A STATE Of THE ART ANOMALY IDS FOR HYPERVISOR

By

Muhammad Intisar Ullah Khan Sumbal

A Thesis Submitted to the Faculty of Department of Information Security Military College Of Signals, National University of Sciences and Technology, Rawalpindi in partial fulfillment of the requirments for the degree of
Master of Science in Infomation Security

August 2017

Supervisor's Certificate

It is certified that the final copy of thesis has been evaluated by me, found as per specified format and error free.

Dated: _____ Aug 2017 _____

(Maj Muhammad Faisal Amjad, PhD)

Declaration

I hereby declare that no portion of work presented in this thesis has been submitted in support of another award or qualification either at this institution or elsewhere.

_____

(Muhammad Intisar Ullah Khan Sumbal)

Acknowledgment

I want to express my gratitude toward Allah All-powerful for His innumerable blessings. I am extremely grateful to my family, my class fellows, my friends, and my faculty members for giving their huge support to enable me to do this Masters Thesis. Without their steady support, help and supplications, I would not have come to finish point in serene perspective.

I extend my most profound appreciation to my thesis supervisor, Major Dr. Muhammad Faisal Amjad, who gave me a stage and gave me the freedom to work in the territory of my advantage and extended his gigantic support preceding and additionally throughout this reasearch. His specialized direction, consolation, thoughts and point of view were imperative for consummation of this monotonous work. His support gave me certainty and helped me to comprehend about the topics profoundly and roused me towards the objectives.

I also want to express my sincerest gratefulness to Assistant Professor Dr. Hammad Afzal and Lecturer Waleed Bin Shahid for being an important part of my Research Supervisory Committee. Their academic direction, help and learning have been significant for effective fruition of my reasearch.

At last, I am appreciative and grateful to Military College of Signals and National University of Sciences and Technology for giving me an opportunity to help accomplish perfection by being related with the prestigious organization.

Muhammad Intisar Ullah Khan Sumbal

(Aug 2017)

Summary and Conclusions

Virtualization provides high availability, adaptability towards evolving business demands and cost effectiveness. In a virtualized environment, hypervisors are especially vulnerable to security threats because they manage all the shared resources. Detection of intrusions in hypervisors is therefore one of the major security challenges and is addressed in this research. To that end, I have proposed a hybrid machine learning technique which detects intrusions through anomalies in hypervisor's network traffic with a combination of fuzzy logic and genetic algorithm. To test the effectiveness of my intrusion detection technique, I deployed a testbed comprising a cloud enviornment with ESXi 6.0 having 30 virtual machines (VMs) and developed datasets of network traffic. I also tested the intrusion detection technique on DARPA 1998 dataset. Experimentation results show 84% detection accuracy under 89% positive predictive value and 77% negative predictive value. Furthermore, because training of my proposed learning algorithm is performed offline, it incurs no additional computational overhead.

Contents

List of Figures

List of Tables

List of Algorithms

Abstract

Virtual machine is a SW based replica of a real machine, built on the underline hardware machine. Virtualization enables today X86 computers to run multiple operating systems and applications, making your infrastructure simpler and more efficient. Applications get deployed faster, performance and availability increase and operations become automated, resulting in IT that's easier to implement and less costly to own and manage. I will study and review the major security threats and vulnerabilities aimed at virtualization environment. In order to analyse the security mechanism available for virtualization technology, dev within last decade, analyse their effectiveness against threats and vulnerabilities targeted specifically to virtualization environment. Setup and create a baseline for a secure virtualized environment. The aim of this research is to propose an anomaly detection system for fully virtualized environment. All the network traffic and performance metrics of the guest machines maintained by hypervior are used for modeling the behaviors. Analyzing the metrics and classifying them with the normal and abnormal activities. Utilizing SVM (support vector machines) in Intrusion Detection, the generalizing ability of IDS becomes good when the sample size is small (less priori knowledge). In this research I will be using the data sets already available for research. The model / framework of an Intrusion Detection System for Hypervisor using any one of the available data analysis techniques, best suiting my research work.

Chapter 1

Introduction

In the field of computing, virtualization is creating a simulated adaptation of a machine or resource like a server, network or an operating system (OS) where the framework splits the subject into one or more execution environments [1]. Virtual machines (VMs) host individual OS, applications and services. This virtualized environment is created and managed by a hardware, software or a firmware called Hypervisor or Virtual Machine Monitor (VMM) [2]. Various VMs can be introduced on a solitary hardware as these are not dependant on the state of physical resource [3]. However, the segregation of VMs on a single machine has many security vulnerabilities and threats like VM Sprawl, VM Hyperjacking, VM Escape, Incorrect VM Isolation, VM Theft, VM Mobility, VM to VM Attacks, VM Hopping, Hypervisor Intrusion, Network Security, Denial of Service attacks, Flooding Attacks, Network Channel Eavesdropping [4], [5] etc. Intrusion detection mechanism is an useful methodology to mitigate threats associated with a cluster of VMs in hypervisors [6].

Intrusion Detection System (IDS) is an effective tool to safeguard computer

network system from invasions or attacks. The rapid developments in the expanding field of virtualization has provided an opening to prowlers, attackers and hackers to find different illegal ways to exploit a machine or network [7]. Hence, with the evolution of new skills the threat vector is also rising and the major problem encountered today is the fortification of great volume of network traffic data gathered in network communication called 'Big Data' [8]. The Network Intrusion Detection System (NIDS) examine the attacks by spotting the network traffic packets, impacting several hosts which are associated to the network. NIDS have proved to be effective against network based attacks and a favoured choice where networks have conquered many other technologies. Hence, a security analyser must place this sensor at an appropriate area as it will affect its adequacy [7]. Anomaly based IDS (ABIDS) is an important variation in NIDS which identifies both new and novel attacks. ABIDS system utilizes an alternate philosophy which veers off from the signature based approach and identify the zero day, insider or new attack approaches. Nevertheless, as a side effect, it creates more noteworthy false alerts which really are not assaults. A carefully trained system is therefore, essential to get the desired outcomes [9].

Machine Learning is valuable in ABIDS with the end goal of recognizing and classifying attacks resulting in higher true positive/negative rates. The parallel preparing methodology is likewise valuable in managing 'Big Data'. Machine Learning centres around building up the computer programs and prepare them to develop and change when they are presented to new information which can either be supervised: applying changes to information what has been realized in the past or unsupervised: where deductions can be drawn from the datasets [10].

In this research, I have used integration of machine learning approaches with ABIDS for prevention of malicious network traffic using a dataset prepared for training and testing of the system.

1.1 Problem Statement

This work will briefly discuss the history of research in intrusion detection techniques and introduce the two basic detection approaches: signature detection and anomaly detection. Finally, it presents the application of techniques developed for monitoring critical process systems, such as Data centers, to anomaly intrusion detection. The purpose of the suggested methodology is to detect network intrusion based on anomaly detection. The experimental results will be analyzed and cleansed for all false negatives and false positives over a stretched period of time and the results will be put through data mining techniques for comparison. Finally, a preventive model/ framework will be suggested for implementation.

1.2 Relevance to Industrial and Military needs

1.2.1 Industrial Needs

The industry is rapidly moving towards virtualized environment. Having a self-sustained anomaly detection system is the dream of every organization. Through our system call tracing system, we will get a large data set of system call sequences. Support Vector Machine (SVM) has two input streams one is the trail stream used for training purpose and one is original data stream from produc-

tion system, which gives us anomaly detection capability

1.2.2 Military Needs

Over last decade Pakistan Army has transformed into a full IT based system. Having such a system, which is mostly working on virtualized environment, we need a self-sustainable system for its intrusion detection. These attacks on this closed network can be reduced by incorporating this state of the art anomaly detection system, using system calls training for IDS.

1.3 Research Objectives

The main objectives of my thesis are:-

- **Task 1**: Study different techniques for Intrusion detection.
- **Task 2**: Creating a cloud environment using virtualized environment for getting data sets.
- **Task 3**: Improvement of analysis tools to get better results.
- **Task 4**: Propose a framework / model to provide intrusion detection for Host based system using virtualized environment.

1.4 Thesis Organization

This study is organized in eight chapters, details as under:-

- **Chapter 2**: This chapter introduces the reader to some Important Concepts in Virtualization
- **Chapter 3**: This chapter relates the ongoing work in the field of Intrusion Dection with my work.
- **Chapter 4**: This chapter gives the complete methodology adopted for installation of Hypervisor and cloud envoronment.
- **Chapter 5**: This chapter gives a complete process of data aquisition, from our system.
- **Chapter 6**: Proposed Model and all necessary components of my research are described in this chapter.
- **Chapter 7**: This chapter gives Evaluation and Results
- **Chapter 8**: Finally thesis is concluded with proposed future work in this chapter.

1.5 My Contributions

Specifically following contributions have been made in this thesis:

- Development of raw dataset with various network protocols and attacks at Hypervisor level.
- Extracting training and test datasets through data preparation and preprocessing to yield desired input vectors.

- Development of ruleset with Machine learning algorithm using a combination of fuzzy logic and genetic algorithm.
- Intrusion detection through classification of test data to identify anomalies in network traffic.

Chapter 2

Important Concepts in Virtualization

2.1 Introduction to Virtualization

Operation framework in typical PC frameworks offers level of observation over the equipment, on which many procedures can run simultaneously. This empowers a solitary OS to work on single framework (hardware) on a given time.

A Hypervisor is a piece of software that runs beside or beneath an OS which is designed to be an efficient and isolated duplicate of the real (physical) machine. Furthermore, a single hypervisor can also run on multiple networks hardware systems. However, we will only discuss Hypervisors related to single hardware system.

Hypervisors or Virtual Machine Monitors(VMM) are of two types, bare metal and hosted. Type 1 VMM is installed directly on hardware and has full control over all VMs using it. Type 2 VMM sits beneath an OS above the hardware.

2.2 System Virtualization Architecture

There are numerous utilitarian parts of virtualizations which are utilized to enhance the way virtualization is utilized and control.

2.2.1 Motivating Factors for Security using Virtualization

Confidentiality, integrity and availability being the prime components of computer security are achieved through their properties of a virtualized system which are termed as isolation, oversight and duplication respectively.

Isolation, covering confidentiality is achieved by placing OS inside a virtual machine thus separating software running on the OS and hardware itself. This technique allows few risky services to be run in a virtual machine[4], [5] whereas critical ones on other.

Oversight and duplication which covers integrity and accessibility are accomplished by utilizing a property of VMM called introspection which can catch and reestablish framework state and in this way an exceptionally helpful property of VMs.

2.2.2 Security in Virtualization

Security covers disclosure an alteration of data which includes, but not limited to the confidentiality, integrity and availability of:

- Software Data
- Software and Hardware Operational State
- Control and Network channels

Secure design of any system requires accomplishing a threat model comprising of following processes, vision, diagram, validate, identify threats and mitigate threats.

2.2.3 Security Threats resulting from Virtualization properties

There are certain security implications are needed to be deliberated before configuring a fully virtualized system. Security properties of a virtualized system can have double effect enhancing as well as compromising security. These threats[4], [5] occurred due to Hypervisor's trust model, transparent nature and introspection capabilities.

The major vulnerability area in a hypervisor system is the blind trust on its platform. Usual VMs cannot detect the hypervisor being fully transparent and this inability can have drastic consequences for some softwares.

VMM insertions, introspection, VM cloning, non-linear VM operations and software delinking from physical and hardware environment are few of the security threats that can create following issues in a virtualize environment.

- Inability to locate a VM
- Hidden or coward VM
- Additional consolidated hardware
- Physical location issues

2.2.4 Security Consequences from weakly Implemented System

Many security consequences hail from improper, incomplete and compromised implementation of virtualization requirement. Two main types are transparency breaches and resource control breaches. Virtualization transparency is breached[4], [5] when any of the three requirements is breached. This leads to information disclosure which deduces the presence of VMM. As VMM is the most important part of virtualized systems, compromised VMM puts on risk the whole system. If VMM software is interrupted, this will cause interruption to all VMs running upon it. Any alteration to VMM can affect all VMs and also poses the risk to underline hardware.

2.2.5 Control, data and software flows Implication

[8] Besides the security issues ascending as a result of virtualization properties, implementations of virtualization can also have security issues. For example, we know that VMM and VM data flow administrative channels and software install for guest VMs make virtualization a valuable but in real deployments this can add some additional attacks surfaces.

Threats pertaining to control channels are of unauthorized access and denial of service. Virtualization alone only offers a little extra protection for an OS running on it. The resource control property of VMM can be used to improve not only detection but also mitigation and recovery from compromised through introspection and modification of VMM.

2.2.6 Secure Implementation of Hypervisor System

System administrator must focus on rollout planning and managerial issues, hardening threat prevention and vulnerability detection measures[4], [5], intrusion detection and prevention measures and recovery and continuity protection measures. These items are consistent with any secure system implementation however hardening and threat prevention has particular concern for system virtualization platform.

2.3 Intrusion Detection

An Intrusion Detection System (IDS)[7] collects and analyzes data from a computing system on continuous basis, the main aim is to detect any intrusions. IDS has two main types working with analysed data. [1]: Network-based IDS (NIDS) – works on monitoring network traffic flowing through the systems, and Host-based IDS (HIDS) – works on watching local activity on a host, for example processes, network connections, system calls, logs, etc. Host-based intrusion detection is its relative fragility and its considered weaker: an HIDS agent should be installed in the machine to monitor and collect system activity data. An intruder can deactivate or temper this agent, in order to facade his/her presence, thus turning the detection system useless.

Methods used for evaluating collected data in order to detect intrusions can be categorized in: signature detection, when collected data is compared to a base of earlier known attacks arrays (signatures), and anomaly detection, when collected data are compared to previously collected data demonstrating the normal activity of the system. Normality deviations are then highlighted as

threats. We will be using anomaly detection for our work.

Chapter 3

Related Work

In [11], Kayvan Atefi et al. used Visual Architecting Process methodology to evaluate different algorithms and hybrid models used for intrusion detection. They further used the essential parameters to compare the results. The idea proposed was to utilize different techniques to evolve the best process for finding accurate and acceptable results in IDS thus reducing false positives and false negative alarms in the network.

Hossein et al. proposed an Intrusion Detection model using Genetic Algorithm (GA) in [12]. They combined developmental and traditional algorithms to accelerate search speed thereby reducing the false alarm rate on training dataset. They constructed classifying models based on the training data observed from the GA taking K-Nearest Neighbors (KNN) Algorithms into consideration.

Asry et al. made an endeavour to detect anomalous behaviour in network traffic using Classification and Regression Tree (CART) and Fuzzy Logic in KDD Cup' 99 dataset [13]. They used CART to build rules or models and those mod-

els were implemented using Fuzzy inference engines. Fuzzy logic was used in performing tests and the resulting rules were used for classification to produce desired results. They proposed that CART combined with Fuzzy Logic can be effectively used to build a classifier that can be helpful for observing anomalies in an intrusion detection system. The average accuracy obtained through their experiments was considerable to gain from the hybrid effect of both utilized techniques.

In [14], Nguyen et al. proposed a system of detecting anomalies using Neural Networks on GPU using KDD Cup99 dataset. They implemented and evaluated a graphic processing unit which accelerated the process of intrusion detection utilizing the parallel processing capability of neural networks. The technique can eventually be utilized to produce an effective IDS based on anomaly detection. The proposed work allows the system to meet a set of requirements such as time constraint management, robustness, high processing speed and reconfigurability.

Xiaohua Li et al. proposed a joint machine learning and human learning design approach to make the training data labelling tasks in linear regression problems more efficient and robust to noise, modelling mismatch and human labelling errors in which they applied active learning for the search of better quality training data and used outlier detection process for the removal of human labelling errors. They also derived thresholds in order to remove data which was prone to errors to keep the sparsity of the labelling data [15].

The efficacy of DARPA and KDD datasets have sublimed over a period of time. This aspect has been addressed in our research by incorporating present day attacks and protocols in the developed dataset. The precise application of

machine learning algorithm on this dataset to achieve desired results is distinctive feature of our work.

Chapter 4

Installation of Server

4.1 Powering On

First of all we've to verify that the ability wire has been connected which standby power is on.

- Shortly when power is applied to the server, the SP OK/Fault diode blinks because the service processor boots. The front panel begins flashing slowly, indicating that the host is in standby power mode.
  - Press and unleash the Power button on the front panel.
- When main power is applied, the Power/OK LED begins to blink quickly and system boots.
  - The power-on self-test (POST) takes several minutes to complete.

4.2 Loggin on to Oracle ILOM

Power on the system. Oracle ILOM Administrator account is not needed for this

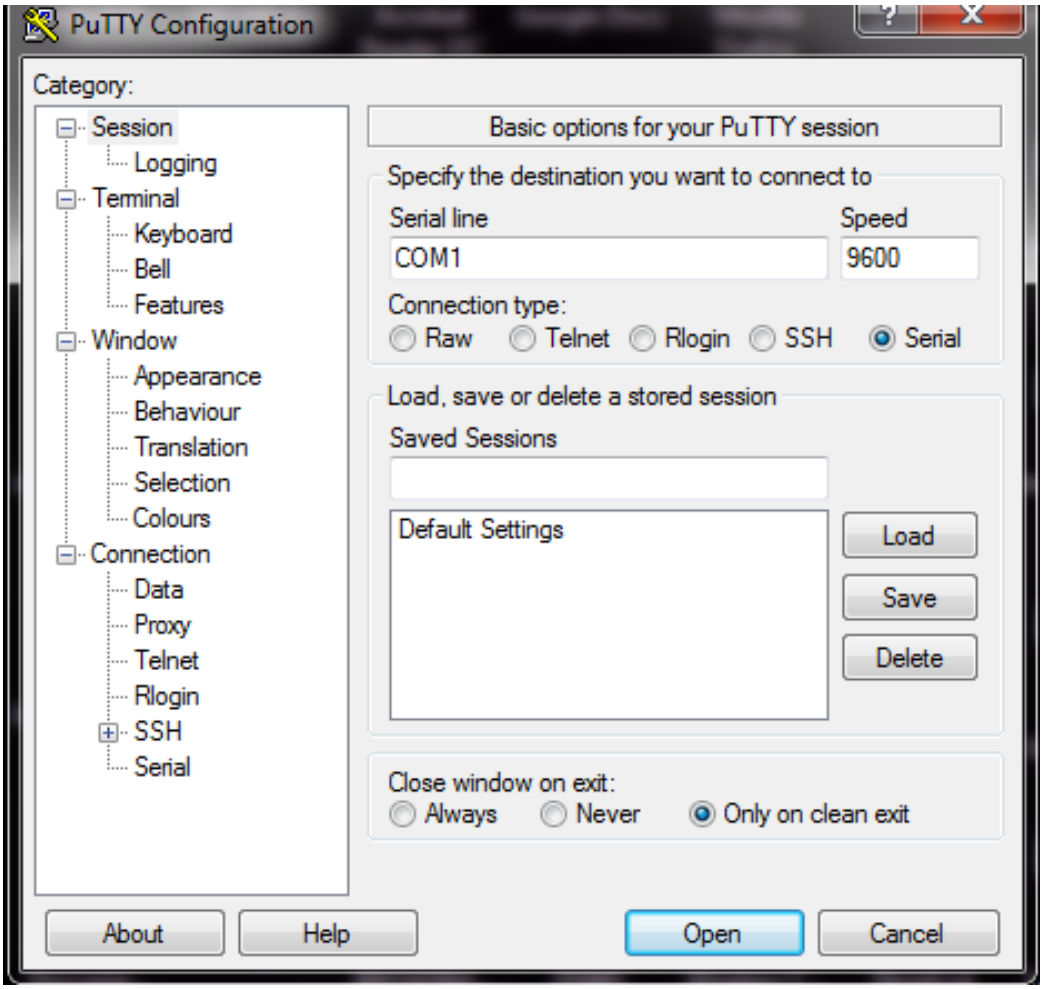

Figure 4.1: Putty Console for ILOM Configurations

step.

- Verify that serial console connection to the server is secure and operational.

- Serial communication settings are configured as below:
  - 8N1: eight data bits, no parity with one stop bit.
  - 9600 baud
  - We disable hardware flow control (CTS/RTS)
- Once the connection has been established between your serial console and Oracle ILOM. A login prompt to appears.
- Log in to the Oracle ILOM command-line interface (CLI) using an Administrator account. Default command prompt $->$ appears, indicating that you have successfully logged in to Oracle ILOM.

4.3 Network Configurations

Apply required network settings after the configurations are done.

- For network configurations(IPv4), use the cd command to navigate to the network directory.
  - show /SP/network
- Do one of the following:
  - Following command will be used to view the settings assigned to the server by the DHCP server.
    - $-> cd/SP/network$

– If there is no DHCP server, or if you want to assign settings, use the set command to assign values for the properties listed in the following table. For example

  * $->set/SP/network/pendingipdiscovery=static$
  * $->set/SP/network/pendingipaddress=XXX.XXXX.XXXXX$
  * $->set/SP/network/pendingipnetmask=255.255.255.0$
  * $->set/SP/network/pendingipgateway=XXX.XXX.XXX$
  * $->set/SP/network/commitpending=true$

4.3.1 Log In to Oracle ILOM Using an Ethernet Connection

Procedure after ILOM IP address has been set using serial connection.

Secure shell (SSH) session is used to log in to Oracle ILOM by specifying your Administrator account user name and the IP address of the server SP.

Enter Administrator password.

4.3.2 Exit Oracle ILOM

Typing exit will end the Oracle ILOM session, at the CLI prompt.

4.4 BIOS Setup Utility

Following steps are used to access BIOS Setup Utility:

- Power-on or power-cycle the server.
- F2 key is pressed, while the system is performing POST.

- Attach a bootable USB containing operating system and set boot priority.

Utility.png

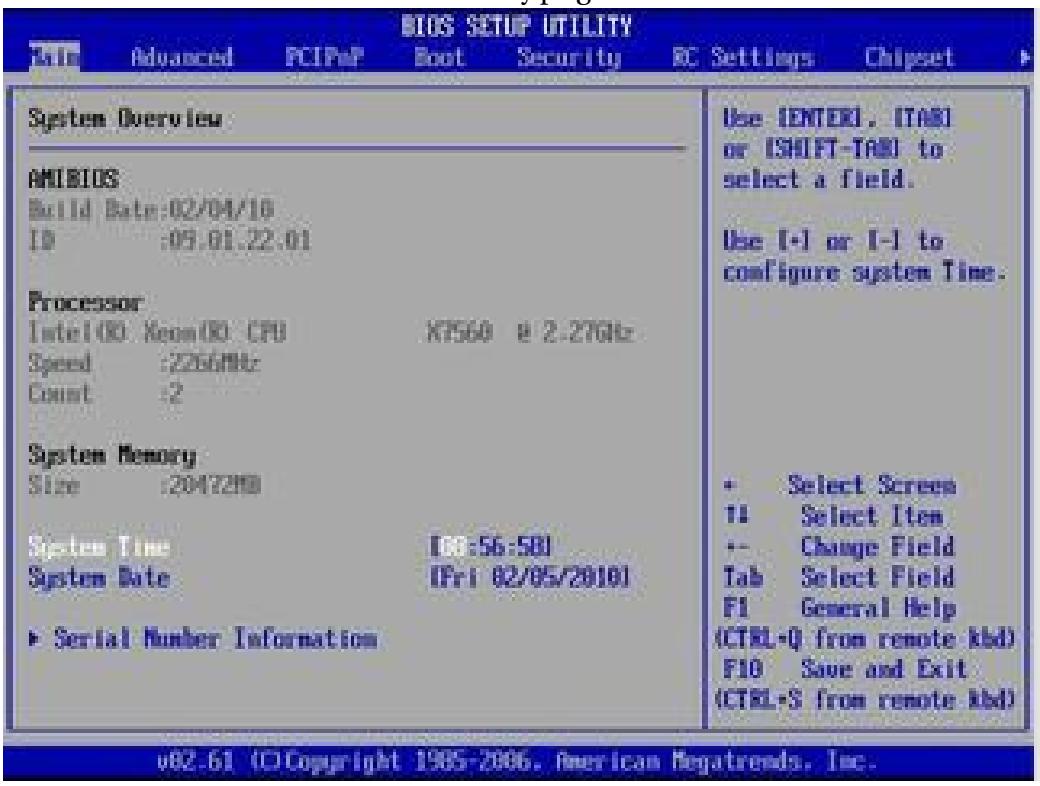

Figure 4.2: BIOS Setup Utility Console

4.5 Installing ESXi

Installations of ESXi was interactive as well as scripted, and several options were available to boot the installer and access the installation media as shown in figure:

4.5.1 Storing the ESXi Installation Media and Script

A USB flash drive can be used for storing the ESXi installation media and installation script that will be used during scripted installation of ESXi. When multiple USB flash drives are present on the installation machine, the installation software searches for the installation media and the installation script on all attached USB flash drives. We should note that we do not use the same USB flash drive as the storage location for the installation media and as the installation boot device.

4.5.2 Pre-requsites

We must have the following files and hardware to create the USB with ESXi installation media and script:

- ISO image of ESXi
- Installation script (Kickstart file)
- USB flash drive

4.5.3 Installing ESXi Using the Interactive Mode

We used the ESXi CD/DVD to install the ESXi software onto a SATA, hard drive of our server.

- Insert the ESXi 6.0 Installable media through bootable USB.
- Restart the machine.
- Set the BIOS to boot from the CD-ROM or USB device.

- On the Welcome screen, press Enter to continue with the installation
- Read the VMware end-user license agreement and accept it by pressing F11

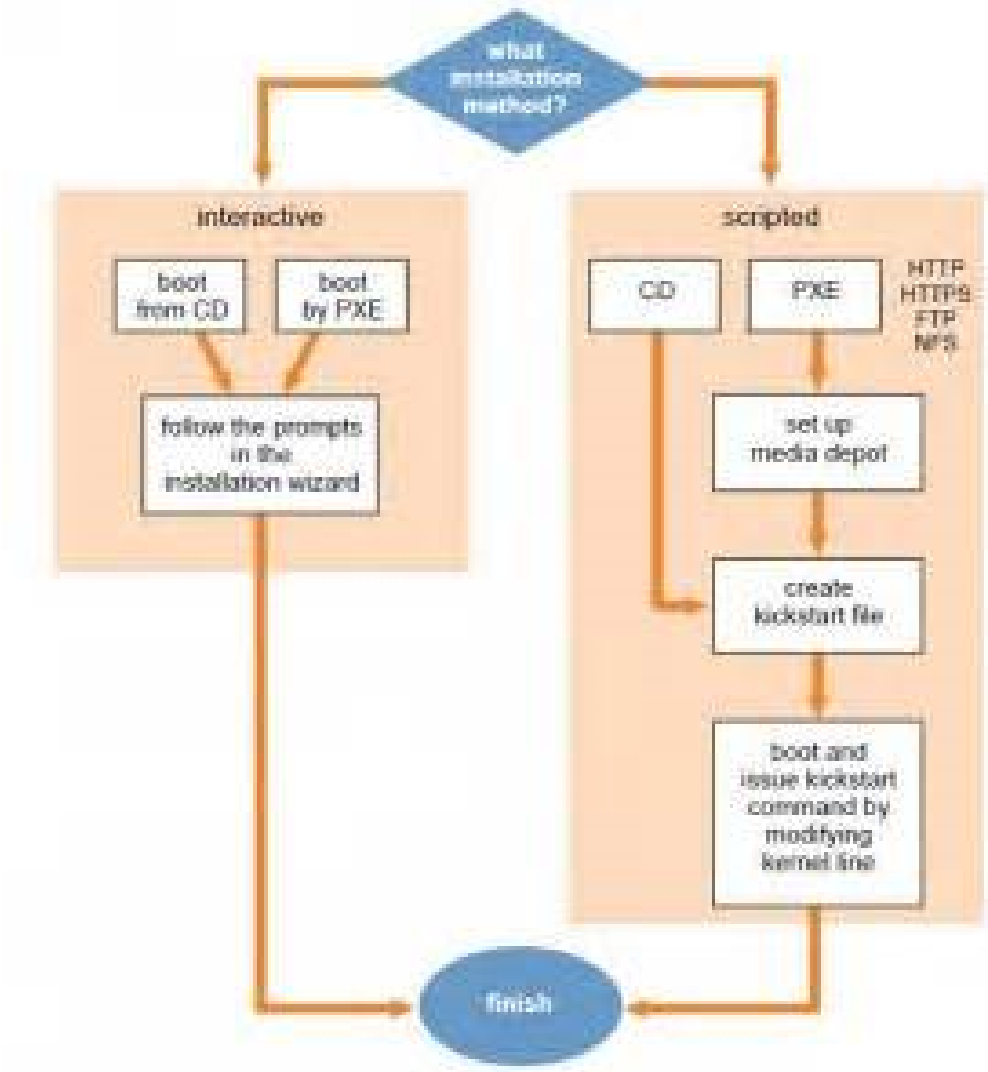

Figure 4.3: Installation Modes

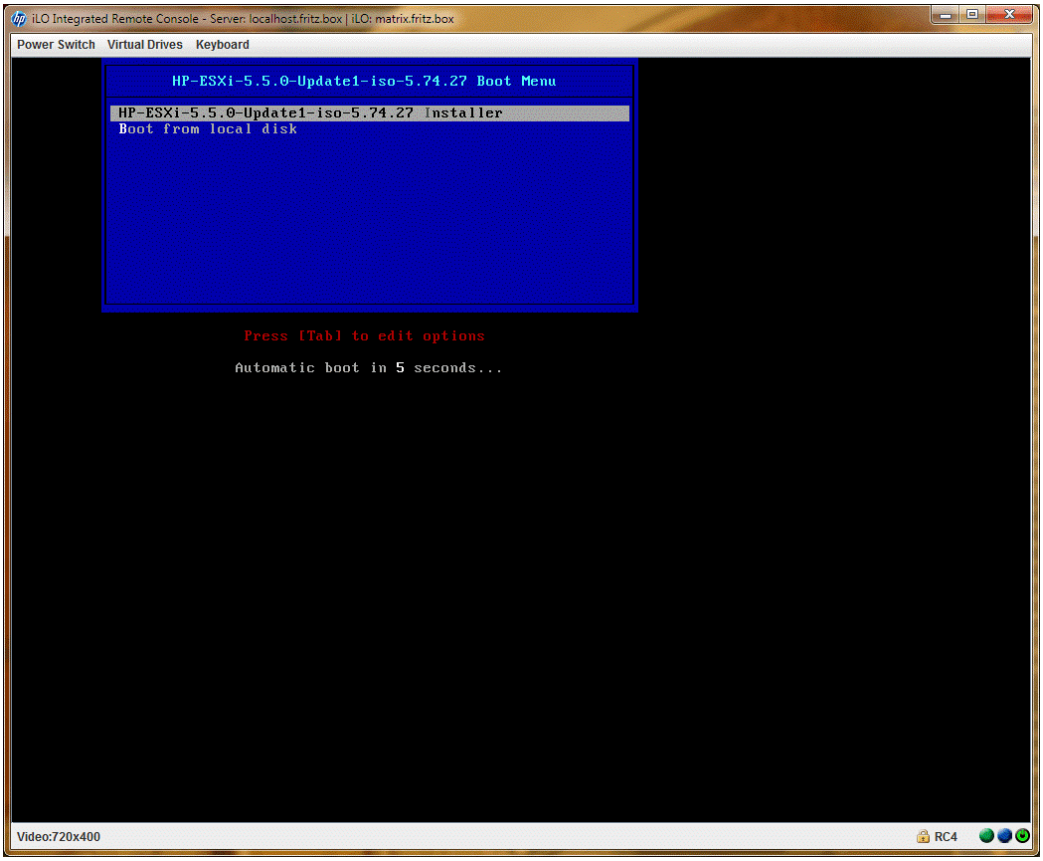

Figure 4.4: System Boots After the Restart

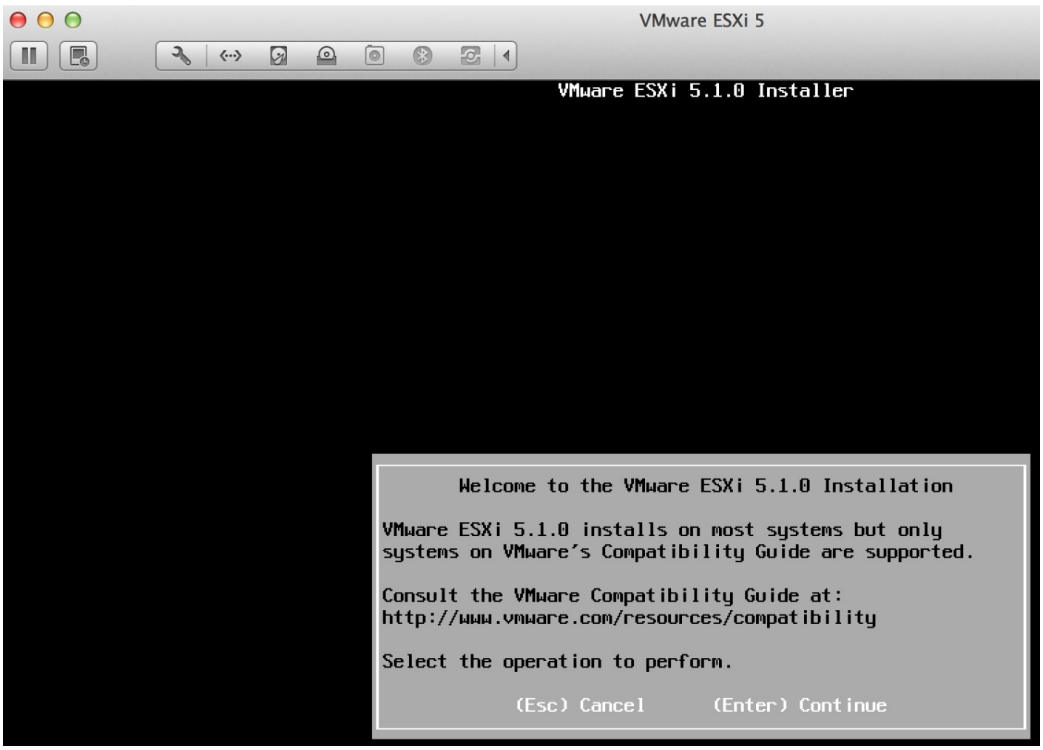

Figure 4.5: EXSi Installation Begins

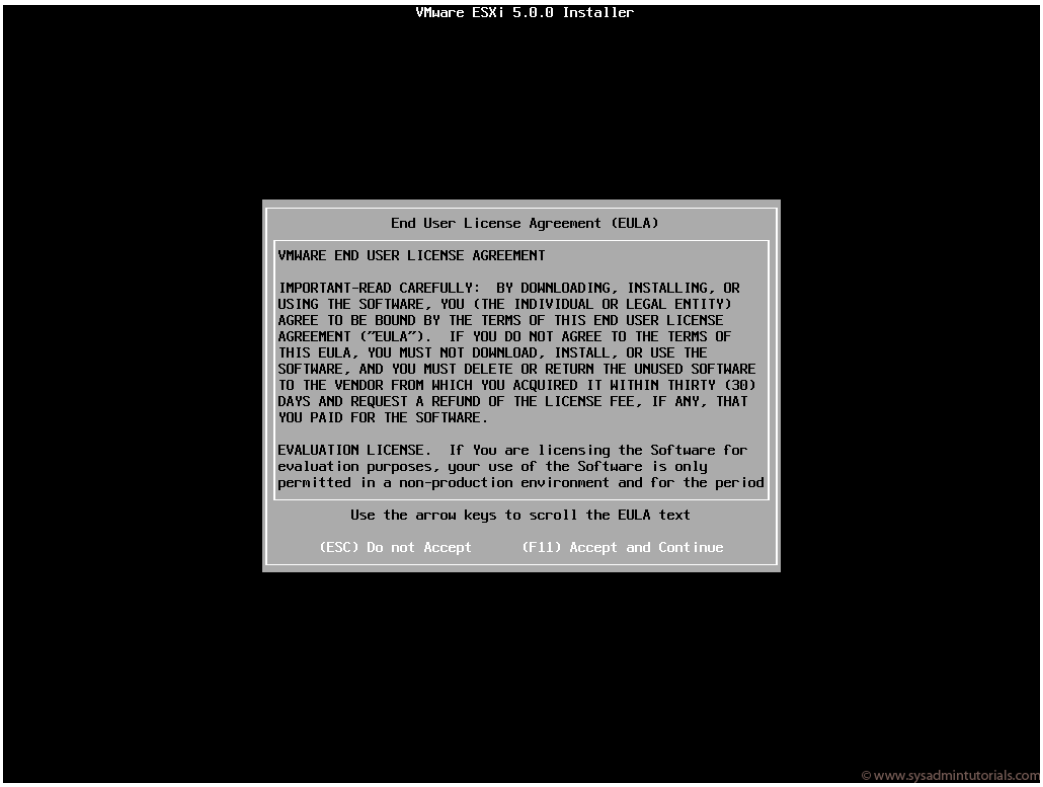

Figure 4.6: Accepting User Licence is Mandatory for the Installation to Complete

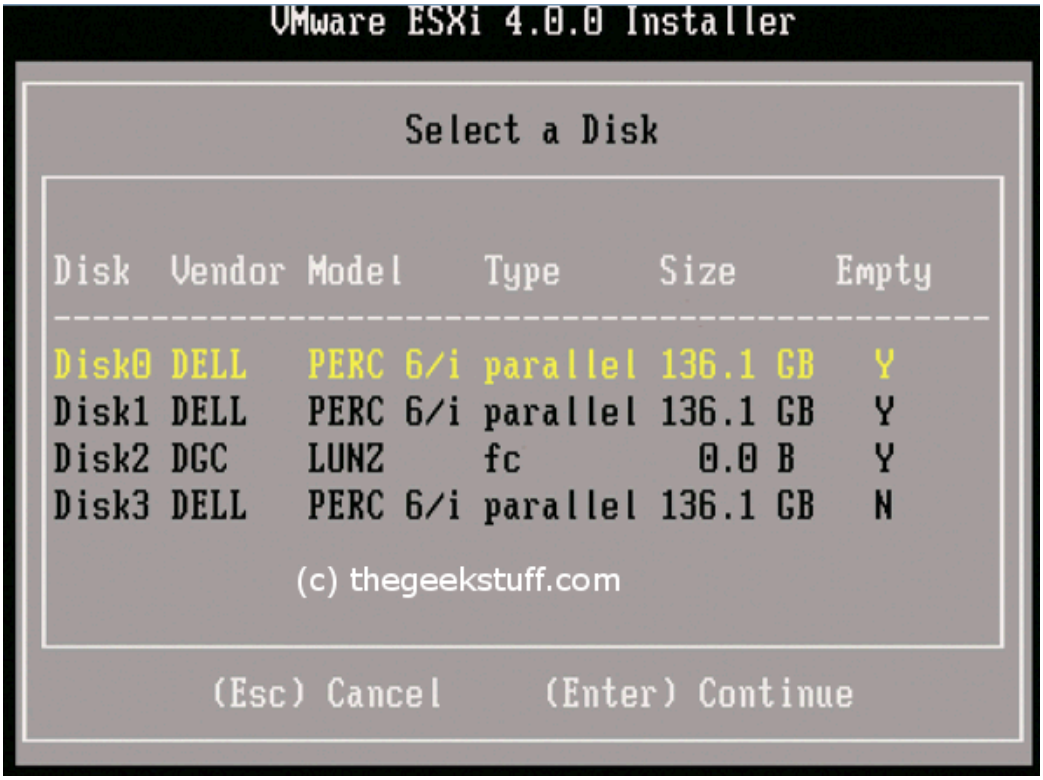

Figure 4.7: Selecting Harddisk/ Media for Installation

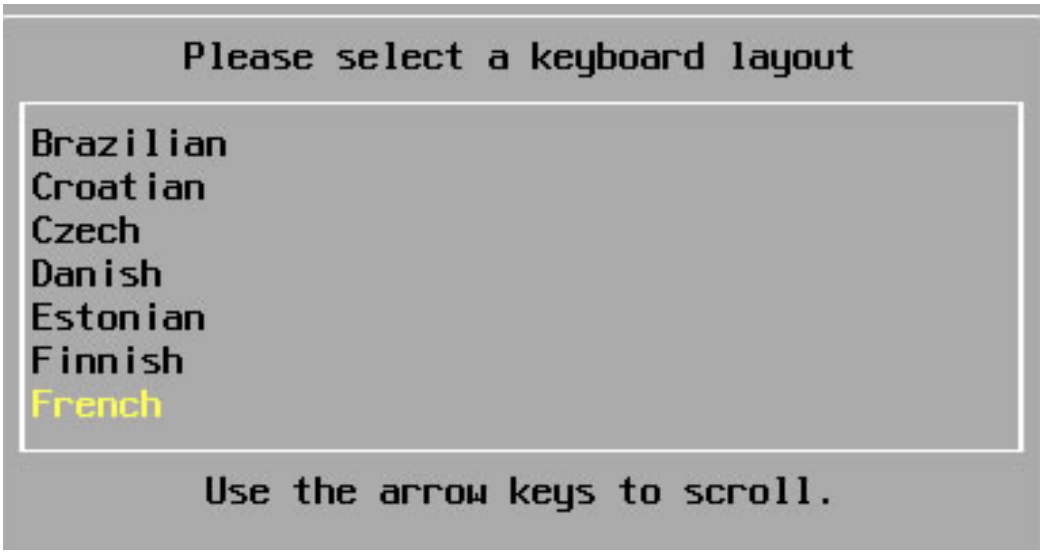

Figure 4.8: Selecting of User Language

Figure 4.9: Setting of Root Password for Administrator Account

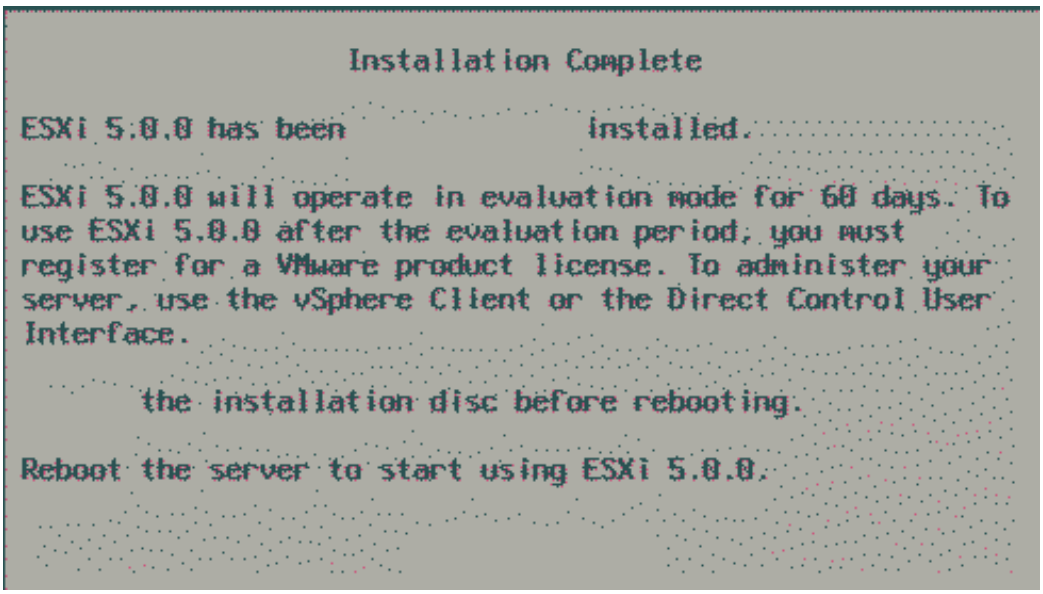

Figure 4.10: System will Reboot Automatically After the Installation is Complete

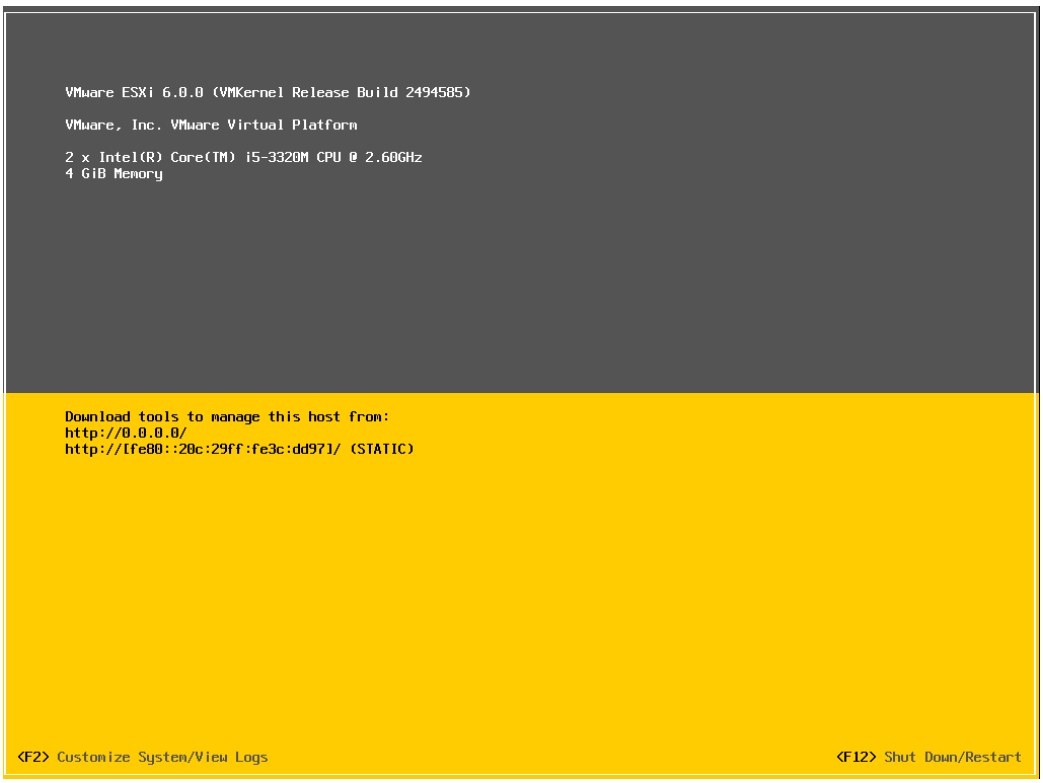

Figure 4.11: This Screen shows a successful Completion of Installation

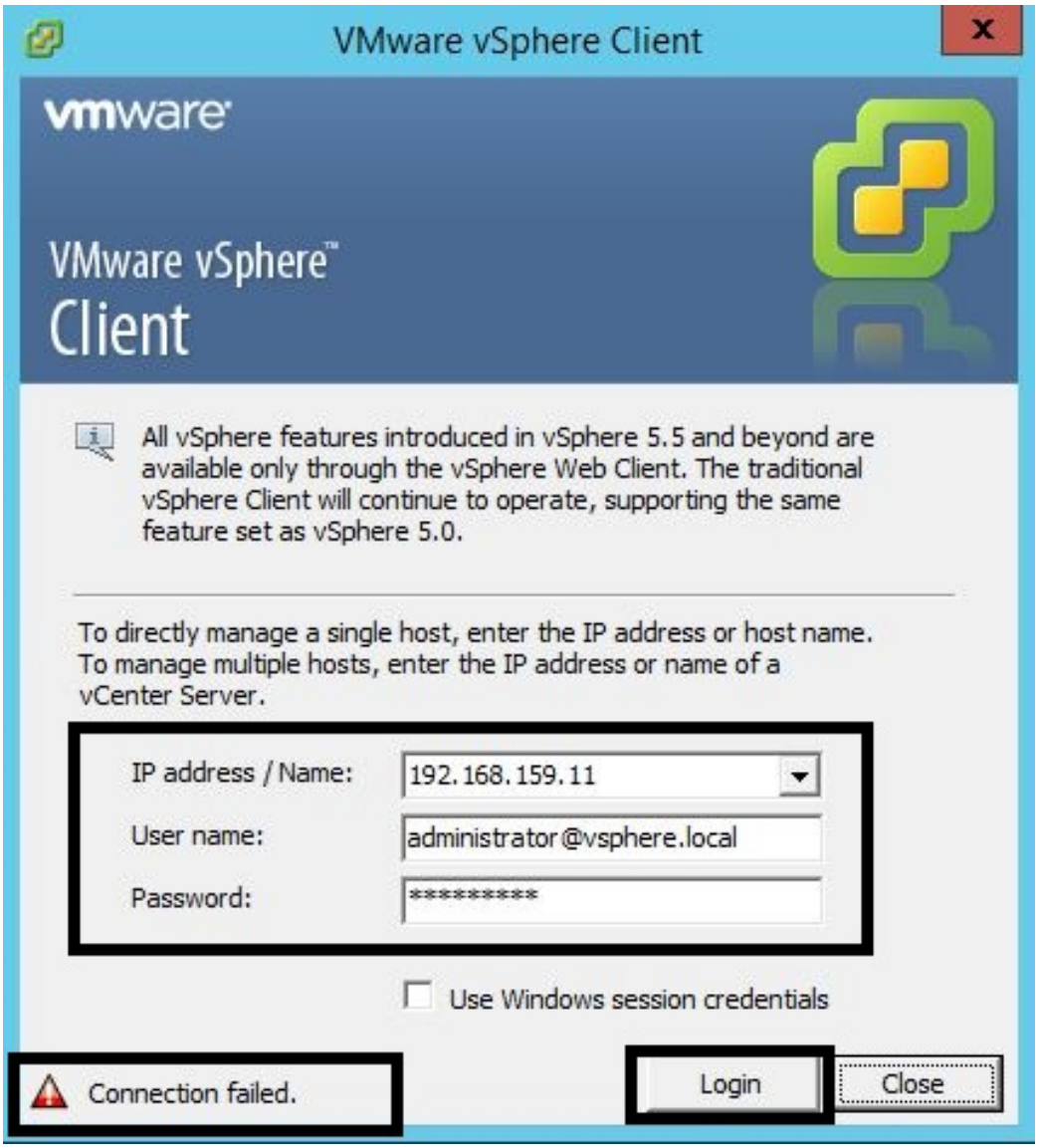

Figure 4.12: After Installing The Hypervisor, We must install VMware VSphere Client on our laptop to access our Hypervisor

Chapter 5

Data Logging in Hypervisor

5.1 Tools used

5.1.1 Hardware Calls / System Calls

“OS kernel is often accessed through System Calls. Certain tasks are there which can only be done if we are running the process in kernel mode. for example, interacting with hardware etc.“

5.1.2 Virtualization Setup

We configured 30 x Virtual machines on single Oracle blade server. For our experimental purpose.

5.1.3 Solaris BSM Auditing

“When enabled, the Solaris Basic Security Module (BSM) will produce a particularly elaborate audit path for all processes on the system. the only description

of BSM auditing that i have been able to come back up with is to imagine running truss–the Solaris call tracing tool–on each single method on the system and saving the ensuing output to a file. BSM really provides even additional elaborate data than that."

5.1.4 Support Vector Machine

SVM has 2 input streams one is that the path stream used for coaching purpose[10] and one is original information stream from production system, which provides the anomaly detection capability.

5.1.5 Metasploit Framework using MSFconsole

"The msfconsole[7] is probably the most popular interface to the Metasploit Framework (MSF). It provides an "all-in-one" centralized console and allows you efficient access to virtually all of the options available in the MSF. "

5.1.6 Kali Linux

Kali Linux may be a Debian-based Linux distribution geared toward advanced Penetration Testing and Security Auditing. Kali contains many hundred tools geared toward varied info security tasks, like Penetration Testing, Forensics and Reverse Engineering. Kali Linux is developed, funded and maintained by Offensive Security, a number one info security coaching company.

5.2 Proposed Experimental Setup

Trial setup[9] was composed of 4 modules, namely Raw data set, Pre-processing, data set and labelling as shown in Figure 5.1.

Raw data was processed into recognizable and usable data, after getting workable data final data set was collected and labelled as per the requirements.

This trial data set was then fed to our software for machine learning , which trained itself as per the input data.

Application server was configured and measured attacks with metaspoiled framework using kali linux were used for depicting production system anomalies, which give us a real system monitoring capability using linux inherent basic Security monitoring tool (BSM). The test values from BSM were then fed to out own developed software to give us the required anomaly detection figures./par

Software data learning has two input streams one is the trail stream used for training purpose and other is original data stream from production system, which paves way for anomaly detection capability./par

5.3 Collection of Trial Data

The experimental dataset was developed utilizing an oracle sunfire server. 30 Virtual machines were establised under three VLANs in a simulated environment. Figure 5.2 illustrates the data set generation process. The step wise process is explained as under:

5.3.1 Data Logging in Hypervisor

ESXi 6.0 was used to simulate five Windows 7, four Windows 8.1 and one Kali Linux VM in each VLAN. The network logs were collected over a period of five days for four weeks. Required data was collected in PCAP format. The metasploit framework and Kali Linux were used to generate attack traffic. Latest attack information was obtained from Common Vulnerabilities and Exposures (CVE) website. Our data set has 7 main attacks categories including DoS, Fuzzers, Reconnaissance, Shellcode, Worms, Backdoors and Exploits along with the normal network traffic. The Hypervisor was then setup to obtain the logs of all network activity for the prescribed time period.

5.3.2 Dataset Feature Extraction

Logs of the Hypervisor were used to create PCAP files using the packet capturing tool. The dataset features were then extracted using network traffic auditing and monitoring tools to give rise to 47 major attributes of the dataset in the form of basic, flow, content, connection, time and labeled attributes and saved in a database where rows representing records with distinct attributes in columns. Further more, the outcome of the database was exported to CSV format for further processing.

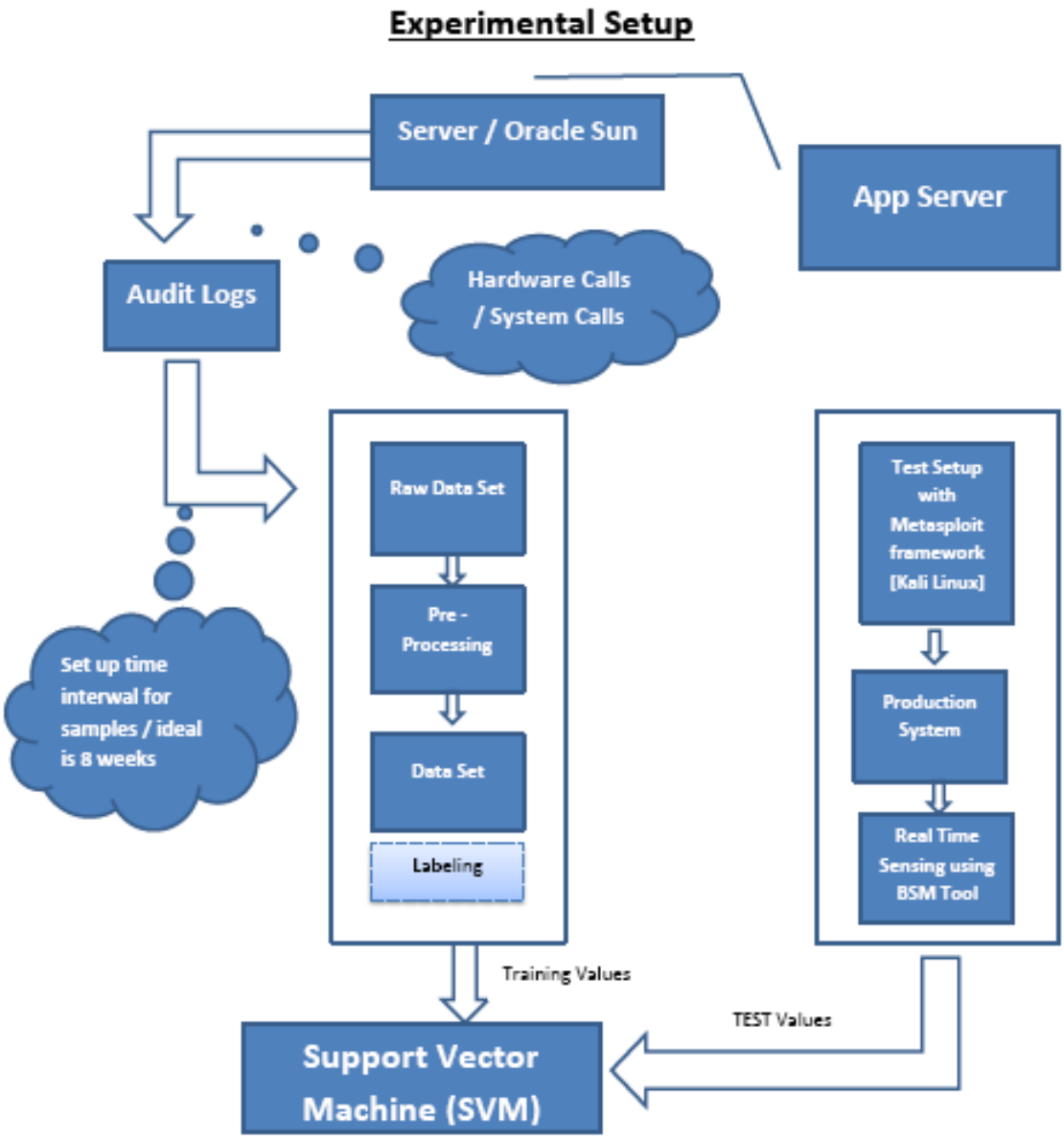

Figure 5.1: Basic Building Block for my Research.

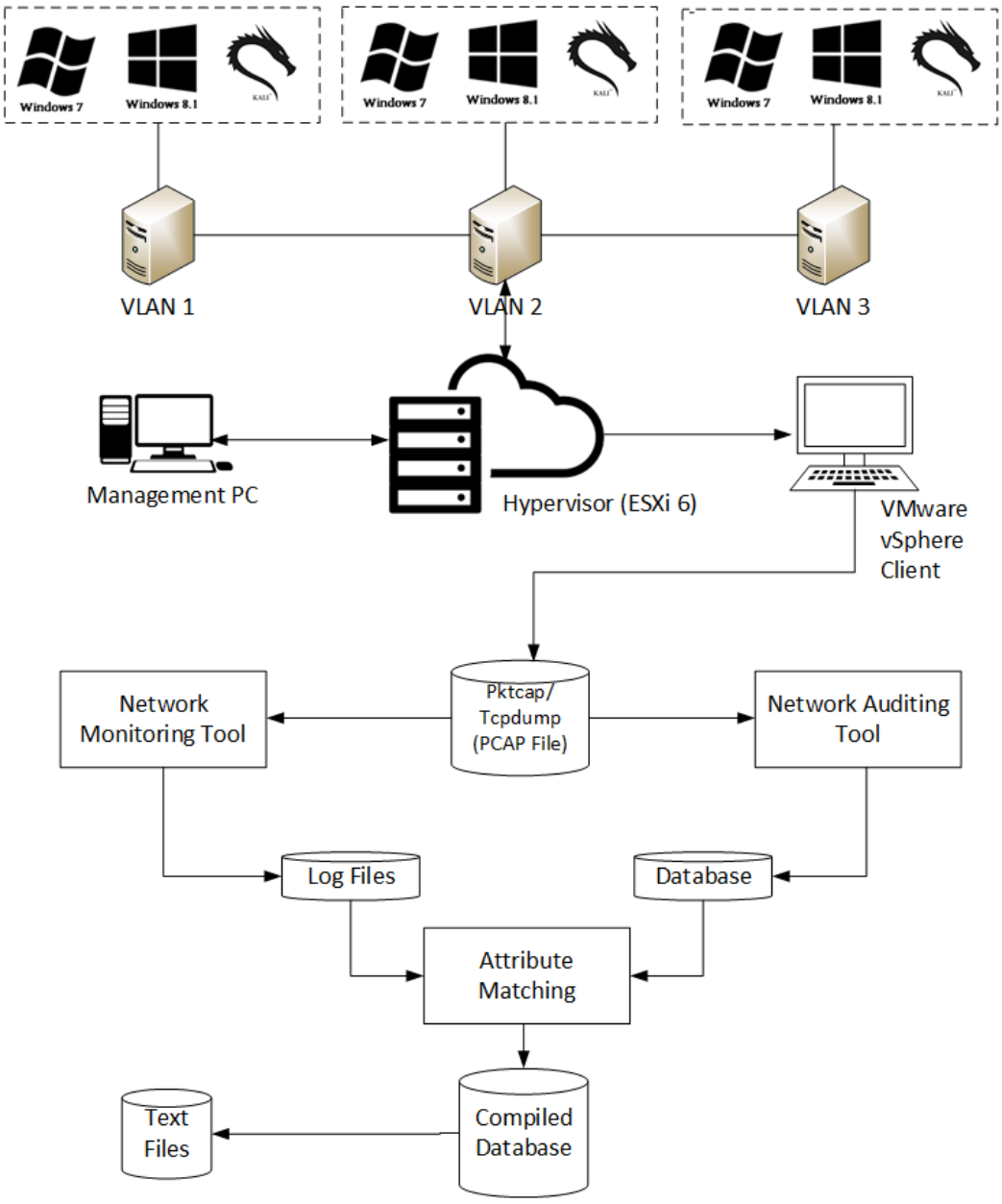

Figure 5.2: Dataset Generation

Chapter 6

Proposed Model

Our proposed model is mainly divided into three stages being Data Preparation, Data Pre-processing and Application of hybrid machine learning techniques for classification. The proposed model is depicted in Figure 6.1. The stage wise implementation is described as under:

6.1 Data Preparation

The tcpdump contained in the prepared dataset was utilized for our proposed model. This dump was analysed and a total of 47 attributes for each record were listed. Table 6.1 shows the distribution of training and test data in the dataset.

Serial	Detail	Training Data	Test Data
1.	Normal Records	69064	20992
2.	Attack Records	47452	14336
	Total	116516	35328

Table 6.1: Distribution of Training and Test Data.

6.1.1 *Normal / Attack Feature Vectors*

The tcpdump of the dataset was created for both normal and attack traffic. The vectors from both traffic were collected separately.

6.1.2 *Labelling*

In this stage, the normal and attack vectors were flagged by labelling as normal or attack record so that the user could easily understand it.

6.1.3 *Mixing*

After labelling both records were combined in a single pool and displayed as a table from the csv file.

6.2 Data Pre-processing

The data pre-processing stage is combination of two processes, Feature Reduction and Data Filtration.

6.2.1 *Feature Reduction*

In Feature Reduction stage the dataset is retrieved from the database and attributes are reduced to only Record ID, Duration of connection, Type of services (protocols), Source IP address, Destination IP address and Type of connection (with attack name). These are the essential features as which are mandatory to recognise a connection in a network and used for further processing.

6.2.2 *Filtration*

In Data Filtration stage the reduced features are examined for any incomplete records. These records are filtered out and eliminated from the data to avoid processing flaws of the next module.

6.3 Application of Hybrid Machine Learning Techniques for Classification

The process of classification is proposed to be dependent on combination of two machine learning techniques. The utilised techniques in this paper are Fuzzy Logic and Genetic Algorithms.

6.3.1 *Attribute Extraction*

From the filtered, pre-processed, data three types of attributes were extracted; Binary, Symbolic and Time. The binary attributes helped to distinguish a record as normal or attack. Furthermore, the sub-attributes were allocated from the various types of attacks simulated in the dataset. The symbolic attributes, recognised a connection from the type of service used over the network to establish link between a source and destination. These attributes were translated in terms of network protocols like http, tcp, udp, smtp etc. Lastly, the time attributes referred to the time taken by a particular binary attribute using a specific symbolic attribute. Each specimen of binary, symbolic and time attribute were obtained from the dataset and stored in the database to process further.

6.3.2 *Fuzzy Parameters*

Fuzzy Logic [15], [18] was used to translate continuous attributes into linguistic terms to facilitate recognition of zero day or novel attack patterns in the network traffic. We used an approach to transform the time attributes into three different linguistic terms; Low, Medium and High to recognise those patterns which possibly present an attack like characteristic.

Furthermore, each record will be assigned a fuzzy value and the linguistic terms will be utilised to produce fuzzy parameters.

Algorithm 6.1 was used to effectively separate the time values in high , med and low categories:

We calculated Alpha (α), Beta (β) and Gamma(γ) using eq:6.1 the same time values:

$$\begin{cases} \beta = \frac{1}{no\ of\ records} \sum_{n=0}^{endofdata} time \\ \gamma = Maximum\ Time \\ \alpha = 2 * \beta - \gamma \end{cases} \tag{6.1}$$

6.3.3 *GA Based Rule Extraction*

GA is a machine learning technique inspired by evolutionary biology. Genetic Network Programming (GNP) is a further advancement to GA which employs focused graph structures instead of strings and trees of chromosomes.

We extracted the required attributes as shown in Table: 6.2 from the dataset depending upon following characteristics:

Algorithm 6.1: Generation of Fuzzy Values

Data: Max_{time}, Min_{time} from stored dataset.
Result: Fuzzy Parameters for Continuous time values.
Variables: Stings: Value, Fuzzy values. Float: f_{low}, f_{med}, f_{high} ;
Initializations: $T1 = \frac{Max_{time}}{3}$ & $T2 = \frac{Max_{time}}{2}$

```
for all data time records (Time) do
    if Time < T1 then
        Set Fuzzy Value = Low
    else
        if T1 ≤ Time < 2T1 then
            Set Fuzzy Value = Medium
        else
            if Time == T2 then
                Set f_med = 1.0
            if T1 ≤ Time < T2 then
                Set f_low = 0.5 & f_med = 0.5
            if T2 < Time ≥ 2T1 then
                Set f_med = 0.5 & f_high = 0.5
        if time ≥ 2T1 then
            Set Fuzzy Value = High
```

- Attack or Normal (A=0 or A=1).
- Time duration (Low, Med or High).
- Type of service.

Normal, Attack, Attack Types ($Attack1, Attack2 ..... Attack7$), Protocols ($Protocol1, Protocol2, ..$ and Time divided in ($low, medium\ and\ high$). Iterative programming was done to have a complete set or possible combinations:

category	**Attibute**
Normal(0) / Attack(1)	Attack 1
	Attack 2
	.....
	Attack n
Service used	Protocal 1
	Protocal 2
	.....
	Protocal n
Time	LOW
	MED
	HIGH

Table 6.2: Selected Attributes

Few categories are shown below:

- $Cat_1 : Attack1 = 1$
- $Cat_2 : Attack1 = 1 \ \& Protocol = 1$
- $Cat_3 : Attack1 = 1 \ \& Protocol = 1 \ \& count = low$
- $Cat_4 : Attack1 = 1 \ \& Protocol = 1 \ \& count = med$
- .....

- $Cat_{101} : Attack1 = 1 \ \& Protocol = 20$
- .....
- $Cat_{121} : Attack2 = 1 \ \& Protocol = 1 \ \& count = low$
- .....
- $Cat_g : Attackn = 1 \ \& Protocol = n \ \& count = high$

This is done until all possible combinations have been created using the above listed attributes and the corresponding values are stored in database as rule.

Two separate sets are generated for intrusions as well as normal attributes corresponding to each attack component. The process is illustrated in Figure 6.2.

Now once we have generated complete set of combinations, we calculated the class association rule mining utilizing the support and confidence values for each category in the generated set. For example if our generated data is set $M = \{cat_1, cat_2, cat_3, cat_4, cat_5 ..... cat_g\}$ and suppose we have have N set of tuples where each tuple O is a subset of M. X and Y are any two attributes mentioned in Table 2. We have to associate X in any tuple with other members e.g Y, where X and Y both are any two of the selected attributes from our dataset and $X \cap Y = \emptyset$. This is called association rule. $X \Rightarrow Y$ means that X will be an antecedent and Y will be a consequent.

The GA based rule extraction using fuzzy logic and class association rule mining effectively chains binary, symbolic and continuous values in a single rule by using sub attributes. Whenever a continuous attribute represented by

a judgment node with a linguistic term, the fuzzy value is applied to determine the transition from the current judgment node to the preceding node. Here the fuzzy value is the probability of moving a judgement node to the Yes side of the network. The total number of tuples in the database that are moving to True side of the judgment nodes is stored in the buffer of the processing node from where rule extraction starts based on node transition. The No side of the judgement node is connected to the next processing node. These rules are finally stored in a pool. Once the system is exposed to the test data, this extruded rules pool compares the connections with the new data and decision is made whether the connection is a normal network activity or an attack.

The whole process of rule mining is explained in Figure 6.3.

If the fraction of tuples containing X in $N = x$ then we can say

$$Sup(X) = x \tag{6.2}$$

Similarly,

$$Conf(X \Rightarrow Y) = \frac{Sup(X\ U\ Y)}{Sup(X)} \tag{6.3}$$

Based on confidence (Conf) and support (Sup) of this association of X and Y, rule of $X \Rightarrow Y$ will be calculated.

Suppose we have $Sup(X) = a$, $Sup(Y) = b$ and $Sup(XUY) = c$, then to find out the positive relation between the associated members we calculated Chi Square(χ^2) for every member in the set M. Let K be the total number of records in the dataset.

$$\chi^2 = \frac{K(c - a * b)^2}{a * b(1-a)(1-b)} \tag{6.4}$$

These values for support, confidence and χ^2 are calculated for every member of M set. The values which we considered for the three parameters were than the minimum values of these parameters.

6.3.4 *Updating Fuzzy Values*

These class association rules are stored in a central pool of rules. All the above calculated parameters including parameters of fuzzy function the support, confidence and Chi square values are stores in the same pool. Whenever some already stored rule is used by GNP, the value for chi square will be calculated again, and if this value is more than the previously stored value, it will replace the old entry. This is how these values keep on updating themselves.

6.3.5 *Test Data Input*

Test data is the part of raw dataset that has never been exposed to the system so far. This dataset contains both normal and attack traffic vectors. For our own convenience and understanding we flagged this sub-dataset to verify our results at the end.

6.3.6 *Degree Matching*

We perform GNP based classification using fuzzy logic on the data bases where both discrete and continuous attributes are present. For our test data set first of all we performed degree matching of our individual data item (using Algorithm

6.2) with the present rule set already extracted. Three items from the test data are pitched against the collected rule sets for both intrusion as well as normal rules.

We took Time duration ($Tnew$), protocal ($Protnew$) and attack type from test data set. Fuz_{val} was defined depending upon the time comparison with rule set values($tmax$).

$$Fuz_{val} = \begin{cases} Low\ (0.0), & if\ T_{new} < \frac{1}{3} t_{max} \\ MEDIUM\ (0.5), & if\ \frac{1}{3} t_{max} \geq T_{new} < \frac{2}{3} t_{max} \\ HIGH\ (1.0), & if\ T_{new} \geq \frac{2}{3} t_{max} \end{cases} \tag{6.5}$$

$$MtDeg_{Con,Dis} = MFn(Fuz_{Val}, Dis) \tag{6.6}$$

These defined fuz values will be compared with the complete list of rules and corresponding values where protocals and fuzzy rules match ((Dis) and (Con))respectively. This is done using following equation for the new connection. "C" is a Flag which will be 1 when protocol is matched and 0 otherwise.

$$match(new_{con}, g) = \sum_{i \in M} \frac{1}{Con + Dis} (MtDeg_{Con,Dis} + C) \tag{6.7}$$

Value of $match(newcon)$ ranges from 0 to 1. Where 1 means a full match where as 0 means a full mismatch. We calculate the average of all these calculated values in order to get the distance of this new data attribute from the stored rules set (g). $ERset$ is the total no of extracted rules that match to a certain new data.

$$Match_{avg} = \frac{1}{ERSet} \sum_{g \in ERSet} match(new_{Con}, g) \tag{6.8}$$

This $match(avg)$, as shown in figure 6.4, is calculated as $matchint(avg)$ for intrusion rule set as well as $matchnor(avg)$ for normal rule.

6.3.7 *Classification*

We suggest the normal vs intrusion value to the new data item depending upon the distance of this item from the average value of the calculated matches. For all values we resort to some standard statistical values by calculating standard deviation σ and mean μ for the calculated $matchnor(avg)$. Equations (6.10) and (6.11) provides result of normal an intrusion record:

We calculate classification constant (K) through Algorithm 6.3

$$Match_{intr}(avg) \geq Match_{nor}(avg) \tag{6.9}$$

$$Intrusion = Match_{nor}(avg) < (\mu - K * \sigma) \tag{6.10}$$

$$Normal = Match_{nor}(avg) \geq (\mu - K * \sigma) \tag{6.11}$$

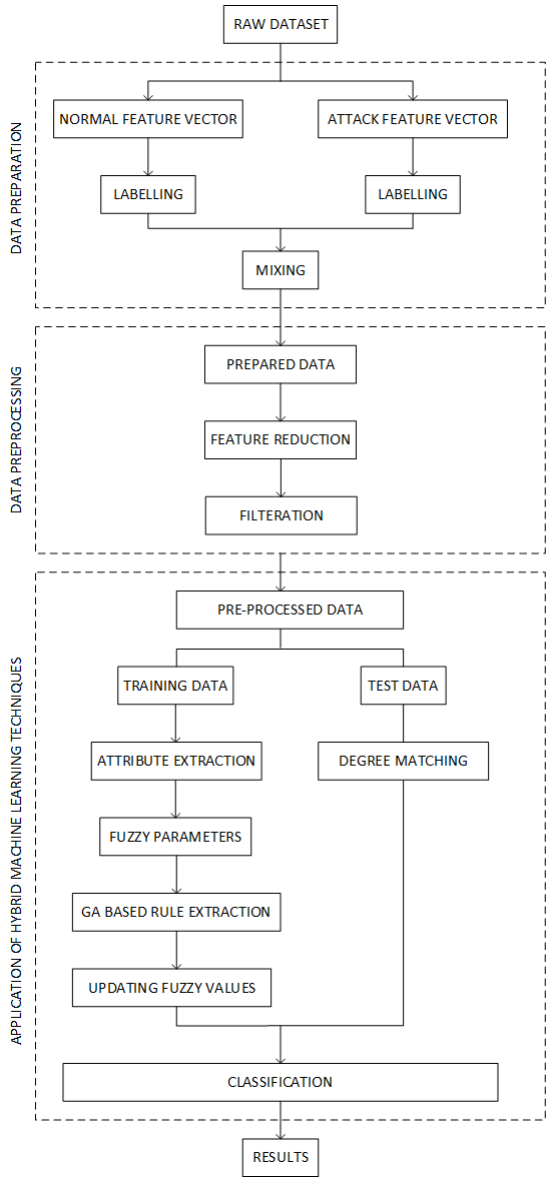

Figure 6.1: Proposed Anomaly Detection Architecture

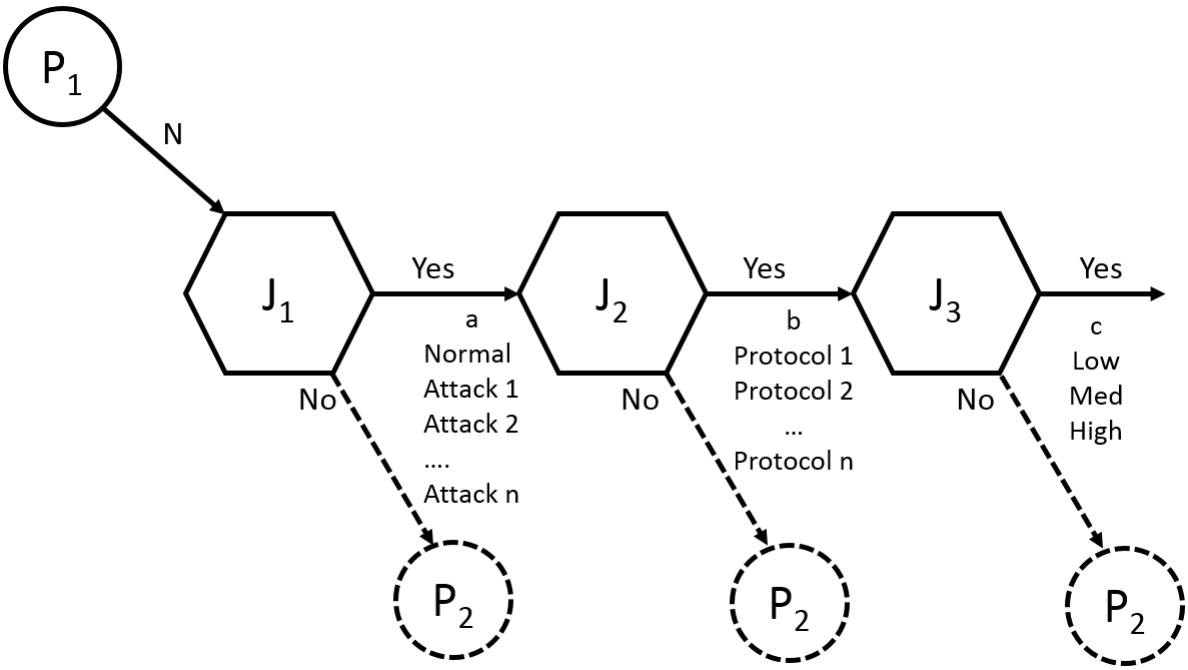

Figure 6.2: GNP Graph Structure Formation Methodology.

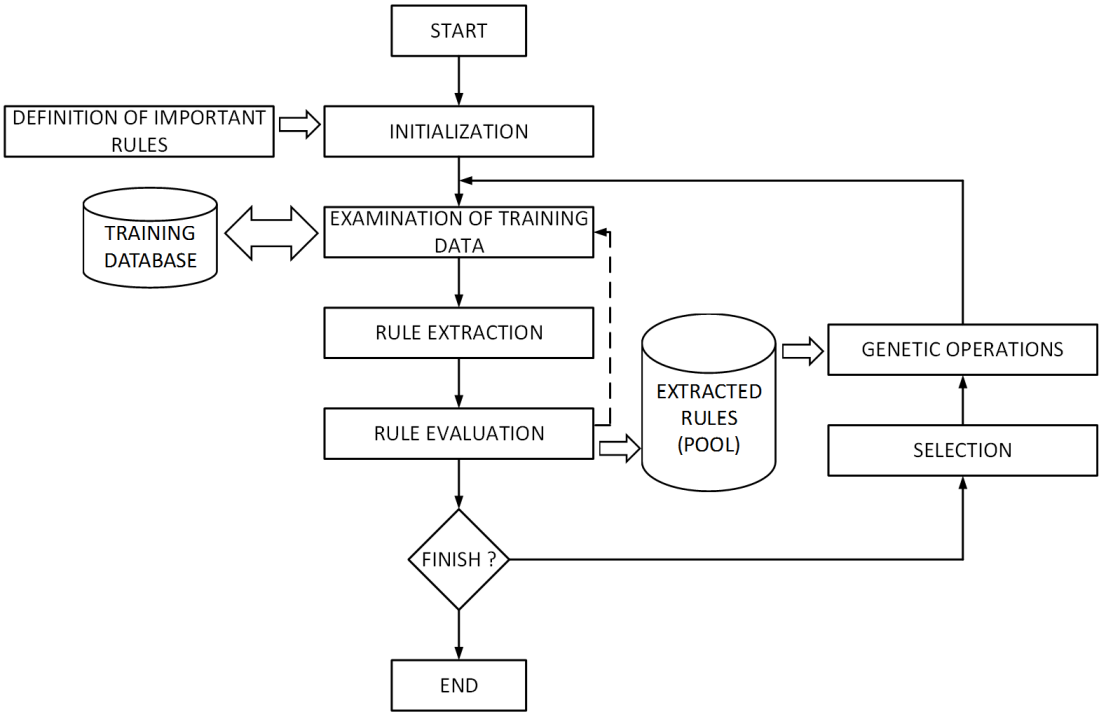

Figure 6.3: Rule Extraction Procedure.

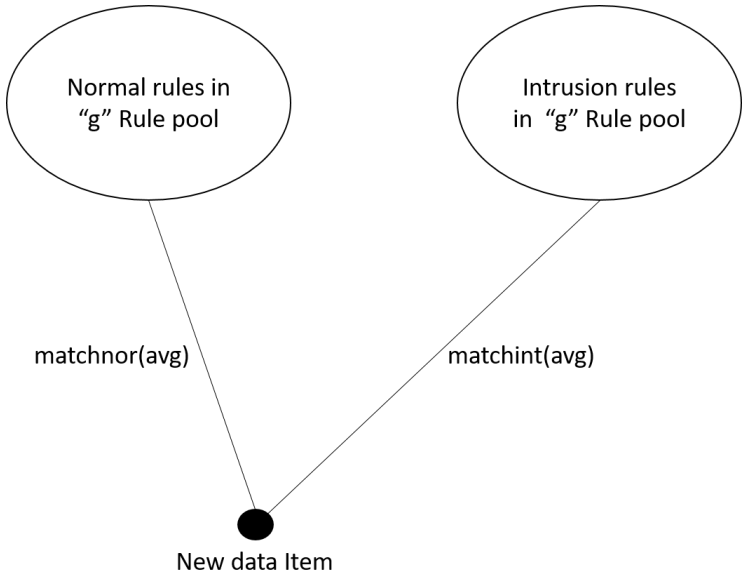

Figure 6.4: Classification Method by GNP-Based Fuzzy Class - Association Rule.

Algorithm 6.2: Degree Matching

Data: $RuleSet(g)$ from stored dataset.
Result: Matching Degree Using Fuzzy Values.
Variables: Integers: Con, Dis, C, ERSet ;
Initializations:
$(Con = 0, Dis = 0, C = 0, ERSet = 0, Match_{avg} = 0, Match(new_{con}, g))$

while $Not\ end\ of\ RuleSet(g)$ **do**
    $ReadRule_{no}$, Dis_{value}, Con_{value} and
    Protocal from rule set **if** $Con_{value} \neq true$ **then**
        Set Con = 0;
    else
        Set Con = 1
        Set Dis = 1
    if $ProtocalAttribute == Some\ protocal\ value$ **then**
        Set C=1
    else
        Set C=0
    if $Con == 1$ **then**
        if $(Con\ attribute == Fuz_{val})$ **then**
            $(SetMtDeg_{(Con,Dis)} = Fuz_{val})$
        else
            $(MtDeg_{(Con,Dis)} = 0)$
            $(Match(new_{Con}, g) = \frac{1}{(Con+Dis)}(MtDeg_{(Con,Dis)} + C))$
    else
        $(MtDeg\ =\ 0)$
        $Match(new_{Con}, g) = Match(new_{Con}, g) + MtDeg_{Con,Dis}$
    if $ERSet == 0$ **then**
        $(Mk = MtDeg_{(Con,Dis)})$
        $(ERset = ERSet + 1)$
    $(Match_{avg} = \frac{1}{ERSet}(Match(new_{Con}, g)))$
    $(intr[CO] = Match_{avg})$
    set $Match_{avg} = 0$, $ERSet = 0$, $Match(new_{Con}, g) = 0)$

Algorithm 6.3: Classification Constant

Data: $nrm[i]$,$intr[i]$ from stored data.
Result: Classification Constant.

Variables: Float: $Pk = 0$, $K = 0$;

for *all records i* **do**
    $Pk = (1 - nrm[i])+(intr[i] - 0)$
    K=K+Pk
    $i = i + 1$

$K_{avg} = \frac{K}{No\ of\ records(i)}$

Print the value of K_{avg}

Chapter 7

Evaluation of our Work

Evaluation of our Anomaly detection system, has been done on our own created dataset as well as DARPA 98 dataset.

7.1 Evaluation

The training database was made intrusion free, because for anomaly detection model, the training was required to be done on intrusion free dataset. This dataset only had normal records. Pre-processing was carried out, after that a certain amount of attributes were included in every record. New judjement function was configured in JAVA for generating the rule set.

A huge number of ruleset is extracted using the already configured generation method. Which included both intrusion data as well as Normal data records.

The testing database contains 35328 connection records, including 20992 unlabeled normal records and 14336 unlabeled intrusion records. As our train-

Algorithm 7.1: Classification

Data: $nrm[i]$,$intr[i]$ from stored data.
Result: Classification Results.
Variables: Integers: TN, TP, FP, FN & Float: $Pk = 0$, $K = 0$;
for *all records i* **do**
    $Pk = (1 - nrm[i]) + (intr[i] - 0)$
    Calculate distance from average MtDeg
    if $PK > K_{avg}$ **then**
        if $X == 1$ **then**
            TN=TN+1
        else
            FN=FN+1
    else
        if $X == 1$ **then**
            FP=FP+1
        else
            TP=TP+1
Print the values of TP, TN, FP, FN;

ing was done on intrusion free dataset, all type of intrusions, which were included in our own made dataset, were taken to be totally unknown.

Post classification using our classifier, we get the results given in Table: 7.1. Accordingly True Positive and True Negative rates along with Positive and Negative Predictive values has been calculated using Algorithm 7.1 using equations (7.1) to (7.5).

	Intrusion	Normal	Total
Intrusion	12240(TN)	2096(FP)	14336
Normal	3556(FN)	17436(TP)	20992
Total	15796	19532	35328

Table 7.1: Confusion Matrix

7.1.1 Our Contribution

Results of confusion matrix are shown in Figure 7.1. True and False Positive rates along with Positive and Negative Predictive values has been calculated using equations (7.1) to (7.5). The results gave us an overall accuracy of 84%.

$$TruePositiveRate = \frac{TP}{TP+FN} = \frac{17436}{20992} = 83\% \tag{7.1}$$

$$FalsePositiveRate = \frac{FP}{FP+TN} = \frac{2096}{14336} = 14.6\% \tag{7.2}$$

$$PositivePredictiveValue = \frac{TP}{TP+FP} = \frac{17436}{19532} = 89\% \tag{7.3}$$

$$NegativePredictiveValue = \frac{TN}{TN+FN} = \frac{12240}{15796} = 77\% \tag{7.4}$$

$$Accuracy = \frac{TN+TP}{TP+FP+TN+FN} = \frac{29676}{35328} = 84\% \tag{7.5}$$

7.1.2 Comaparison with DARPA 98 Dataset

We compared our results with that of DARPA 98 dataset, True positive rate for Darpa was slightly more than our dataset, whereas it had more False positive rate than our's. Positive predictive value of Darpa came out to be 93% as compared to our set which was 89%. Negative predictive value using Darpa was 66% where as our data set had 77% negative predictive value. Results for both

datasets are shown in Figure 7.2. It is noteworthy that DARPA dataset contained only four major attacks whereas in our case there are seven major attack vectors.

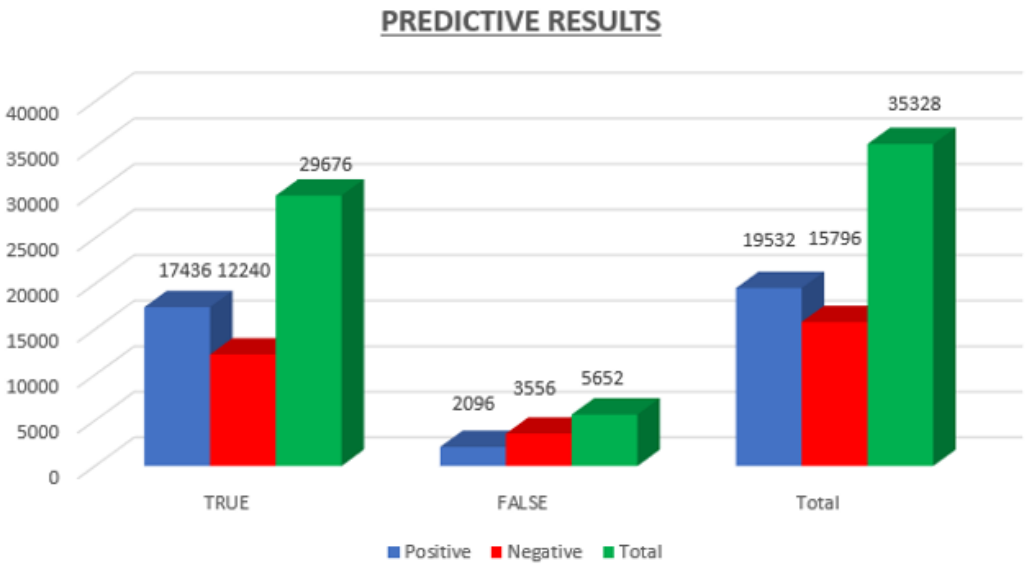

Figure 7.1: Predictive Results

7.1.3 Discussion

DARPA 98/99 is a well known dataset for intrusion detection and networking . The only limitation is that since DARPA 98/99 is old there was a need to incorporate more protocols and attacks. We introduced a total of seven new attacks alongwith corresponding protocols in our dataset. The comparison results show the same accuracy level for both datasets.

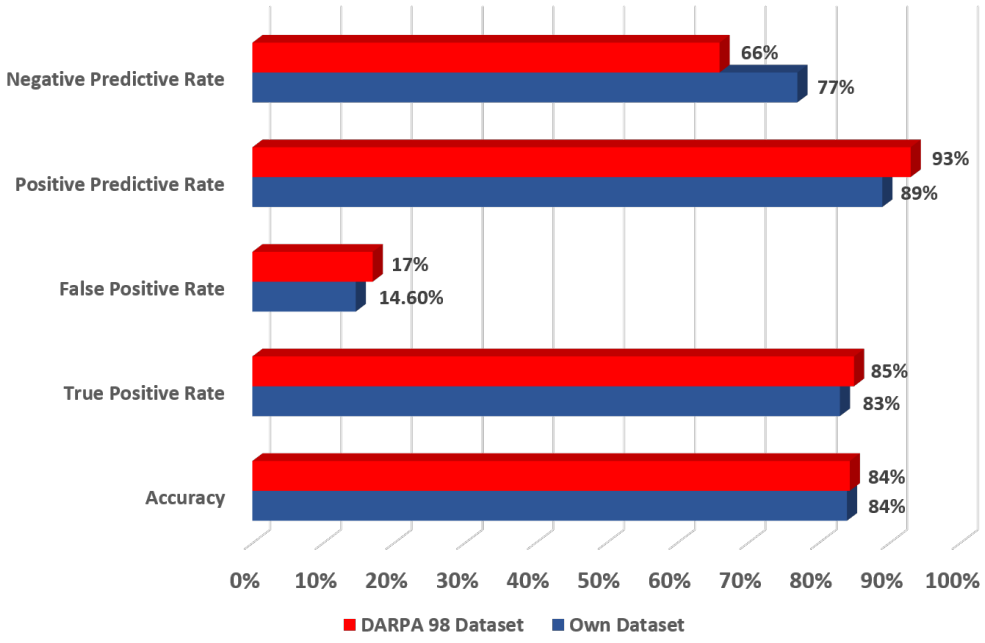

Figure 7.2: Comparative Results

Chapter 8

Conclusion

In this research, we have proposed and implemented an efficient system based on hybrid machine learning techniques for intrusion detection in hypervisors. The main achievement of this research was the creation of our own data set, which can be further used for development of new systems and research. We have proposed a rule based, GNP using fuzzy class association rule technique.

We have used time and discreate attributes in rule making, efficiently extraced a huge number of ruleset, which further proved to be great for classification.

A software has been developed, which extracts the ruleset and then classify the given data according to the given parameters. Anomaly detection technique for intrusion detection system has been developed using JAVA platform.

The results have been confirmed using our own dataset as well as the internationally known, DARPA 98. From the results we see that our scalability, True Positive rates and thus Efficiency has come out to be better than the similar techniques, which have been resorted to in many papers, where, close to this ,

work has been done.

An important character of our platform, is that it very efficiently, extracts the rule based data, which are significant statistically as well as it can be used in many further processes if developed and worked upon accordingly. This is a beauty of GNP that it gives us a very useful ruleset database for normal as well as intrusion records, from the given training dataset.

For signature based detection system, we can use both rulesets, but for our Anomaly based intrusion system, we only use the normal ruleset.

The experimental results were compared with DARPA 98 dataset and the accuracy was found equal. Since the proposed system performs its processing in offline mode, there are no additional overheads.

As a future work, it is recommended that density function of normal and intrusion accesses basing on fuzzy GNP be focused. By using this distribution, we can classify data into , "normal", "known intrusion", and "unknown intrusion" classes. And test data can be further worked upon to give better results. The software which has been developed can be further utilized for the development of hardware based intrusion detection system. I am also hopeful that, with a some further enhancement this software can be further developed into an Intrusion Prevension system. There is still alot of room for improvement and study for future purposes.

Appendix A

Acronyms

POST Power On Self Test

ILOM Integrated Light Out Manager

FTA Fault tree analysis

MTTF Mean time to failure

RAMS Reliability, availability, maintainability, and safety

Con Continuous Attributes

Dis Discreate Protocals Used

Intr Intrusion

Norm Normal Data Item

Bibliography

[1] Po-Jui Tsao, Yi-Feng Sun, Li-Han Chen and Chuan-Yu Cho. “Efficient Virtualization-Based Fault Tolerance”. In Insertional Computer Symposium (ICS), pp 114-119, 2016.

[2] Asraa Abdulrazak Ali Mardan and Kenji Kono. “Containers or Hypervisors, Which is Better for Database Consolidation?”. In IEEE 8th International Conference on Cloud Computing Technology and Science, pp 564-571, 2016.

[3] Andreas Blenk, Arsany Basta, Martin Reisslein and Wolfgang Kellerer. “Survey on Network Virtualization Hypervisors for Software Defined Networking”. In IEEE Communications Surveys & Tutorials (Vol: 18, Issue: 1, First quarter), pp 655-685, 2016.

[4] Ch.Ramesh and Dr.K.Venu Gopal Rao. “Security In Cloud Computing At Virtualization Level”. In International Journal of Computer & Mathematical Sciences (IJCMS), Vol: 3, Issue 8, pp 10-13, 2014.

[5] Priti Bali. “Cloud Computing: Security Issues in Infrastructure-As-A-Service Model”. In International Journal of Multidisciplinary and Aca-

demic Research (SSIJMAR), Vol:4, No 5, 2015.

[6] Emre Karakoc and Ferhat Dikbiyik. “Rapid Migration of VMs on a Datacenter Under Cyber Attack over Optical Infrastructure”. In HONET-ICT, pp 54-58, 2016.

[7] Rebecca Bace and Peter Mell. “Intrusion Detection Systems”. NIST Special Publication 800-31, 2001.

[8] Ming Zhao, Arun Kumar, G.G. Md. Nawaz Ali and Peter Han Joo Chong. “A Cloud-based Network Architecture for Big Data Services”. In IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing, 14th International Conf on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress, 2016.

[9] Xiaohua Li and Jian Zheng. “Joint Machine Learning and Human Learning Design with Sequential Active Learning and Outlier Detection for Linear Regression Problems”. In Annual Conference on Information Science and Systems (CISS), pp 407-411, 2016.

[10] Yagang Zhang. “New Advances in Machine Learning”. ISBN: 978-953-307-034-6, InTech, 2010.

[11] Kayvan Atefi, Saadiah Yahya, Amirali Rezaei and Siti Hazyanti Binti Mohd Hashim. “Anomaly Detection Based on Profile Signature in Network Using Machine Learning Technique”. In IEEE Region 10 Symposium (TENSYMP), pp 71-76, 2016.

[12] M. Hossein Ahmadzadegan, Ali Asgar Khorshidvand and Mehdi Ghalbi Valian. “Low Rate False Alarm Intrusion Detection System with Genetic Algorithm Approach”. In 2nd International Conference on Knowledge based Engineering and Innovation (KBEI), 1045-1048, 2015.

[13] Asry Faidhul Ashaari Pinem and Erwin Budi Setiawan. “Implementation of Classification and Regression Tree (CART) and Fuzzy Logic Algorithm for IDS”. In 3rd International Conference on Information and Communication Technology (ICoICT), pp 266-271, 2015.

[14] Nguyen Thi Thanh Van and Tran Ngoc Thinh. “Accelerating anomaly-based IDS using Neural Network on GPU”. In International Conference on Advanced Computing and Applications (ACOMP), pp 67-74, 2015.

[15] Xiaohua Li and Jian Zheng. “Joint Machine Learning and Human Learning Design with Sequential Active Learning and Outlier Detection for Linear Regression Problems”. In Annual Conference on Information Science and Systems (CISS), pp 407-411, 2016.

[16] Darapa Dataset:
“https://www.ll.mit.edu/ideval/data/”.

[17] KDD Cup 99 Dataset:
“http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html”.

[18] Kayvan Atefi, Saadiah Yahya, Amirali Rezaei and Siti Hazyanti Binti Mohd Hashim. “Anomaly Detection Based on Profile Signature in Network Using Machine Learning Technique”. In IEEE Region 10 Symposium (TENSYMP), pp 71-76, 2016.