intended service is lost due to DoS attack it is termed as direct DoS attack, whereas, if another service(s) provided by the same server is lost while processing an immense number of packets due to flooding, it is termed as indirect DoS.

Root attacks: Here, the aim is to gain root level access, perhaps via user's account hacked by simple means like sniffing password or phishing attacks. The goal can be easily targeted by generating root shells via buffer overflow, for instance. These security breaches are hard to detect. In cloud systems, root access of VMs can be gained in a similar way via user's instance.

Port Scanning: The attacker scans for open ports and attacks the services running on the same, which can provide the attacker with MAC address, IP address and other network related details.

Attacks on a hypervisor or VM (Virtual Machine): A hypervisor can be thought of as a program that allows multiple copies of OS to run using single hardware. The attacker aims at gaining control over all the installed VMs by gaining control over the lower layer, the hypervisor.

Further, an increase in the usage of resources, because of compromised VMs and hypervisor can cause DoS attack [2]. “Zero-day vulnerability” (NIST[1]: National Vulnerability Database 2011) [3][4]) is an attack-prone zone, that can be used by the attacker, even before it is known to the developer of the software.

Backdoor channel attacks: It is classified as a passive attacker. Here, the attacker tries to attack user confidentiality via infected node. The hacker can make it worse by controlling victim’s resources to attempt a DDoS attack.

Some of these attacks can be taken care of via firewalls working at different levels. But for the attacks on VMs and hypervisor, flooding attacks and backdoor attacks, much more sophisticated methodologies are required [9].

For intrusion detection systems the intrusions are, generally, divided into four categories, namely, Denial-of-Service (DoS), Probe, User-to-Root (U2R) and Remote-to-Local (R2L) attacks, as per their frequency from high to low, respectively [19].

DoS: As the name suggests the attack floods the server with an enormous amount of fake packets obstructing the service from its authorized consumers.

Probe: This attack gathers the vulnerabilities within system or network by scanning ports and hosts within the network [23].

User to Root: In this type of attack the user tries to gain administrator or root authorization, basically, tries to gain full access and control over the system [13].

Remote to Local: In this attack, the remote host or user tries to gain local access to the physical machine on which its serving VM (Virtual Machine) is running [9].

Less frequent attacks don’t volunteer as excusable if successful they can be catastrophic.

1.2 Firewall

Firewalls secure the network from unauthorised access using some predefined policies to permit or block the packets as per their IP’s or network address, etc. Firewalls can be said as the initial level of defence against intrusions [1]. As firewall eavesdrop on every packet entering the network on the boundary itself, it protects from malware and malicious software, but have no measure to detect severe attacks like Probe, DoS, U2R and R2L [9].

[1] NIST: National Institute of Standards and Technology

1.3 Intrusion Detection System

IDS is currently the most appreciated way to deal with the intrusions [13] [9]. However, there are many types of IDSs based on varied methodologies to deal with diverse intrusion categories, with their own pros and cons. As of now, there is no proper classification of all the major IDS present. In fig. 2.1 we attempt to provide an overall classification of all the major IDS narrowed down to three classes, namely, monitored resources (the resources they monitor for intrusions, or the level at which they are placed), detection methodologies or techniques (algorithm running the IDS), and finally, response behaviour (i.e., to detect and inform to administrator (passive) or take measures to stop the intrusion (active)); for better understanding of the overall picture.

With intrusions getting complex and more frequent each day, traditional specialised IDSs, i.e., IDS capable of handling only one intrusion type or signature, cannot be used anymore. With improvements in intrusions creative and effective ways are needed to improve IDSs as well.

1.4 Motivation

The world, its networks and internet, with its dependencies is growing at a tremendous speed in the present era. Internet has become a major aspect of our lives.

With growing network speeds, the faster response time from servers is crucial, which in turn pushes for better and faster processing of data and packets. Earlier specialised IDS are no longer feasible, with intruders getting smarter and more frequent there is paramount need of intelligent IDS and creative ways to boost their performance and speed. With data leaks from big MNCs like Facebook [28], Twitter [29], sony [30], there is growing concern for individual's privacy and scams.

The last decade saw a great advancement in machine learning and artificial intelligence, many of these algorithms are now the core of IDS running worldwide, still, they possess a great trade off between security and response time, and it will be the same ever. Most of the researchers [3] [4] [5] [6] [9] [11] are focused on using the complete model, be it signature-based, anomaly-based or hybrid, on every packet, moreover, on complete network traffic, ignoring the fact that every request packet may not need same modules for intrusion detection. Additionally, in need of better intrusion detection accuracy more sophisticated methods or a collaboration of techniques, are in trend, which further exhaust resources.

On the other hand, there are researchers that intend to improve the service response of the cloud systems keeping security in the mind. One such approach is given by C. Mazzariello et al. [5], the authors proposed a system where the IDS is deployed at each physical machine along with the cloud controller, instead of a single IDS on the latter. The proposed system tries to take the advantage of divide-and-conquer principle; the performance of the affected VM (Virtual Machine) and the respective host (physical machine) will decrease leaving the rest unaffected. But keeping the diversity of attacks in mind, the secure network will opt for anomaly-based detection, which is heavy to run on each physical machine. This particular concept remains unexplored, as per this phase, in this report. Further, there are researchers using other means like multi-threading to boost the performance of the IDS [4]. But all ignoring the observation that every packet can be classified using the traffic flow, its origin etc. to provide necessary security checks instead of bombarding every packet with the complete heavy detection model.

Currently, all upcoming and running IDS packs clustering in some way or another in itself and this clustering provides the data that is further classified using different modules or classifiers [20] [22] [23] [9], fuzzy clustering in particular.

In this thesis, we propose a lightweight modular approach for IDS, which runs fuzzy c-means clustering (explained later) at its core and the clusters are further analysed and next module is selected as per the need of respective clusters, every cluster is not equipped with one or all classification modules, the results of experiments further prove our point.

1.5 Problem Statement

To design an IDS capable of handling most of the intrusions and at the same time is easy on resources, with a good average response time.

1.6 Thesis Organisation

Chapter 2 briefs about various IDSs, their pros and cons. Identifying research gaps and limitations.

Chapter 3 describes about the proposed model and all the algorithms used.

Chapter 4 provide details about the experiments and the results.

Chapter 5 concludes the work with future amendments.

Chapter 2 BACKGROUND AND LITERATURE SURVEY

Before discussing the various approaches and approaches, we first present the outline of various IDS, along with different classifications for deeper and quicker understanding of sundry categories.

2.1 Background

Based on the study [24] [25] [1] [27] [13], we summarise various IDS, along with our proposed classification (fig 2.1).

2.1.1 Monitored Resources

This classification is based on the monitored resources for intrusions by the IDS or the level at which IDS is programmed to work [1] [25].

Host-based IDS (HIDS): As the name suggests they work on host machines and detect any deviation from expected behaviour. They accomplish the task, by monitoring and analysing the normal behaviour of the machine via system logs, system files, system calls, network events, etc., and then, checking for any changes in the activities, to report to the concerned authority. Additionally, they may also map the suspicious activities against their knowledge as an intrusion. The efficiency of HIDS highly depends on the use of host machine. They can work very efficiently for static working behaviour, i.e., the machine that goes through the same set of activities daily. On the contrary, they may find a hard time working efficiently in mercurial or dynamic working environment. HIDS, for cloud computing, can be positioned at host machines, VM(s) or hypervisor. Insider attacks go unnoticed via HIDS, that is one major drawback [6].

Network-based IDS (NIDS): These systems detect intrusions by analysing the network and the packet traffic within the network. NIDSs protect all the hosts, VMs connected to the network from network related intrusions like DoS attack, port scanning or attempts to break into machines. Most widely used NIDSs use anomaly and/or signature-based detection techniques, inspecting every packet at IP and transport layer headers, entering the network. However, like HIDS, NIDS cannot detect any intrusions or attacks which are within the virtual network running entirely within the hypervisor [1].

Application-based IDS (AIDS): Lately, it was observed that many intrusions take place at the application level, which leads to specific IDS, termed as AIDS. Like HIDS they analyse log files, but unlike, HIDS they analyse application logs, not system logs. Application behaviour including network events is first studied and then checked for any deviations. Application data source acts as an input for IDS [25].

Hypervisor-based IDS: This is specially designed to monitor hypervisor along with its communication with VMs, communications between VMs, and even communication within the hypervisor. Information availability is one major advantage of this system [9].

Distributed IDS: For large networks, Distributed IDS having a network of connected IDSs is required. Each IDS along with its duty communicates with other IDSs in the network or the central server for analysing and book-keeping of the intrusions [1].

Wireless-Based IDS: They are similar to NIDS but deal wireless network traffic, i.e., they capture data from ad hoc networks, wireless sensor networks, etc. [6].

Network Behavioural Analysis: They inspect the traffic flow and infers intrusions from abnormal network behaviour or traffic flow [25].

This concludes all the major categories within this classification. Next, we classify as per techniques used within IDS.

2.1.2 Intrusion Detection Techniques

Due to diverse intrusion methods and categories, varied number of IDS are available with different techniques or algorithms powering them. Mainly they are classified under three main heads, namely, misuse based (detecting intrusion via mapping against available knowledge base), anomaly-based (deviation from expected behaviour), and hybrid-based (incorporating both misuse and anomaly based and using a method to get a final conclusion on the basis of results from both modules). All three types and their main sub-categories are briefly explained below [1] [25] [6] [9] [2] [23] [22] [20].

A. Misuse Based detection

It is also sometimes referred to as Signature Based detection by some authors. This is one of the most basic and yet, effective detection approach. The idea is to compare or map the activities or behaviour against an available knowledge base to detect any intrusions. Albeit experiments show it to be effective, it cannot detect unknown intrusions, i.e., it can only detect

intrusions whose definitions are already present in its knowledge base. There are few sub-categories depending upon the strategy they follow; some famous ones are:

State-Based detection technique

This technique is encircled around finite automata. The knowledge base consists of a huge number of definitions of various intrusions, and along with a finite automaton to match the activity against them. If there is a match it is simply an intrusion and necessary actions are taken depending upon active or passive nature (explained later, Section 1.3.3) of the IDS. If there is no match, as per misuse detection approach they classify it as normal behaviour.

String matching

This technique looks for a matching with the pre-saved signature strings of intrusions in the packets. The complexity and the procedure both are encircled around an efficient string matching algorithm. But the ever-increasing number of definitions call for higher computational power and time.

B. Anomaly-Based detection

It uses somewhat opposite approach than misuse based detection, where misuse based IDS looks for intrusions as per their knowledge base, Anomaly-based detection approach first learns the normal activity or behaviour of the machine, then any deviation from the expected behaviour is classified as suspicious with some level of suspicion, and depending upon the severity of the level, it is classified as an intrusion or not. It is used to detect the unknown intrusions. As the methodology is encircled around deviation from normal activity, it usually finds a hard time in highly varying or dynamic working environment, normal activities may be sometimes classified as intrusions and vice versa. And thus, it usually gives high false alarm rates. There are a large number of techniques or algorithms under this category, for the sake of understanding we have grouped the important ones under groups, and these are explained below.

i. Self-Learning

The aim is to detect the unknown intrusions, and the best system is the one that can learn new intrusions with minimal or no human intervention. Thus, it is one of the most important and active areas of research. There are many important and actively used techniques under this head. For easy understanding, we have provided a brief explanation of all the techniques in an

ordered fashion and grouped under two heads, namely, Time Series and Machine Learning and Data Mining.

Time Series

Time series based models are used to enhance the system, especially, in dynamic changing environments. In these models, the alerts generated by various activities classified as suspicious or intrusions are further analysed. The alerts are stored as a time series with specific alerts at specific time and duration record, and the series is further analysed to differentiate normal activities from intrusions. They are further sub-categories like discrete time series, but we won't be discussing them.

Machine Learning and Data Mining

When it is about making a machine intelligent or autonomous in any sense, then the answer by the computer science field is Machine Learning and various fields under and associated with it. For learning we need properly planned data, Data Mining provides us with that ability. There are various algorithms for machine learning. The most widely used are mentioned below, accompanied by short descriptions.

Artificial Neural Networks (ANN) based: First, in line comes the ANN, to generalise the data and differentiate between normal activity and intrusion. There are many proposed models based on various models of ANN, specifically, multilayer perceptron, SOM, multi-layer feedforward networks, etc. These models provide good accuracy.

Fuzzy Logic based: Fuzzy logics are always used to provide a conclusion with a level of surety. The IDS is useful in case of quantitative features, furthermore, it can handle some uncertainties. But has a high time complexity and usually provides a high false-positive rate.

Support Vector Machines (SVM) based: SVM based models provide good results with low false alarm rates and require less training data than ANN, and are most useful in case of small training data. Moreover, the technique can handle an enormous number of features, but they need to be discrete.

Genetic Algorithms (GA) based: These algorithms are used in tandem or collaboration with other models, because they are used to select optimal parameters or network features for other techniques, in order to attain better results.

Decision Tree-based: They are usually used to combine or infer the final result based on the results of two or more algorithms or modules.

Bayesian Network based: It uses Bayesian probability to classify the event as a normal activity or an intrusion, justified with probability values. Bayesian classifier model can also be clubbed.

Clustering based: The model forms clusters during its learning phase. And then the new events are mapped to these clusters to be classified as normal activity or intrusion.

K-Nearest Neighbour (KNN): Is similar to clustering, the only difference being the algorithm for forming clusters.

Deep Networks: Like ANN, these models attempt to form networks, more like multi-layer perceptron but differs in the architecture and method of training. This technique uses unsupervised learning and uses training data with many layers in hierarchical networks, to conduct classification.

ii. Programmed

These models are programmed by humans and are regularly updated with latest known intrusions or methodologies to improve the outcomes. There are a few techniques under this head.

Simple Rule-based IDS have a knowledge base of rules created as per normal activity, and events are matched against these rules to be classified as a normal behaviour or not.

Statistical Models or Threshold based IDS use statistical data collected over time from normal activity of the machine and sets some threshold for classification depending upon that data earlier collected and analysed, maybe with human intervention.

C. Hybrid IDS

The most basic idea of Hybrid IDS is to have the advantages of both misuse and anomaly-based detection and infer the final result from the results of both modules. Hybrid IDS saves the high complexity of anomaly detection algorithms for the already known intrusions, and more importantly, reduces the overall false alarm rate of the system when compared to IDS using just anomaly-based detection [5].

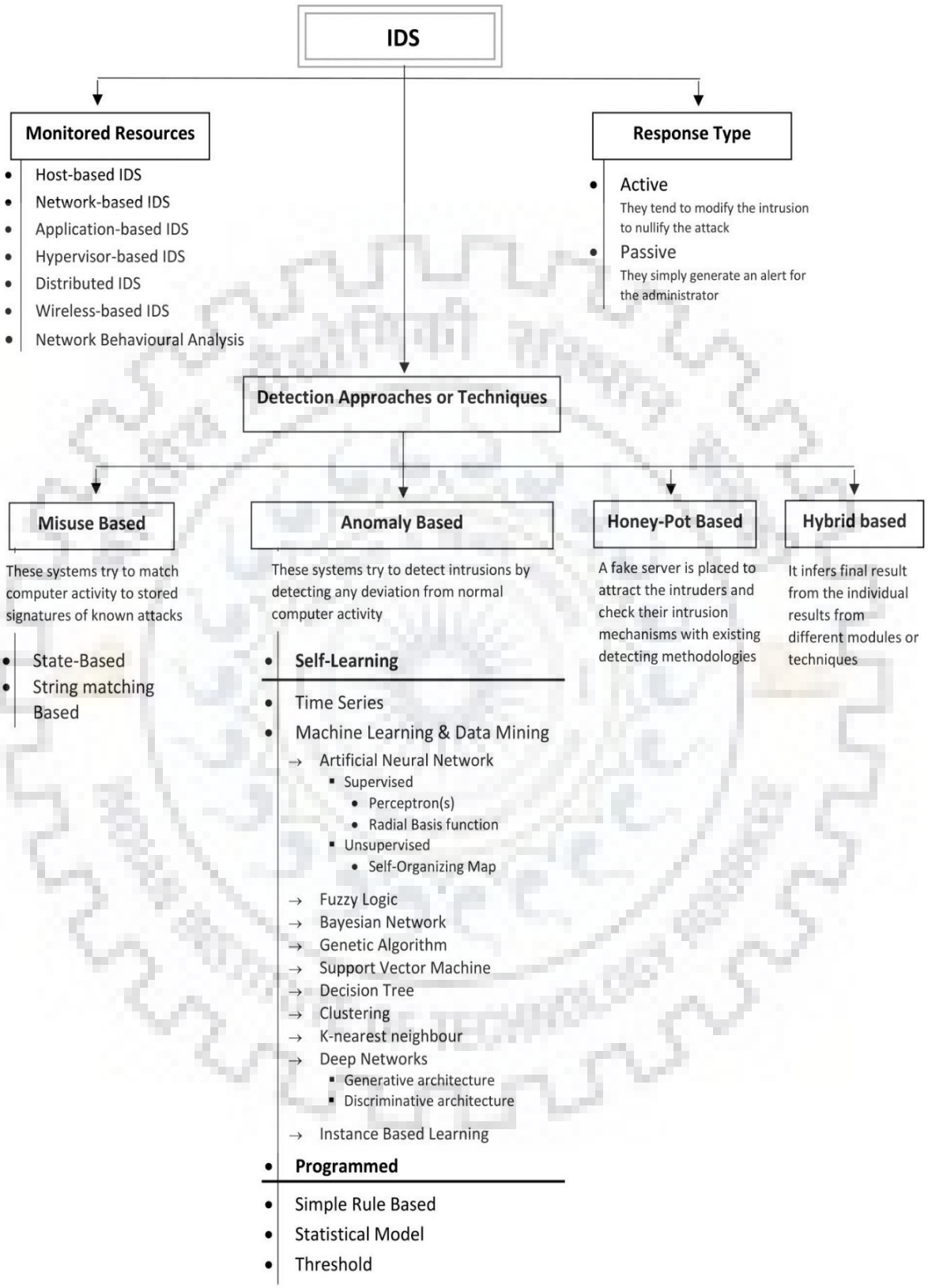
D. Honey-Pot based IDS

In this technique, a fake server is deployed and is safely placed behind a firewall to attract the intruders and use the data to train IDS and safeguard real servers [26].

2.1.3 Response behaviour

Depending upon the response of the IDS, after detecting the intrusion, they are categorised as Active and Passive IDS. If the IDS after detecting an intrusion takes no preventing measures (i.e., blocking or changing the intrusion) other than alerting the administrator or concerned authority on it, then the IDS is termed as Passive. An IDS is Active if other than just alerting, it takes measures to prevent intrusion like blocking or altering the definition of the intrusion, to at least if not protect, make it less severe.

Figure 2.1 IDS Classification



2.2 Literature Survey

2.2.1 Rule-based Approach

A sub-category of signature-based approach, and is mainly useful for known and specific attacks. With the increase in the type of attacks, the signature database exploded in size and matching against each signature or rule despite best-known string matching algorithms require a significant amount of time, though fewer resources. Ozgur et al. [2] proposed a hybrid of rule-based and anomaly detection unit, and the results were in accordance with the above-stated observations. Storing the signatures in the form of regular expressions improved on the database size, but required extra processing time and resources [3]. Famous and most used IDS with this approach are Snort and ClamAV [2] [3].

2.2.2 Software Defined Networking

Most appropriate definition as stated by Open Networking Foundation (ONF) is as follows: "In the SDN architecture, the control and data planes are decoupled, network intelligence and state are logically centralized, and the underlying network infrastructure is abstracted from the applications" [ONF] [7]. With recent developments in SDN, the SDN-based cloud system offers us new tactics to defeat intrusion, specifically, DoS attack. But on the contrary, the security of the SDN is still an open question, thus giving vulnerabilities, grievously, "Zero-day Vulnerabilities[1]" [7]. Furthermore, the addition of SDN can improve cloud scalability, manageability and controllability, the need for today, in addition to dynamism [14].

2.2.3 Anomaly detection and Machine Learning

Self-Organising Maps (SOM): This Machine Learning algorithm takes the advantage of unsupervised learning. Ozgur et al. [2] used SOM in their anomaly detection unit of the module and achieved an acceptable accuracy. The SOM frees the developer from few constraints, but at a cost of accuracy and training time. Once trained, the algorithm provides fast and acceptable results, with an acceptable convergence time of updates.

Naïve Bayes Classifier and Artificial Neural Network: Both of these methods suffer from the common drawback of low detection rate for low-frequent attacks such as R2L[2] (Remote to Local) and U2R[3] (User to Root) [9]. Amjad et al. [15] proposed a combination of Naïve Bayes and Random Forest that yielded better results in many cases but also worse in few cases [26] [23] [9] [20].

Fuzzy Logic: As stated in [9] [8] fuzzy logic based algorithms clearly outperform their original naïve classifiers and other approaches like Naïve Bayes and ANN. N. Pandeshwari et al. proposed a system using Fuzzy C-Means-ANN (FCM-ANN) and tested the same against Naïve Bayes and ANN, the former, as expected, performed better than the rest, but with a significant increase in resource utilisation, and required much intense training. Though, the convergence time of continuous learning was acceptable [23] [22]. Wei-Chao Lin proposed a model, termed as CANN [20], which clusters the data (unsupervised learning) and also keep the nearest neighbours of each cluster and the new data is assigned to a cluster depending upon its distance from both the cluster centre and nearest neighbours. The model showed remarkable accuracy for DoS and Probe attacks, but with a considerable hike in processing power. S. Iqbal [22] describes different types of fuzzy classifiers with their pros and cons for intrusion detection.

Incremental Mining Approach: Ming-Yang Su et al. [11] suggested a NIDS (Network Intrusion Detection System) capable of providing real-time results, tested to render a decision in every 2 secs by the authors, based on "incremental mining for fuzzy association rules". The core logic of the system is to derive features from packet headers only and the approach provides much better results than static mining approaches. As it is in the nature of the fuzzy association rules, only large-scale attacks can be targeted with this approach, but with good accuracy.

K-Nearest Neighbour (KNN) and Genetic Algorithm (GA): As with the fuzzy association rules, the KNN classifier is known to perform well for only large-scale attacks like DoS. The base of classification is the features on the basis of which classification is done, this can be improved by using GA as the methodology for selecting features for different clusters. GA greatly improves the performance, and the same can be observed by viewing one such system proposed by Ming-Yang Su in [10].

Hybrid Approaches: There are various hybrid systems proposed each with their pros and cons. In [4] the authors suggested an interesting approach by diving the system into components and keeping the storage safe and an exclusive component termed as "service component", by the authors, is used to establish communication with the storage device once the network flow is tested for intrusions by various algorithms running separately on separate machines to distribute the load from the main cloud controller to various nodes. Another approach is given by Ozgur et al. [2] using SOM and Rule-based units together to deduce a final conclusion.

Divide-and-Conquer: With the evolution of IDS from singular units of Anomaly or Signature-based to Hybrid approaches, the question of resource utilisation and service response time

became more and more prominent. Few types of research [3] [5] [6] believed in "*Divide-and-Conquer"* approach. The approaches include dividing the load from central cloud controller to all the physical machines by running IDS on each machine separately [5]. P. K. Shelke [6] proposed a NIDS, that uses multi-threading to boost the performance of the system. In the overall implementation, leaving and entering packets are collected by a collector module and are pushed onto a common shared queue, boosted by multi-threaded processes, and finally, the alert is generated by the concerned alerting module, once the threat is identified. Moreover, there is a third party deployed to keep a watch over complete procedure and to inform the service provider and the user, instantly.

2.3 Research Gaps

With tremendous advancements in machine learning and artificial intelligence, yet there is no algorithm or model capable of detecting all the known intrusions with good accuracy. Table 2.1 summarizes the prominent IDSs with their characteristics and main limitations.

The present era needs an IDS capable of handling or at least reducing the chances of all the known intrusions with considerable accuracy, perhaps with a slight increase in human supervision at the beginning. There is a need of transformation from specialised and exclusive IDS capable of handling only a few intrusion types to much intelligent IDS capable of handling much bigger class of intrusions. Less frequent attacks no longer means that their detection accuracies can go unnoticed, with data leaks and hacks [28] [29] increasing day by day, less frequent but severe attacks like U2R, R2L and probe are becoming a major security concern.

Though there are IDS capable of handling a wider class of attacks but at a serious cost to resources, like evolving fuzzy neural networks, nearest neighbours, etc., refer to table 2.1. Chapter 5 further compares and briefs about various models.

Conclusively, there is a serious need of a lighter IDS with low run and response times, simultaneously, addressing the security concern.

Table 2.1 Summary of Prominent IDS

IDS/IPS technique	Characteristics/Advantages	Limitations
Misuse-based • State-based • String matching	Matches against all known patterns High accuracy Low computational cost*	Cannot detect unknown or variants of known intrusions High false-positive rate for new attacks
Anomaly-based		
ANN-based	Efficiently classify unstructured network packets Multiple hidden layers improve efficiency	Requires large amount of data during training High time complexity Not fit for real-time detection Less flexible
SOM-based	A sub-category of ANN but requires less training data Faster detection speed for specific categories, mainly, binary data	Useful for less dimensional or dimensionally independent data Not easily adaptable to changing behaviour
Fuzzy Logic-based	Quantitative features Flexible Handles uncertainties better	High false-positive rate, worse than ANN High time complexity
Bayesian-based	Assigns relations between various features selected, using probability Can handle small data loss	Not useful if the features are independent, high false alarm rates in that case Based on probability, uncertain detection accuracy
GA based	Basically, provides best features to be used in detection Makes the system efficient	Can be used specifically depending upon features, not generally Complex method
SVM based	Can perform well in case limited data is available Can handle large number of features	Can only be applied to discrete features
Simple rule-based	Unlike all others explained under anomaly section, it is programmed-based instead of Self-learning-based Rules in varied forms are stored, and patterns are matched for intrusion detection	Can be used against unknown intrusions, but for specific sets Low computational needs when compared to other anomaly-based IDS

Chapter 3 PROPOSED APPROACH

The following model is proposed considering the observations in chapter 2. As suggested earlier the IDS should be resource friendly with good response time with no compromise to the security. The former is achieved via the concept of modularity, dividing the work into separate modules running independently. There are already a lot of different detection systems on the same concept, but how we tend to use it is different.

The model consists of two phases, namely, clustering phase (unsupervised learning) and a classification phase. Clustering, done on the scaled and processed data, provides n number of clusters created by the algorithm from the difference between different dimensional or feature values within the data. The purpose of unsupervised learning is to give weight to the already available feature values and to craft the natural differences between records. Directly supervised learning on the complete data may lead to overfitting of the model, providing high false alarm rates or lower accuracy. The main idea behind clustering is to find unique characteristics within all possible clusters, in order to accomplish better results with different classification models in phase II. Many of the researchers till today tries to accomplish better results from a single structure model, perhaps having parallel modules but all running same algorithms [9] [20] [2] [23] [26] [27] [6]. We believe results, especially response rate, can be improved by uniquely identifying unique distinguishing properties of each cluster, the results back up our belief.

The second phase, classification module, is basically a module with several sub-modules equipped with different algorithms like k-NN (k-Nearest Neighbours), SVM (Support Vector Machine) or ANN (Artificial Neural Network). After clustering, i.e., phase I, depending upon cluster properties specific submodule is selected for classification. The benefit it provides is rooted in pros and cons of above-used algorithms. Every machine learning algorithm suffers from some kind of drawback, for example, ANN shows great results for DoS and Probe attacks and equally or even worse for U2R and R2L attacks [9]. This model gets most votes in nullifying or considerably reducing these limitations. The above mentioned sub-module algorithms aren't final and are taken to demonstrate the key features of the model. These models can be replaced with the better ones in accordance with the clusters they are associated. Sophisticated modules needed for clusters with greater variance in intrusion signatures and simple classification modules can work for other clusters, summarizing the main concept and idea behind the model.

3.1 Phase I: Data Preprocessing

Before clustering can commence the raw data needs to be scaled. The main idea behind scaling is to modify data to have a mean of 0 and a standard deviation of 1. Scaling algorithm:

3.1.1 Standardization of Data

Algorithm:

For each feature:

For each value in feature:

$$x_{new} = \frac{x_{old} - \mu}{\sigma} \quad (1)$$

Where, mean

$$\mu = \frac{1}{N} \sum_{i=1}^{N} (x_i) \quad (2)$$

And standard deviation

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i - \mu)^2} \quad (3)$$

3.1.2 Dimensionality Reduction:

The data still contains 41 features or dimensions, and there is a need to reduce the number of dimensions for faster processing and improving on overfitting.

Principal Component Analysis (PCA), used for linear dimensionality reduction, uses "singular value decomposition" and orthogonal transformations to convert a set of observations to linearly uncorrelated variables from possibly correlated variables. The conversion is such that first component selected possesses highest possible variance within data values, and the process of choosing max variance component, orthogonal to already selected components, from remaining components continues till criteria of desired number of components or desired variance is met.

Algorithm (PCA):

For the best results PCA requires standardized data, i.e. mean 0 and standard deviation 1.

Input: Data Matrix, X with standardized data column wise or feature wise.

Transformation: each row vector, $x_i \in X$, to a new vector of principal component values $t_{(i)} = (t_1 , \ldots . , t_L)_{(i)}$, structured by a set of p-dimensional weight vectors $w_k = (w_1, \ldots , w_p)$; where,

$$t_{k(i)} = x_i \, . w_{(k)} \qquad \text{for } i = 1,\ldots..,n \text{ and } k = 1,\ldots..,L \tag{4}$$

First component is calculated as:

$$w_{(1)} = \arg\max \{ \frac{w^T X^T X w}{w^T w} \} \tag{5}$$

And further components, say k, by subtracting first ($k - 1$) selected components from X:

$$X_k = X - \sum_{s=1}^{k-1} X w_{(s)} w_{(s)}^T \tag{6}$$

And then using (5) on X_k instead of X to find k^{th} component.

Repeat till stopping criteria.

3.2 Phase II: Clustering

The preprocessed data is clustered using fuzzy c-means algorithm.

Algorithm:

The algorithm attempts to classify ‘n data points’ (n-dimensional) to ‘c clusters’.

Input: $M = \{x_1, x_2, \ldots, x_n\}$; finite data

Output: $K = \{c_1, c_2, \ldots., c_c\}$; c clusters and a partition matrix

Partition matrix: This matrix contains the membership value between 0 and 100 (in percentage) for each data point towards each cluster. The maximum value gives the point its final membership in a specific cluster.

Note: In case of a tie between membership values, the first cluster having max value is selected.

The objective function:

$$\arg\min_{C} \sum_{i=1}^{n} \sum_{j=1}^{c} w_{ij}^{m} \|\mathbf{x}_i - \mathbf{c}_j\|^2, \tag{7}$$

Where, $$w_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\|\mathbf{x}_i - \mathbf{c}_j\|}{\|\mathbf{x}_i - \mathbf{c}_k\|} \right)^{\frac{2}{m-1}}}. \tag{8}$$

And update the centers as $$c_j = \frac{\sum_{i=1}^{n} u_{ij} x_i}{\sum_{i=1}^{n} u_{ij}} \quad (9)$$

The algorithm aims to minimize the above-mentioned function.

The points, M, keeps on changing their clusters depending upon their distance values from new centroids (centres) of each cluster, till either the distance makes it a permanent residence of a cluster or the algorithm itself converges.

The algorithm converges when the objective function can no longer be minimized or the difference between two consecutive objective function values is less than threshold fixed by the programmer.

Once the clusters are formed specific classification models, if necessary, are used to provide final inference regarding network packet.

The model is supposed to be deployed at the cloud controller, the first entry point of network traffic after firewall.

3.3 Phase III: Classification

As stated above the module is further divided into sub-modules. The algorithms selected for various cases are as follows, clusters having (**rules**):

Table 3.1 Selected classification model for various attack groups

Attack group	Classification Module
Only DoS	ANN [23] [9]
Only Probe	k-NN (data reduced to 6 dimensions) [20]
Probe + DoS	CANN [20]
R2L and U2R	FCM-ANN [23]
>= 3 types of intrusions	k-NN (19 dimensions) [20]

3.2.1 Artificial Neural Network (ANN)

The data, from the cluster as per above rules, is processed by ANN. ANN is a network of simple processing units clubbed into a network to provide fuzzier results with the autonomous learning and improving capabilities [23]. There is an input layer, an output layer and hidden layers between the former and latter.

Algorithm:

Each node, with input node having value a_i, multiplied with a weight w_{ij}, between input layer and hidden layer. Each node in j processes its value as:

$$In(j) = \theta_j + \sum_{i=1}^{n}(a_i \,.\, w_{ij}) \tag{10}$$

Where "bipolar sigmoid function" process *In(j)* as:

$$\mu(x) = \left({}^{2}/_{(1 - e^{a})} \right) - 1 \tag{11}$$

The output is than broadcasted to output layer as:

$$y_k = \theta_k + \sum_{j=1}^{m} w_{jk} f(\ln(j)) \tag{12}$$

Where, θ_j and θ_k are hidden and output layer biases, respectively.

3.2.2 k Nearest Neighbours (k-NN)

The k nearest neighbour is another standard machine learning algorithm, and classified as a lazy runner, as all the values are approximately recorded with actual computations deferred until classification [20].

Algorithm:

For a test data point d, and present data set D with its labels.

Step 1: Calculate the distance of d from all points in D using 7, 8 or 9.

Step 2: Gather k nearest neighbours of d.

Step 3: d belongs to the class that has majority of points in d's nearest neighbour circle.

Where, k is the minimum number of points to be present in the nearest neighbour circle with a maximum permissible radius (distance) of c, specified by the programmer.

Distance function could be:

$$\text{Euclidean} : \left(\sum(\alpha_i - \beta_i)^2 \right)^{1/2} \quad (13)$$

$$\text{Manhatan} : \sum|\alpha_i - \beta_i| \quad (14)$$

$$\text{Minkowski} : \left(\sum(|\alpha_i - \beta_i|)^q \right)^{1/q} \quad (15)$$

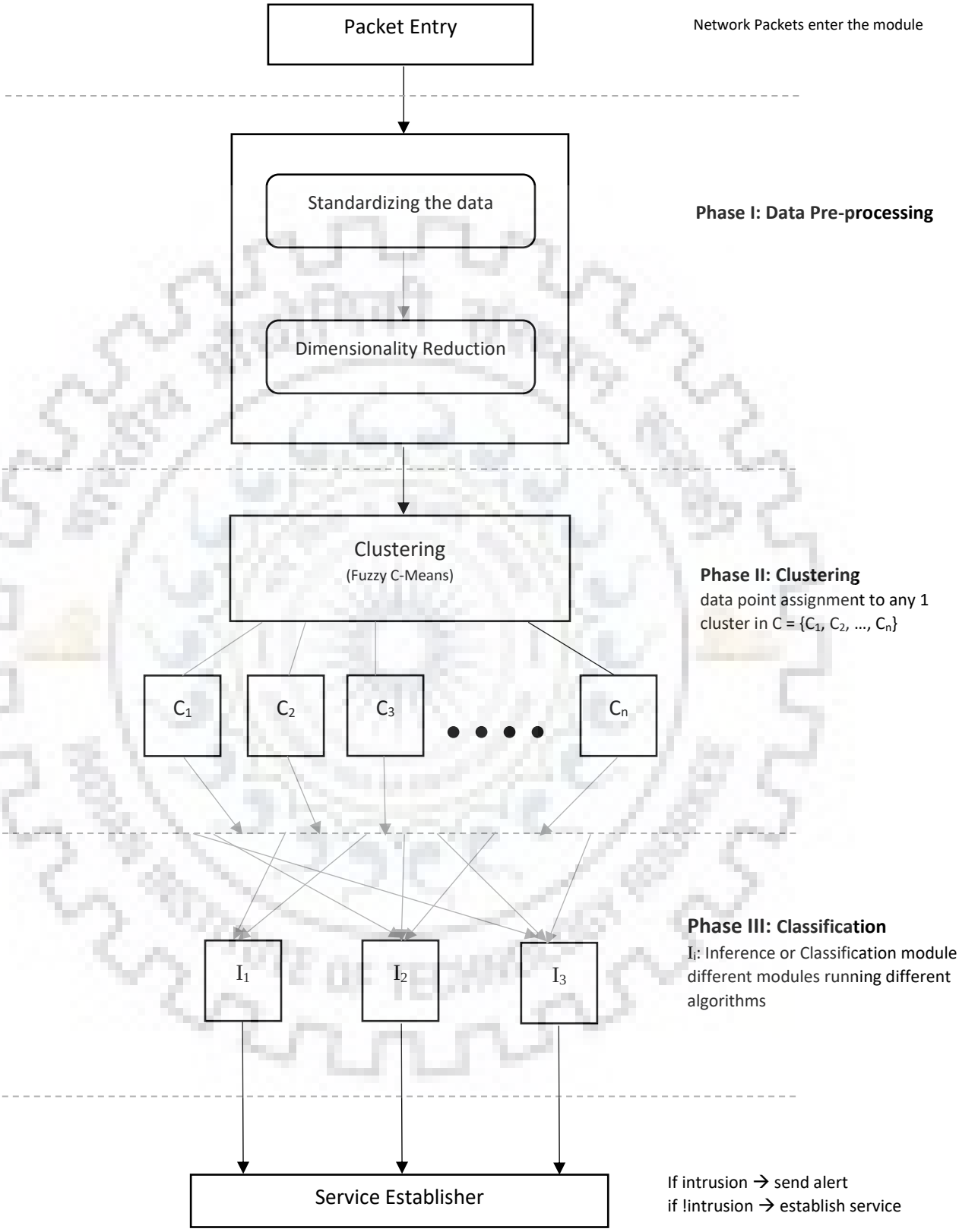
The main drawback of k-NN is the time complexity since it is a lazy learner, all the computations are done during the testing time which could significantly increase the response time.

3.2.3 Cluster And Nearest Neighbour (CANN)

W.C. Lin [20] proposed a method, he termed as CANN (combine clusters and nearest neighbour). CANN is a two-stage process, first clusters the data and stores k nearest neighbours of each cluster. The test data point is classified via aggregation of distance between point & cluster centre and an average of distances of a point from nearest neighbours. For stage 1 the author used k-means clustering and transformed the problem to five-class classification problem.

Fig 3.1 otlines the overall model, with packets entering at the packet entry block of the model, preprocessed in phase I. Phase II assigns the packet to one of the n clusters as per its feature values, which is then classified as intrusion or normal via classifiers in phase III, if not already classified in phase II itself. Finally, service to a workstation or cloud or any machine, to which the model is attached for security, is established if the packet or the request classifies as normal, otherwise, an alert to the administrator is generated.

Figure 3.1 Proposed Model



Chapter 4 EXPERIMENTS AND RESULTS

4.1 Experiment Environment

The experiments were conducted on a machine running Ubuntu 16.04 LTS, powered by Intel i5 Core-5200 CPU @ 2.20 x 4 and 8 GB physical memory. The Kddcup'99 dataset, as on 1st May, ’18, was used for experiments as shown in table 4.1.

Table 4.1 Dataset Composition

Labels	Normal	DoS	Probe	U2R	R2L	Total
Records	9,72,781	38,83,370	41,102	52	1,126	**48,98,431**

4.2 Data Preprocessing

The Data is divided in 4:1 ratio or 80% to 20% between training and testing datasets, respectively. Thus, total train records and test records are 39,18,739 and 9,79,692, respectively. Further, the string values in the dataset are mapped to specific numerical values as in table 4.2

Table 4.2.a Service [19] to numerical values mapping

Service		Service		Service		Service		Service	
private	1	echo	15	hostnames	29	courier	43	ecr_i	57
smtp	2	discard	16	iso_tsap	30	exec	44	eco_i	58
http	3	systat	17	pop_2	31	shell	45	tim_i	59
ftp_data	4	daytime	18	netbios_dgm	32	efs	46	urp_i	60
IRC	5	netstat	19	netbios_ns	33	login	47	red_i	61
telnet	6	ssh	20	sql_net	34	printer	48	remote_job	62
domain	7	name	21	bgp	35	netbios_ssn	49	X11	63
finger	8	whois	22	vmnet	36	csnet_ns	50	http_8001	64
other	9	time	23	Z39_50	37	nntp	51	urh_i	65

ftp	10	mtp	24	ldap	38	supdup	52	aol	66
imap4	11	gopher	25	nnsp	39	http_443	53	auth	67
pop_3	12	rje	26	kshell	40	uucp_path	54	harvest	68
sunrpc	13	link	27	klogin	41	domain_u	55	http_2784	69
pm_dump	14	ctf	28	uucp	42	ntp_u	56	tftp_u	70

Here, protocols above transport layer are considered as services [9].

Table 4.2.b Protocols and Flags to numerical values mapping [19]

Protocols		**Flags**		**Flags**		**Flags**		**Flags**	
tcp	1	SF	1	S1	4	RSTR	7	RSTOSO	10
udp	2	SH	2	S2	5	REJ	8	OTH	11
icmp	3	S0	3	S3	6	RSTO	9		

Before we begin with the clustering we need to select a number of features and features themselves to be used for clustering. Generally, there are two ways of using PCA, one that delivers the data with the desired number of features, and another concerns itself with the final variance left in the data. We will stick with the latter.

For scaling standard scaling algorithm and rules, i.e., unit variance and mean as 0, is used as per machine learning algorithms guidelines [31] [9].

The kddcup'99 dataset provides 41 features for classification, PCA is used to scale down the number of features at a cost of 3% variance, i.e., after PCA the final data consist of 97% variance of the original data with 22 features. 97% is selected via experiment results on 99%, 99.99%, 97%, and 95% variance data. The former delivered the best results, refer to fig 4.1.

4.3 Evaluation Criteria

The evaluation criteria of the model are simple and standard, comprising Precision values and detection accuracy for all the attack types and normal record. The model's other focus is on improving response time, which makes the latter 2nd evaluation criteria. Table 4.4 provides the

confusion matrix that explains True-Negative (TN), False-Negative (FN), False-Positive (FP) and True-Positive (TP), as shown in table 4.3.

Table 4.3 Confusion Matrix

Actual \ Predicted	Normal	Intrusion
Normal	TN	FP
Intrusion	FN	TP

Precision values are calculated as:

$$Precision = \frac{TP}{TP+FP} \quad (16)$$

Detection accuracy:

$$Accuracy = \frac{Number\ of\ records\ correctly\ classified}{Total\ number\ of\ records} \quad (17)$$

Overall accuracy:

$$Overall\ Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (18)$$

Fuzzy Partitioning Coefficient (FPC) is a measure of the amount of overlap of data between clusters [29]. It ranges from 0 being all overlapped to 1 being hard clusters, i.e., no overlap. And it is computed as follows:

$$FPC = \frac{1}{n} \sum_{i=1}^{c} \sum_{j=1}^{n} (\mu_{ij})^2 \quad (19)$$

Where n: is the number of data records (vectors)

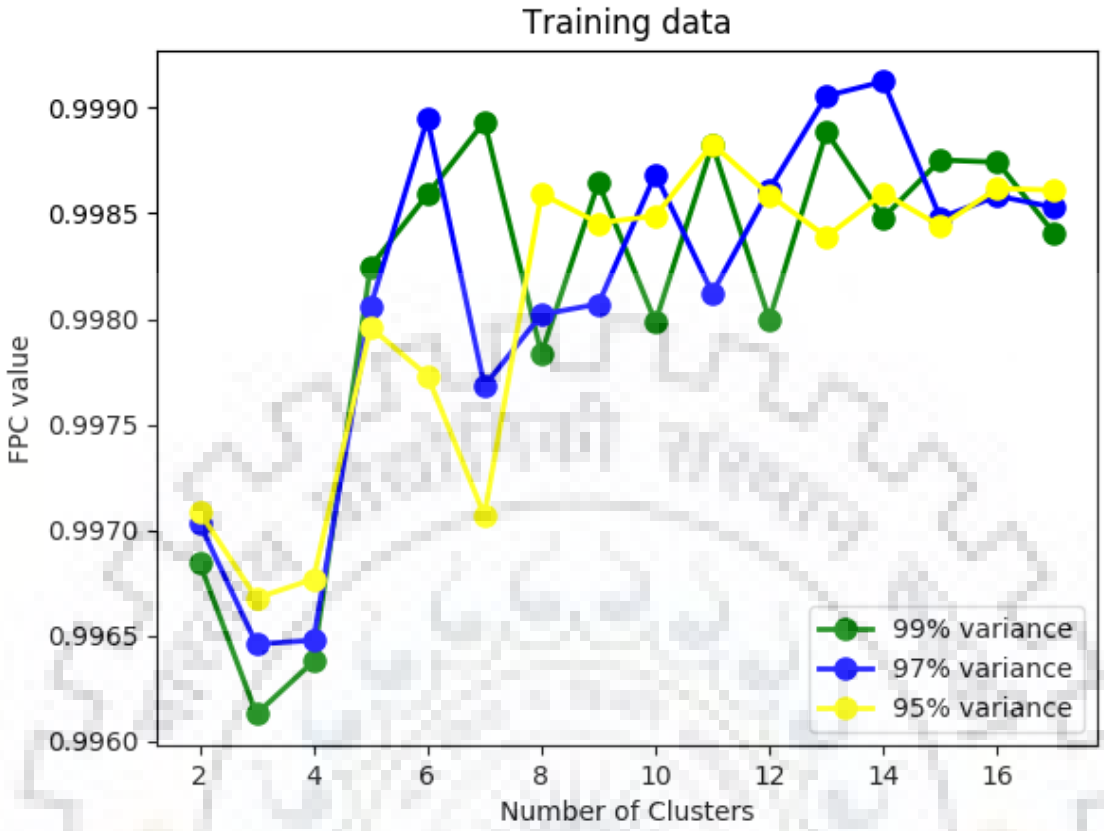
c : number of clusters

μ_{ij}: membership of vector j to cluster i.

4.4 Experiments and Results

As per the proposed approach the dataset with 41 features is transformed to dataset with fewer features, for experiments into 3 datasets: 20 features and 95% variance, 22 features and 97% variance, and 25 features and 99% variance. FPC values for all 3 datasets against number of clusters, ranging from 2 to 17 are given in fig 4.1. The figure 4.1 also clarifies the choice of 97% variance data, though experiments were conducted on 99% as well to compare the results.

Figure 4.1 Training Data vs FPC



With 97% data we have 22 features scaled and transformed by Standard Scalar and PCA, similarly, 99% data possesses 25 features. With the aid of FPC, we reduced our experiments to cluster values 14 and 15 on 97% data and 14 on 99% data.

Table 4.4.a Comparison of Clusters with 14 clusters of 97% and 99%, and 15 clusters on 97%

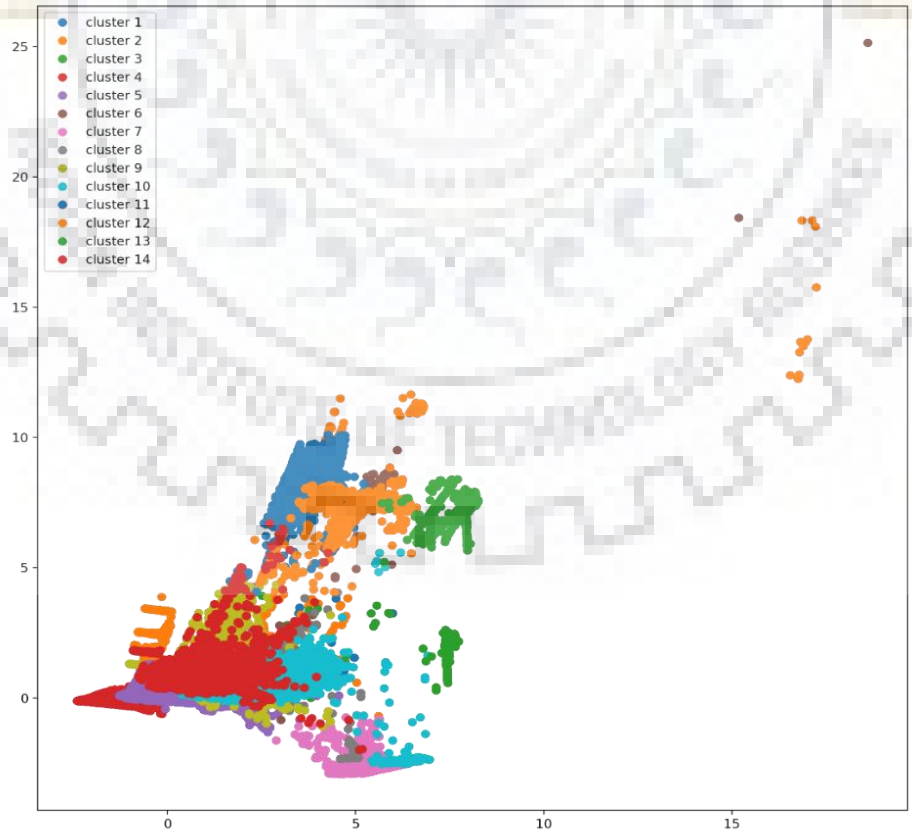
Clusters	Dataset	Data	Cluster						
			1	2	3	4	5	6	7
14	97% variance	Normal	8058	4105	8326	12	66777	57575	15682
		Intrusion	2	325	173951	368	424	325	248
14	99% variance	Normal	10058	4105	8570	15	66523	57575	15619
		Intrusion	6	315	173867	312	413	325	251
15	97% variance	Normal	8588	737	62986	9891	3346	915	13331
		Intrusion	422	64	331	16	3317	3	37

Table 4.4.b Comparison of Clusters with 14 clusters of 97% and 99%, and 15 clusters on 97%

Cluster	Dataset	Data	Cluster							
			8	9	10	11	12	13	14	15
14	97% variance	Normal	449	29171	2	1651	3	0	353	-
		Intrusion	42217	551	561325	66	3169	204	2349	-
14	99% variance	Normal	451	66	1659	5	29109	0	371	-
		Intrusion	41217	561667	64	3209	493	204	2197	-
15	97% variance	Normal	60781	0	619	13	3202	323	12	29393
		Intrusion	356	204	42233	561471	218	2350	174076	427

Table 4.4 shows the comparison on one such experiment, it can be inferred 14 clusters on both 97% and 99% varied data performs very close, but 97% provides lesser dimensions thus speeding the process. An average of several experiments on the random division of data was taken to conclude 14 clusters on 97% varied data as the final FCM clustering data, figure 4.2.

Figure 4.2 Training data clustering; 14 clusters and 97% variance data



14 clusters decision holds maximum votes in “Training data and testing data versus FPC”, figure 4.3. 7 clusters is also a good choice from figures 4.1 and 4.3, but experiments ruled out this choice due to the poor division of intrusion and normal records within the cluster as per our criteria.

Figure 4.3 Train Test data (FPC vs Number of clusters)

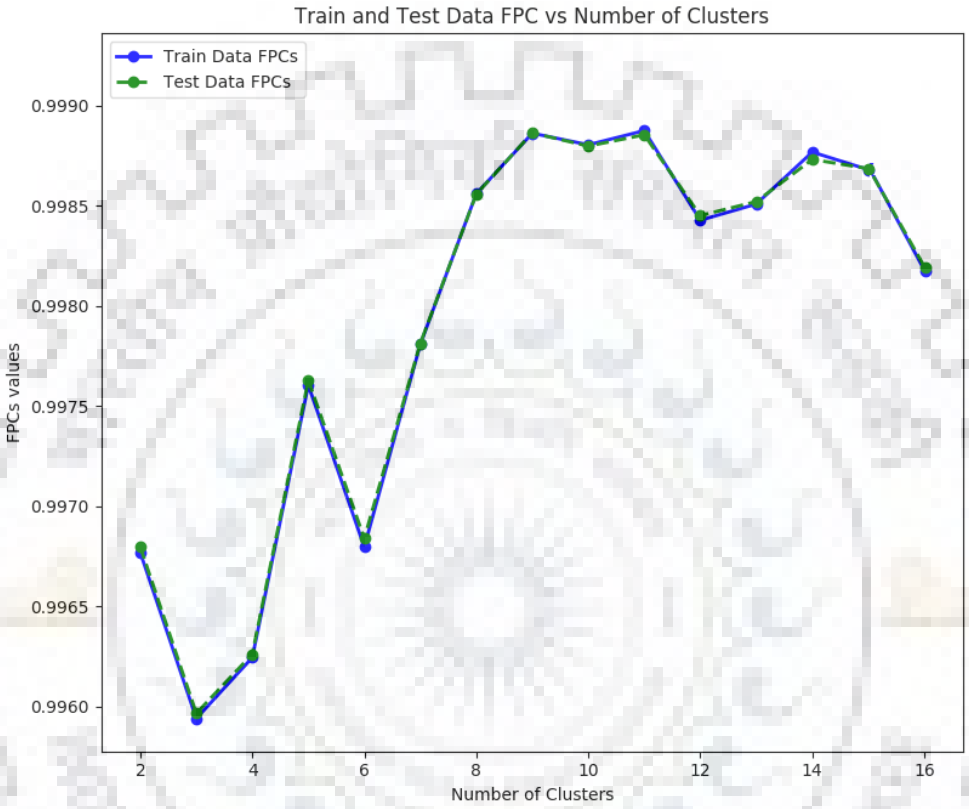


Table 4.5 shows the cluster composition from one experiment, and the data is further analysed to select classification modules in phase II.

Table 4.5.a Cluster composition

\ Cluster Data	1	2	3	4	5	6	7
Normal	8058	4105	8326	12	69777	57575	15682
DoS	2	227	173877	98	0	314	46
Probe	0	86	73	145	424	0	202
U2R	0	1	0	7	0	2	0
R2L	0	11	1	118	0	9	0

Table 4.5.b Cluster composition

\ Cluster Data	8	9	10	11	12	13	14
Normal	449	29171	59	1651	3	0	353
DoS	40691	244	561325	0	0	204	20
Probe	1523	303	0	0	3164	0	2320
U2R	1	4	0	0	0	0	2
R2L	2	0	0	66	0	0	7

The analysis of the above data (table 4.5) reveals some interesting facts, better than expected earlier. The most attracting clusters are 10, 12 and 13. With cluster number 10, directly classifying 14.32% of intrusions with a false alarm rate of just 0.01%. Next on the line are cluster numbers 12 and 13, again classifying 0.08% and 0.0052% of intrusions, respectively. 0.08% might not seem appreciable but the key property is the cluster contains only probe attacks, thus eliminating 38.32% of all the probe attacks, a significant portion of a severe attack. The major problem of all intrusion types detection is considerably reduced to 47%, clusters 2, 4, 6, 8 and 14, with respect to normal records, a stupendous achievement, a phenomenal reduction. Rest of the 53% of normal records are divided among clusters with almost 2 types of intrusions comparatively easy to detect. One more and an important observation is detention of a major number of R2L by clusters 4 and 11. R2L attacks are less frequent but very severe and are often very hard to detect if we consider cluster 4 to be an all intrusion cluster, we decrease our normal records detection accuracy by 0.0062%, a figure that can be considered as it eliminates 55.14% of this deadly attack. Finally, other clusters can be provided by algorithms as per rules mentioned in chapter 3, section 2. The reduction along with cluster 1 plays a vital role in reducing the response time, the clustering results directly aimed the main objective of reducing the response time significantly.

Following tables 4.6, 4.8 and 4.7 gives the confusion matrix, accuracy comparison and comparison of precision values (TP) of various classification modules as per desired criteria.

Table 4.6 Confusion matrix of model

\ Predicted Actual	Normal	DoS	Probe	U2R	R2L
Normal	192473	1645	23	7	14
DoS	777	776271	-	-	-
Probe	8	1	8253	-	-
U2R	3	-	-	14	-
R2L	20	-	-	-	194

Table 4.7 Comparison: Precision (TP) values in % of various models [20] [23]

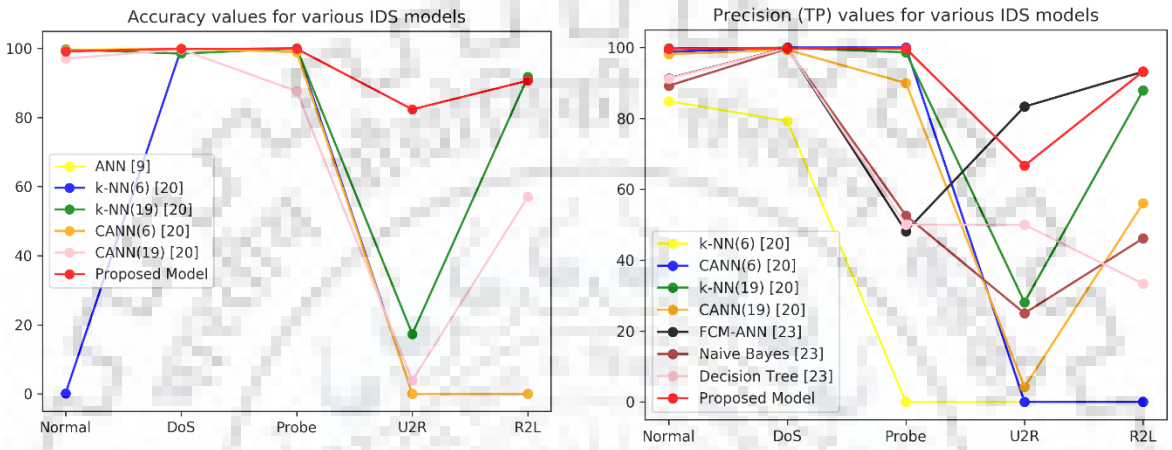
\ Record type Model	Normal	DoS	Probe	U2R	R2L
k-NN(6) [20]	84.88	79.25	0	0	0
CANN(6) [20]	98.81	99.99	100	0	0
k-NN(19) [20]	99.78	99.96	98.65	28.12	87.91
CANN(19) [20]	98.14	99.37	89.99	4.16	56.06
FCM-ANN [23]	91.32	99.91	48.12	83.33	93.18
Naïve Bayes [23]	89.22	99.69	52.61	25.00	46.15
Decision Tree [23]	91.22	99.84	50.00	50.00	33.33
Proposed Model	99.58	99.78	99.72	66.66	93.26

*Table 4.8 Comparison: IDS models and their accuracies in % [9] [20]; (*): study on * dimensional dataset*

Model\Attack	Normal	DoS	Probe	U2R	R2L
ANN [9]	99.59	99.99	98.98	0.0	0.0
k-NN(6) [20]	0.075	99.79	99.99	0.0	0.0
k-NN(19) [20]	99.68	98.49	99.98	17.31	91.74
CANN(6) [20]	99.44	99.91	98.93	0.0	0.0
CANN(19) [20]	97.04	99.68	87.61	3.85	57.02
Proposed Model	99.13	99.90	99.89	82.35	90.65

Final Accuracy and Precision of the model, provided in table 4.8 and 4.7, respectively, are calculated using the accuracies of various classification algorithms on the processed and confined small data of clusters, the overall accuracies are then calculated as per the data division in various clusters. Fig 4.4 graphically shows the comparison, as shown in tables 4.8 and 4.7, respectively.

Figure 4.4 Accuracy and Precision (TP) values of various IDS Models



Overall accuracy of the proposed model is 99.82% which is better than CANN [23] model with an accuracy of 99.76%. Classification modules CANN and FCM-ANN are comparable in terms of there execution time, but k-NN being a lazy learner, execution time shoots of the charts. The model is designed to utilise good detection abilities of k-NN, simultaneously diluting its high execution time. By analysing the data in table 4.5, we can derive that only 3.9% and 7.1% of total data is pushed to k-NN(19) module and k-NN(6) module, respectively. Moreover, 58.4% of total test data was directly classified from clustering only, which took only 19.247 seconds on 9,79,686 records, eliminating classification time totally, slightly increasing the false alarm rate, compensating the loss from k-NN. Experiments show an average of 51-59% of data fall in this category. Thus reducing the execution time considerably.

Chapter 5 CONCLUSION AND FUTURE WORK

5.1 Conclusion

The model clearly stands on our beliefs mentioned earlier and provides better results via dividing data into clusters. The model tries to advantage of various classification algorithms, suppressing their cons and limitations. Experiments show the overall accuracy of 99.82% for the model which is better than CANN [23] with 99.76%, which in turn is better or similar to the best classifiers in 19-dimesional dataset [23]. The main advantage of the model is the ability to use the advantages of various great classifiers while suppressing their disadvantages like run time or small data or feature space.

The overall model is designed keeping in mind the pace of technology, with new algorithms being invented every day and previous ones getting better and optimized, and all that the system administrator has to worry is replacing or adding the module.

5.2 Future Work

- Data Mining amendment: The results might improve by aiding the model with a data mining module capable of defining a security level depending upon a user's profile (his/her network usage, sites they access, etc.).
- Further, due to time constraints some of the classification models weren't programmed, instead there statistics were taken from other research papers, and were statistically placed. There might be better classification algorithms for the various Inference engines in phase II, time constrained this exploration.
- Outlier Removal: Importantly, during clustering we overlooked outliers, the model may perform better if the outlier removal is done prior to clustering.
- The overall performance may increase by dividing the network traffic as per protocols, i.e., ICMP, TCP and UDP. ICMP is free from U2R and R2L attacks.

REFERENCES

[1] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, M. Rajarajan, “A survey of intrusion detection techniques in Cloud”, Journal of Network and Computer Applications; 2013; Volume 36; pp. 43-45, 50.

[2] Ozgur Depren, Murat Topallar, Emin Anarim, M. Kemal Ciliz, “An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks”, Expert Systems with Applications, 2005, Volume 29; Issue 4; pp. 721.

[3] J. H. Lee, M. W. Park, J. H. Eom, T. M. Chung, “Multi-level Intrusion Detection System and Log Management in Cloud Computing”, ICACT, 2011, pp. 552-555.

[4] K. Vieira, A. Schulter, Carlos B. Westphall, and C. M. Westphall, “Intrusion Detection for Grid and Cloud Computing”, IEEE Computer Society, (July/August 2010).

[5] C. Mazzariello, R. Bifulco and R. Canonico, “Integrating a Network IDS into an Open Source Cloud Computing Environment”, Sixth International Conference on Information Assurance and Security, 2010.

[6] Ms P. K. Shelke, Ms S. Sontakke and Dr A. D. Gawande, “Intrusion Detection System for Cloud Computing”, International Journal of Scientific & Technology Research, May 2012, Volume 1; Issue 4; pp. 67-71.

[7] Qiao Yan, F. Richard Yu, Qingxiang Gong and Jianqiang Li, “Software-Defined Networking (SDN) and Distributed Denial of Service (DDoS) Attacks in Cloud Computing Environments: A Survey, Some Research Issues, and Challenges”, IEEE Communications Survey & Tutorials, 2016, Volume 18.

[8] Saeed M. Alqahtani and Robert John, “A Comparative Study of Different Fuzzy Classifier for Cloud Intrusion Detection Systems’ Alerts”, IEEE SSCI, 2016.

[9] N. Pandeeshwari and Ganesh Kumar, "Anomaly Detection System in Cloud Environment Using Fuzzy Clustering Based ANN", Journal: Mobile Networks and Applications, Springer, 2016, Volume 21, Issue 3, pp 494-505.

[10] Ming-Yang Su, "Real-time anomaly detection systems for Denial-of-Service attacks by weighted k-nearest-neighbor classifiers", Expert Systems with Applications, 2011.

[11] Ming-Yang Su, Gwo-Jong Yu and Chun-Yuen Lin, "A real-time network intrusion detection system for large-scale attacks based on an incremental mining approach", Computer and Security 28, Elsevier, 2009.

[12] Dimitris Gavrilis and Evangelos Dermatas, "Real-time detection of distributed denial-of-service attacks using RBF networks and statistical features", Computer Networks 48, 2005.

[13] Yashir Mehmood, Muhammad Awais Shibli, Umme Habiba and Rahat Masood, "Intrusion Detection System in Cloud Computing: Challenges and Opportunities", 2nd National Conference on Information Assurance, 2013.

[14] Y. D. Lin, D. Pitt, D. Hausheer, E. Johnson and Y. B. Lin, "Software-defined networking: Standardization for cloud computing's second wave", Computer, 2014, Volume 47, no. 11, pp. 19-21.

[15] Amjad HB, Sabyasachi P, Debasish J, "Machine Learning approach for intrusion detection on cloud virtual machines", International Journal of Application or Innovation in Engineering & Management 2, 2013, pp. 57-66.

[16] Vitaly Shmatikov and Ming-Hsiu Wang, "Security against probe-response attacks in collaborative intrusion detection", Proceedings of workshop on Large Scale Attack Defence, 2007, pp. 129-136.

[17] NIST, Website < http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-S2009-3733 >; National Vulnerability Database, 2011.

[18] Splay Tree, Website < https://en.wikipedia.org/wiki/Splay_tree >

[19] KddCup dataset, Website <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>

[20] Wei-Chao Lin, Shih-Wen Ke and Chih-Fong Tsai, "CANN: An intrusion detection system based on combining cluster centers and nearest neighbors", Knowledge-Based Systems, Elsevier, 2015.

[21] Nathan Keegan, Soo-Yeon Ji, Aastha Chaudhary, Claude Concolato, Byunggu Yu and Dong Hyun Jeong, "A survey of Cloud-Based network intrusion detection analysis", Human-centric Computing and Information Sciences, 2016.

[22] Salman Iqbal, Miss Laiha Mat Kiah, Babak Dhaghighi, Muzammil Hussain, Suleman Khan, Muhammad Khurram Khan and Kim-Kwang Raymond Choo, "On cloud security

attacks: A taxonomy and intrusion detection and prevention as a service", Journal of Network and Computer Applications, Elsevier, 2016.

[23] Gang Wang, Jinxing Hao, Jian Mab and Lihua Huang, "A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering", Expert Systems with Applications, Elsevier, 2010.

[24] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin and Kuang-Yuan Tung, "Intrusion Detection System: A comprehensive review", Journal of Network and Computer Applications, Elsevier, 2013.

[25] Ahmed Patel, Mona Tahavi, K. Bhaktiyari and J. Celestino Jr, "An Intrusion Detection And Prevention System In Cloud Computing: A Systematic Review", Journal of Network and Computer Applications, Elsevier, 2013.

[26] H. B. Baraka and H. Tianfield, "Intrusion detection system for cloud enivornment", Proceedings of the 7th International Conference on Security of Information and Networks, ACM, 2014.

[27] Facebook data leak, Website < http://www.euronews.com/2018/04/09/the-facebook-data-leak-what-happened-and-what-s-next>

[28] Twitter data leak, Website < https://techcrunch.com/2016/06/08/twitter-hack/>

[29] Xuanli Lisa Xie and Gerardo Beni, "A validity measure for Fuzzy Clustering", Transactions on Pattern Analysis and Machine Intelligence, Vol. 13, No. 8, IEEE, 1991.