RISK ANALYSIS AND MANAGEMENT OF SECURITY THREATS IN VIRTUALISED INFORMATION SYSTEMS USING PREDICTIVE ANALYTICS

Kapasa, Rosemary Mulenga

A Thesis submitted in partial fulfilment of the requirement of Liverpool John Moores University for the degree of Doctor in Philosophy.

April 2020

Abstract

The use of online server applications has increased in recent years. To achieve the benefits of these technologies, cloud computing, with its ability to use virtual machine technologies to overcome limitations and guarantee security and quality of service to its end user customer, is being used as a platform to run online server applications. This however brings about a number of security issues aimed specifically at virtual machine technologies. A number of security solutions like virtual machine introspection, intrusion detection and many more, have been proposed and implemented, but the question to combat security issues in near or even real time still remains. To help answer the above question or even move a step further from the existing solutions, which still use data mining techniques to combat the security issues of virtualisation, we propose the novel use of predictive analytics for risk analysis and management of security threats in virtualised information systems as well as design and implement a novel predictive analytics framework used to design build and implement the same predictive analytics model In this project, we adopt the use of predictive analytics and demonstrate how it can be used for managing risks and security of virtualised environments. An experimental testbed for the simulation of attacks and data collection is set-up. Exploratory data analytics process is carried out to prepare the data for predictive modelling. A linear regression predictive model is built using the results from the exploratory data analytics using linear regression algorithm. The model is then validated and tested for predictive accuracy using Naïve Bayes and logistic algorithms respectively. Time series algorithms are then used to build a time series predictive model that will predict attacks (DoS attacks in this case) in real time using new data. Designing and implementing the proposed predictive analytics model, which is aimed at monitoring, analysing and mitigating security threats in real time successfully demonstrates the use of predictive analytics modelling as a security management tool for virtualised information systems as a novel contribution to virtualisation security.

Thesis Organisation

This thesis has seven chapters, each with the relevance to contribute to answering the question of using predictive analytics in information systems for security management for virtualised systems. The thesis is split into four parts:

Chapter 1: Introduction

1.1 Introduction

Section 1.1 introduces the topic in question by providing an overview of what the research covers. The chapter provides a brief overview of the project through the thesis statement, which highlights the activities carried out in the thesis. An outline of the research aims and objectives of the project as well as the tasks that are followed in the predictive modelling process are also described. The motivation for this project together with the contribution to the body of knowledge and to the most needed virtualisation security in cloud computing are also discussed in this chapter.

Section 1.2 explores virtualisation technologies together with their security issues and expected security solutions in depth. This chapter describes the different architectures and types of virtualisation that are available. The advantages and disadvantages of virtualisation in cloud computing together with the collective security issues faced by the different types of virtualisation are highlighted. The implications of various activities such as weak implementations and data control that cause risks to virtualisation are discussed. Security vulnerabilities and threats to different attack vectors such as the hypervisor or the guest operating systems are also covered in this chapter.

Chapter 2: Evaluation of Existing Solutions

Chapter is divided into two parts due to the nature of the project, which endeavours to combine two different technologies to achieve one goal.

The first part of the chapter discusses and evaluates current and existing security solutions for virtualised environments with an emphasis on those that are relevant to the project. It describes the different methods and approaches used in implementing security solutions to various parts of virtualisation and identifies the techniques used to achieve these security solutions. This part also investigates how these security solutions are implemented, as well as the advantages and disadvantages of these solutions to virtualisation.

The second part of this chapter defines predictive analytics. This part of the chapter explores the concepts of predictive analytics; the tools and techniques used in predictive analytics and identifies those that are relevant to the topic in question. The chapter illustrates the traditional predictive analytic processes and provides a simple predictive analytics framework followed in the project. An in-depth insight of the various tools and techniques available, how they are used and implemented in different applications to achieve the required predictive value is also explored and explained. The chapter also discusses the relevance of real time monitoring in conjunction with analytics to help combat security threats in virtualisation. A discussion of what big data analytics is and how it relates to predictive analytics is discussed in this chapter.

Chapter 3: Design Structure, Methodologies and Experiments

Chapter 3 is split into two main parts; the first part discusses the design structure of this project and the second part of the chapter discusses the methodologies of the project.

3.1 Design structure

- In the design part of the chapter, an architectural overview of the set-up environment is presented. The chapter highlights and describes the various components like the programming applications interface, the reporting and analytics tools, the monitoring and data collection tools used, together with the host and the virtual machine matrices that are being monitored. It provides an architectural overview of how the test lab is prepared and set up for the experiments and then discusses in brief what each component in the architecture is and what its relevance to the project is.

4.2 Methodologies

- The second part of the chapter discusses the methodology followed. The chapter highlights and explains the proposed predictive analytics process, the data collection and preparation processes as well as an explanatory data analysis process together with the model design and deployment process. A model evaluation and testing process is also described in this chapter, the choice of predictive analytics platform with the choice of virtualisation used are also highlighted.

Chapter 5: Attack Simulation and Results

Chapter 5 presents the attack simulation framework together with the attack simulation processes followed in the project. The chapter identifies and discusses the types of attacks being simulated with an emphasis on denial of service attacks (DDoS). An illustration of how these attacks are simulated to gather information for the creation of the predictive analytics dataset is also presented. The results of the simulation and the sample of the generated data are also presented in this chapter. The chapter also discusses the attack simulation and detection components used.

Chapter 6: Predictive Analytics Applied and Conclusion

Chapter 6 where the application of predictive analytics is illustrated and chapter 7 where the conclusion of the project is presented.

Chapter 6 Predictive Analytics Applied

Chapter 6 illustrates how predictive analytics is applied to the generated dataset. The chapter presents how predictive models are built using different algorithms such as linear regression, logistic regression, Naïve Bayes and time series predictive models and then presents the results in the form of the graphs and listings. The chapter illustrates how the built models are validated using the same algorithms and how the built models are applied to new data to test the predictive value and lastly how the built models are deployed into day to day activities for the prediction of attacks DDoS in this case.

Chapter 7 Conclusion

Chapter 7 is the final part of the thesis. It provides the conclusion of the project by highlighting the summary of the thesis, the contributions made to the body of knowledge, suggestions for future work, limitations of the project and the final remarks.

Acknowledgements

I would like to thank various people who have helped and prayed for me throughout this PhD process.

Firstly, I would like to give praise, glory and thanksgiving to the Lord God almighty for his Mercy and Grace that he has awarded me to finish this PhD.

I would like to thank and dedicate this work to my father Mr John Marcelinus Kapasa, my hero and mentor. He has been my pillar and stronghold throughout my studies and most especially, through the trials, tribulations and pains of this PhD journey. I would also like to thank my mother Mrs Theresa Kapasa for her endless love, support, prayers and for always encouraging me never to give up, but to always trust and believe in God. To my sisters Dr Monica Kapasa-Mbewe, Mrs Mukupa Kapasa-Musonda, Miss Mwenya Kapasa and Mrs Bupe Kapasa-Kesho for their patience and encouragement and for always checking up on me. I would also like to thank Mr and Mrs Chisha for their support and words of encouragement.

I would like to thank my supervisor Dr Henry Forsyth and Mr Andy Laws for their patience, understanding and for their profound support and guidance in the process of researching and writing this paper. I would also like to thank Liverpool John Moores University for allowing me a chance to complete this PhD.

I would like to thank Father Henry Kalusa Mobela my mentor and spiritual leader for his endless counselling, encouragement, spiritual and emotional support.

To the technicians especially Mr Paul Cartwright for their endless technical support and to all my family and friends for their moral and spiritual support.

Table of Contents

Table of Figures

Table of Listings

Table of Tables

List of Abbreviations

IDS	Intrusion Detection System
IPS	Intrusion Prevention System
VMM	Virtual Machine Monitor
VM	Virtual Machine
PA	Predictive Analytics
PAM	Predictive Analytics Modelling
EDA	Explanatory Data Analytics
API	Application Programming Interface
NID	Network Intrusion Detection
DBMS	Database Management System
O/S	Operating System
VMM-DoS	Virtual Machine Monitor Denial of Service
VMMAPI	Virtual Machine Monitor Application Programming Interface
GDoS	Guest Denial of Service
MLDM	Machine Learning data Mining
NICE	Network Intrusion Detection and Countermeasure Election

Published Work

KAPASA, R. M., FORSYTH, H., LAWS, A. & LEMPEREUR, B. Predictive Analytics as a Security Management Tool in Virtualised Environment. 2015 International Conference on Developments of E-Systems Engineering (DeSE), 2015. IEEE, 102-106

1. Introduction and Background Information

1.1 Introduction

Despite the ever-growing demand for cloud computing and the many benefits it brings to both service providers and end users, security still remains a challenge in cloud computing. To solve the security issues of cloud computing, multi-tenant infrastructures are being used to help address security issues like integrity, confidentiality, availability, identification and authentication that affect cloud computing. Multi-tenant infrastructures consist of several virtual machines running on the same physical platform using virtualisation technologies (Brohi et al., 2012, Padhy et al., 2011, Suresh and Kannan, 2014). Cloud computing has adopted the use of virtualisation to take advantage of the security properties such as;

- **Confidentiality**, which is offered through the isolation of the operating system running on the guest operating systems from the physical hardware. Improved confidentiality is achieved through the isolation of not only the software running on the same hardware but also between the guest operating system and the physical system (Khalil et al., 2014, Sugerman et al., 2001). This provides a level of security because unsecure activities on one virtual machine can run alongside critical ones on a different virtual machine with a very low risk of compromise through the host operating system (Loganayagi and Sujatha, 2012, Lombardi and Di Pietro, 2011, Luo et al., 2011).
- **Integrity** by ensuring that the Virtual Machine Monitor (VMM) has full low-level visibility of operation and is able to intervene in the operations of guest machines. Integrity is achieved through one of the properties that ensure that nothing happens in the virtual machine that the hypervisor cannot observe or intervene in (Padhy et al., 2011). This property allows the hypervisor not only to have full low-level visibility of operation, but also to intervene in the operations of the virtual machines regardless of

the state. This gives cloud computing a level of security by ensuring that all virtual machines observed have their states captured, analysed and easily restored.

- **Availability** of resources by making it easy to capture and restore the system state with very little down time (Pearce et al., 2013, Popek and Goldberg, 1974, Sahoo et al., 2010). Availability is offered through the ease with which the virtual machine state is captured, duplicated and restored as well as through the abstraction which allows the virtual machine to be moved between platforms with very little downtime (Padhy et al., 2011). Virtualisation facilitates aggregation of multiple standalone systems into a single hardware platform (Chung et al., 2013). This hides the complexity of managing physical systems, which simplifies resource scalability that cloud computing requires.

Virtualisation is a very broad concept that goes back in time and it is applied to devices, applications, operating systems, storage systems, machines and networks (Pearce et al., 2013, Popek and Goldberg, 1974, Sahoo et al., 2010). Virtualisation is the use of an encapsulating layer known as the virtual machine monitor (VMM) that surrounds the operating system and provides the same processes as that of an actual physical machine. Virtualisation as defined by the National Institute of Standards and Technology (NIST) is the simulation of software and/or hardware upon which other software runs (Brooks et al., 2012, Pearce et al., 2013). The simulated environment is called the virtual machine (VM) also referred to as the guest operating system (guest OS) and is managed by a virtual machine monitor (VMM) also known as the hypervisor. A VMM is a highly privileged piece of software running parallel or underneath an operating system. It is designed to be an efficient, isolated duplicate of the physical machine (Brooks et al., 2012, Popek and Goldberg, 1974). A single VMM can run on multiple networked physical systems and offers a level of abstraction above the hardware, on which multiple processes can run concurrently. The abstraction of the operating system is when a virtual machine monitor creates a software environment equivalent to that of the host system

but which is decoupled from the physical hardware state giving virtualisation its security benefits.

An example of abstraction layer in a virtualised system is illustrated in Figure 1-1 below.

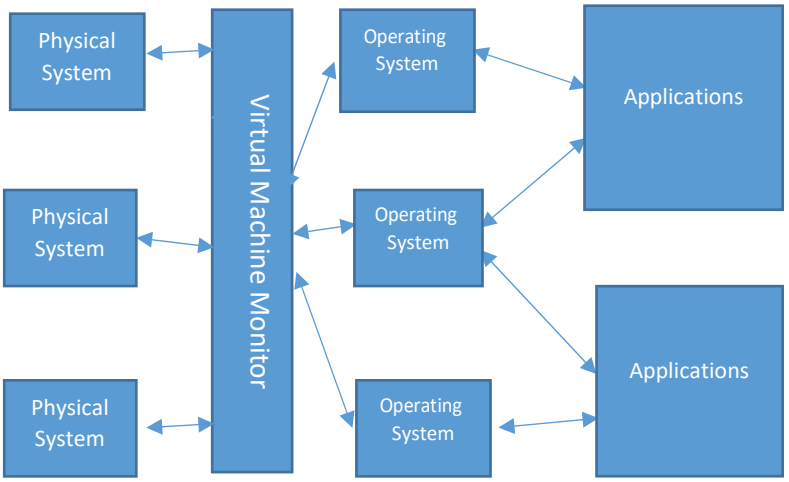


Figure 1-1 Extra layer of abstraction that a VMM offers

1.2 Virtualisation Architectures

Virtualisation comes in many forms distinguished simply by different computing architectures. There are two main types of virtualisation architectures known as:

1. Type I architecture also referred to as Bare Metal virtualisation which is installed directly onto the hardware as the primary boot system. The VMM in this type of architecture, executes at the highest level of privilege and has full control of all virtual machines installed on it. Entries and exits from the guest operating systems to the physical hardware are through the VMM as illustrated the Figure 1-2 below.

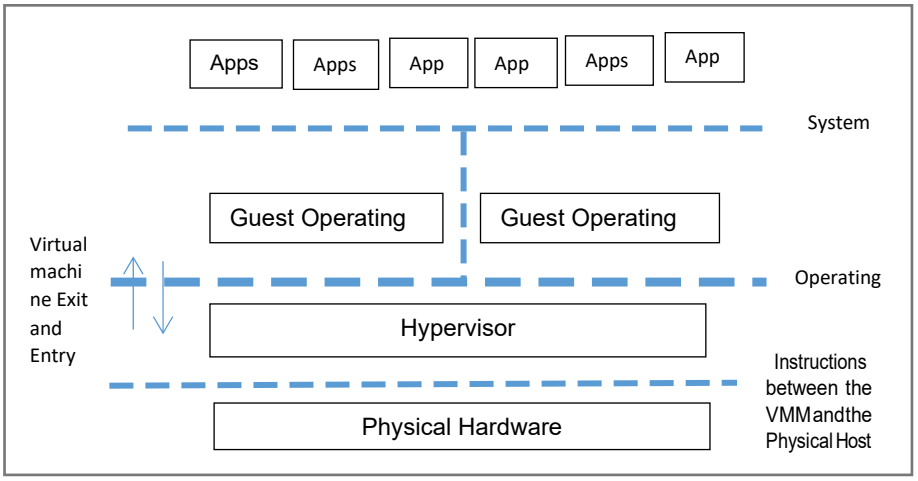


Figure 1-2 hypervisor installed directly onto the hardware

The use of a VMM in this instance introduces a new layer of abstraction into cloud computing. The hypervisor is responsible for emulating software or hardware configurations to the virtual machines. This type of architecture however, creates a large attack surface area for the environment because once the VMM is compromised the entire system from the hardware to the existing virtual machines will be affected. Security for this type of virtualisation architecture is dependent on the individual security of each of its components, from the hypervisor and host operating system to guest operating systems, applications and storage (Brooks et al., 2012).

2. Type II architecture mostly referred to as hosted virtualisation is installed right on top of a host operating system of the physical hardware, and shares the physical resources of the host operating system in order for it to handle Input/output (I/O) processes as illustrated in Figure 1-3 below:

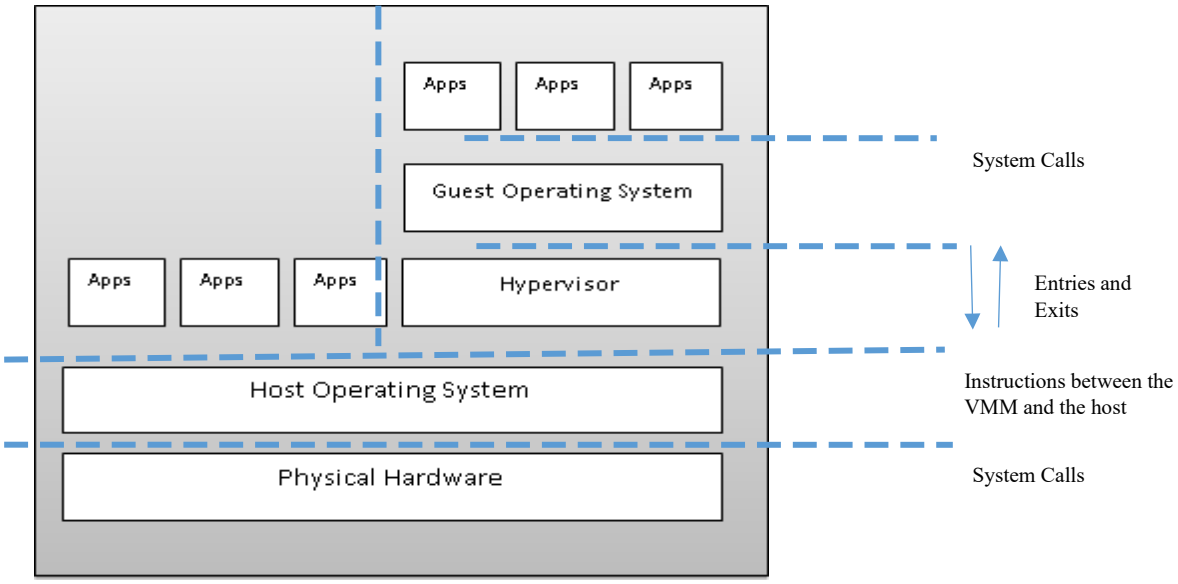


Figure 1-3 Type II Virtualisation Architecture

This type of architecture does not require hardware-specific drivers for I/O operations, but allows the use of virtual machines within an existing environment (Sugerman et al., 2001). The hypervisor in this type of environment needs to emulate every instruction it registers and must register to a certain set of instructions. This becomes a challenge because there is lack of proper specifications, and deciding if the guest machines have the right privilege from root operation is difficult.

The VMM controls the flow of instruction between the virtual machine and the underlying physical hardware as shown in Figures 1-2 and 1-3 respectively. Virtualisation can be applied to all the components of the computer system such as the server, applications, desktop and network. For example, server virtualisation also known as full virtualisation refers to the consolidation of the physical server where a VMM is installed directly onto the physical hardware and offers an abstraction of resources of the physical system from the guest operating system. It separates the underlying resources from the virtual machines. This type of virtualisation is where the operating systems with the applications they contain run on top of a virtual hardware. Each instance of the operating system together with the applications that run

on it are simulated in a separate virtual machine. The VMM controls the flow of instructions between the guest operating system and the physical hardware and has the ability to partition the system's resources as well as isolate the virtual machines resources from each other's resources so that each guest operating system can only access its own resources. The VMM in this case also provides networking capabilities to the virtual machines that allow them to communicate with one another whilst limiting access to the physical hardware. If there is more than one virtual machine available in the virtual environment, the VMM can implement virtual hubs and switches through the virtual network allowing the virtual machines to communicate with each other. This can pose a threat to the system as it makes it difficult to monitor and regulate traffic. Monitoring of traffic in a fully virtualised environment is done through a virtual machine that has privileges and has full visibility of the network traffic using the API allocated to it (Brooks et al., 2012). This however, can be an operational risk because the physical network tools that monitor traffic as it flows across different routers and switches, might not have the ability to monitor traffic that is running in a virtualised environment. The APIs may also provide additional ways for attackers to attempt to monitor network communications that may lead to performance degradation or denial of service activities to occur due to high volumes of traffic (Chung et al., 2013). Full system virtualisation is widely used for a variety of applications, such as the consolidation of physical servers (Armstrong et al., 2008), isolation of guest OSs, and software debugging (Bratus et al., 2008).

Although a Type-II VMM is hosted, full virtualisation is not necessarily a Type-II VMM. Full virtualisation depends on the host OS and does not share the host resources; Type-II VMM on the other hand sits alongside the host OS and shares the host's resources. Removing the dependency of operating systems on a system's physical state by installing multiple VMs on one VMM, full virtualisation provides an isolation of the guest operating systems running on the same hardware and protects against the operating system from being a single point of

failure. This abstraction of the hardware does not only allow multiple operating systems to coexist on the same hardware, but for one VMM to run on multiple physically networked systems concurrently. Utilisation of a VMM between the guest OS and the hardware, changes the one-to-one mapping of OSs to hardware. An example of a fully-virtualised system that has several physical systems running different operating systems on a single hardware using a VMM is illustrated in Figure 1-4 below.

Full Virtualisation System

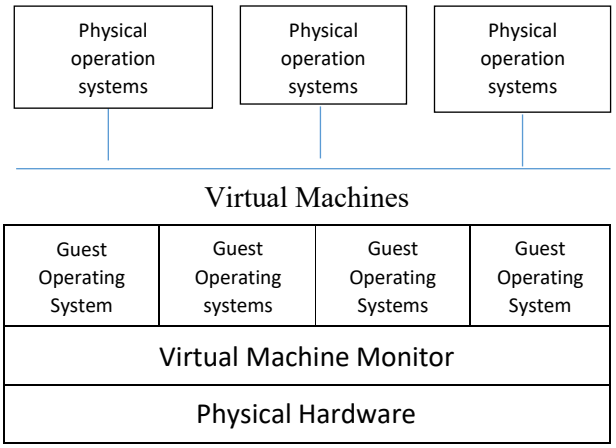


Figure 1-4 full system virtualisation

A virtual machine is a logical equivalent of a physical system. VMs installed on the same hardware are as logically separated as those machines in a physical environment and/or if networked as logically separate as different systems on the same network. The isolation of VMs however, has many implications for security. Virtualisation has become top priority for information technologies especially with the increasing demand for cloud computing security. The properties of virtualisation demand that a well implemented virtual machine monitor (VMM) should run innocuous instructions directly onto the CPU without interfering unless where necessary (Suresh and Kannan, 2014). The properties also state that VMMs should retain full control of resources to ensure that guest VMs do not have access or the ability to manipulate

resources without the VMM's explicit permission or authorization and lastly that VMMs should not be told apart from a physical machine of the same context (Suresh and Kannan, 2014). The use of multi-tenant infrastructures in cloud computing allows hardware or software resources to be logically proportioned between multiple tenants. Virtualisation ensures that guest operating systems in a virtualised cloud environment are completely isolated from one another. The VMM prevents the guest operating systems from seeing or directly accessing each other's computing resources that are hosted on the same platform.

The use of virtualisation technologies however, does not only bring security benefits to cloud computing, but also comes with its own security issues brought about by its strong properties as highlighted and discussed in the section that follows. These security risks can lead to unauthorised disclosure and/or alteration of sensitive data, breaches of privileges and controls. The need to trust a VMM, the transparent nature of an ideal VMM and the introspection capabilities that resource control requirements allow, all pose as security threats to the virtual system (Souppaya et al., 2011), since the VMM is always involved throughout the lifetime of the VM, each interaction between the VM and the guest is then a potential attack vector that can be exploited by the guest machine.

1.3 Security Risks of Virtualisation in Cloud Computing

The strong properties of virtualisation that offer cloud computing the security benefits discussed in the opening statements of the chapter can also pose threats to cloud computing.

The security risks or threats posed by virtualisation in cloud computing infrastructures are presented in Table 1-1 below.

Table 1-1 Virtualisation security threats and vulnerabilities

Security threats due to strong properties	**Security threats due to weak or poor implementation of Virtualisation properties**	**Security threats due to control, data and software flows**
VMM Insertion and Hyper-jacking	VMM Detection Breach	Control Channel • VM Cloning • APIs • Untrusted VMs
Introspection and intervention by VMM	Transparency Breaches	Data flows • Unauthorised communication between the hardware devices (open up side channels) • Hidden network channels
Transparent Virtualisation	VMM Compromise through Introspection and Intervention	
VM Cloning and Scaling	VMM Alteration	
Nonlinear VM Operation and Monotonicity issues	VMM Denial of Service (Local, Network and Host)	Guest operating system Denial of service
Software Decoupling from physical and hardware environment	Resource control breaches and Privilege Escalation • information leakage, • VM escape	Guest software compromise
	Malware	

1.3.1 Security Implications due to the Strong Properties of Virtualisation

- **Transparent Virtualisation**

Transparent virtualisation is when all three of Popek's properties (efficiency, resource control and equivalence) are met, providing a VMM that is ideal and undetectable to software running inside the VM. Reverse engineering in a transparent virtualisation becomes easier due to introspection capabilities because any encryption keys, security algorithms, low-level protection, intrusion detection and anti-debugging measures become easy to compromise. The combination of the basic trust model with transparent virtualisation means that a suspicious VMM can be undetectable, and is automatically trusted. Since the equivalence property of virtualisation requires VMMs to be equivalent to a physical system, it can be very difficult to know that software is running in a virtualised environment. Since a VMM has full control over system resources, it can observe or alter

data running inside the VM, which can cause potential security problems. (Ge et al., 2018, Zhao and Mannan, 2016). Near-transparency also makes VMM-based rootkits possible (Garfinkel et al., 2007, Garfinkel and Rosenblum, 2005, Gebhardt et al., 2008, Gebhardt et al., 2010, Gebhardt and Tomlinson, 2008).

- ***VMM Insertion and "Hyper jacking"***

Virtualising the OS of a physical system to a virtual system with very little downtime, either at boot or whilst the system is running is another strong property of virtualisation. The ability to migrate a physical operating system to a virtualised system can cause serious security risks, as the rootkits can be used to subvert an operating system completely. This form of attack is normally extremely hardware and VMM version specific (Im et al., 2017).

- **VMM Introspection and Intervention**

Introspection and intervention of the VM from the VMM is the ability of a VMM to take control of the resources. Although the VMM requires full control of resources to function, allowing it to observe and manipulate the applications and operations of the VM can result in more control than is necessary. This can cause security to enhance or weaken. Since the VMM runs at a high privilege level, it may observe and modify system aspects that may affect other software, as it remains hidden from the lower privileged software. The resource control property of virtualisation enables process introspection to allow the VMM to observe behaviour of processes within the VM. The VMM may also observe input or output (I/O) channels to, from, and within the VM. When combined with transparency, the VMM cannot only observe and alter any aspect of the VM but can automatically be trusted by the VMs and therefore remains undetectable. If the VMM is untrusted or compromised, the impacts of the security threats due to introspection and intervention are detrimental. The

most obvious approach to mitigating introspection and intervention threats is to attest the authenticity of the VMM and the underlying hardware.

- **Nonlinear VM Operation and Monotonicity Issues**

Since installed virtual machines can be cloned, and their states captured and restored, their execution does not follow a linear path through time, it is reversed and is subject to nonlinear operations of a similar kind. Lack of linearity is usually referred to as lack of monotonicity (Van Cleeff et al., 2009). A lack of monotonicity in applications causes security issues because snapshots, cloning, and restoration of VMs state can break linear operation of available applications, data and programs. Keeping data separate from the snapshotting process can cause security risks, that is, if the right data is not correctly stored or restored. Threats may also arise due to the rolling back of updates, improper configurations as well as from deactivated accounts, which leave the applications vulnerable or non-functional.

- **Software Decoupling from Physical and Hardware Environment**

A virtualised system abstracted from the hardware can be difficult to classify or define. A virtualised system can be cloned, and every single instance can be identified logically in order for it to be properly maintained or secured. Although threats from cloning are excluded in this type of security threat, problems still arise from the abstraction of the VM. If the location of a system is not known, it becomes difficult to maintain and/or manage making resource management and allocation a problem. The virtual network may also not be a true representation of the actual logical location of the physical system network. Although virtualisation allows software to be run independently from the hardware there is still need for physical location control.

1.3.2 Security Implications from Weak Implementation of Virtualisation

- **VMM Detection Breaches**

Detection of a VMM causes problems for some uses of VMMs such as the analysis of malware if it suspects it is being run in a virtualised environment causing it to behave differently, being unable to detect a VMM on the other hand can also cause problems in other cases.

- **Transparency Breaches**

Virtualisation transparency is breached when any of the properties are breached. Transparency breaches lead to unauthorised disclosure of important information that can be used to deduce the presence of a VMM. Transparency breaches cause instability in the virtual environment that can lead to software failure, broken device states and timing problems which can lead to the system making false assumptions about the environment in which it is being executed. It is misleading for example to extrapolate seek times of a physical disk to those of a virtualised disk as not everything that optimises disk layout for speed, like Database Management System (DBMS) for example, will be seen as a layout that does not correspond with the physical layout in a virtualised system. The VMM being the most central part of a virtualised system, if an aspect of it is compromised or breached, then the entire system and everything running on it, is potentially at risk. (Carroll et al., 2011, Ormandy, 2007).

- **Resource Control Breaches and Privilege Escalation**

Resource control breaches are considered among the most serious attacks that affect confidentiality of data in information security. In virtualised systems, these threats include information leakage and privilege escalation. VM information leakage comes in two main types: leakage of information within the VMs and leakage of information out of the VMs.

A virtual machine may leak information about its operations and/or its resource usage to other VMs on the same network through side channels that can be software or hardware based attacks. Leakage of operational data into VMs includes details of the host or the state of the resources of other virtual machines on the virtual network. The type of leakage of information from the hardware is known as cache-based attacks which may enable the VM to detect things that it ought not, such as resource status of the physical machine like the CPU and memory usage, network configurations and details about the operations of other VMs (Hong et al., 2017, Ren et al., 2016, Simma et al., 2015, Subashini and Kavitha, 2011) provide and in-depth discussion on information flow attacks, whereas others (Subashini and Kavitha, 2011, Zhang et al., 2012) explain more on cache based issues in the shared computer in the cloud. The other threats caused by the isolation property of virtualisation, such as VM escape, cause escalation of the application code which breaches the isolation between the host and virtualised environment causing the VM to run codes on the host machine, without the explicit control of the VMM (Criscione, 2010, Gebhardt et al., 2008, Ormandy, 2007).

1.3.3 Security Threats from Control, Data and Software Flows

- **Security Threats from Control Channels**

Control channels in VMMs are commonly used for administrative purposes and VMM Application Programming Interfaces (APIs) facilitate administration of the VMM and VMs that are running on them. This includes activities such as changing the operational state of VMs including shutting them down, modifying existing VM settings, cloning and creating new VMs, and executing commands on the guest OS. Control channels threats consist of both unauthorised access and denial-of-service attacks. The threat presented by control channels is exacerbated as some VMMs contain undocumented hidden control channels that function through undocumented device and CPU instructions (Ormandy, 2007).

- **Security Threats from Data flows**

Data flows for both software and hardware, are primarily vertical functions between different layers with the exception of network channels, which are connected to different components concurrently. Input/output flows and devices in modern systems can be very complex and securing them is a trivial task especially when system performance is a problem (Karger and Safford, 2008) and when software channels include those used by physical devices, virtualised devices as well as the VMM APIs. Physical devices normally present security threats, as they open up side channels that do not go through the VMM or use the shared resources of the VMM. (Gebhardt et al., 2010, Khan et al., 2016) discuss the potential of VMM resource control to be circumvented through the use of DMA technologies to access memory locations and others (Dowty and Sugerman, 2009) discuss establishing secure Graphics Processing Unit (GPU) isolation in the virtualised environment whereas (Raj and Schwan, 2007) and (Sañudo et al., 2018) discuss how pure isolation of hypervisors, where each guest uses a separate device and/or drivers, can differ from sharing the hypervisors where guests use resources on the same device in security and performance that they offer. Network channels present similar security issues as those presented by control channels and shared devices although network channels present a significantly higher risk as they can be accessed remotely and may be connected to every virtualisation component.

- **Security Threats from Non-VMM Software**

Security threats in virtualised systems as observed in the previous sections do not only come directly from the strong properties of virtualisation, but also from the way in which they are used and implemented for example decoupling of software from the physical system presents a larger software and hardware attack surface. A virtualised system

inherently has more software running on its platform than a physical or non-virtualised system. Attack vectors in the implemented virtual machine monitor includes the VMM software, instances of the operating systems and other software inside various virtual machines. This makes the overall attack surface of the software higher because the software is most likely to have many internal data flows.

1.4 Security Risks and Threats aimed at different components of Virtualisation

The most obvious way to attack a virtualized system is to gain access to the hypervisor. Since the hypervisor has the ability to control, monitor and interfere with all the activities of the virtual machines running in the virtual environment, a compromise or improper configuration of the hypervisor can lead to compromise of all hosted virtual machines as well as all physical systems resources like the hard drives and the servers. The hypervisor offers a singular point of access into the virtual environment and is therefore a singular point of failure (Suresh and Kannan, 2014).

The security threats highlighted in Table 1-1 are aimed either at the hypervisor (VMM) or at the virtual machines in the virtual environment.

Threats to the Hypervisor

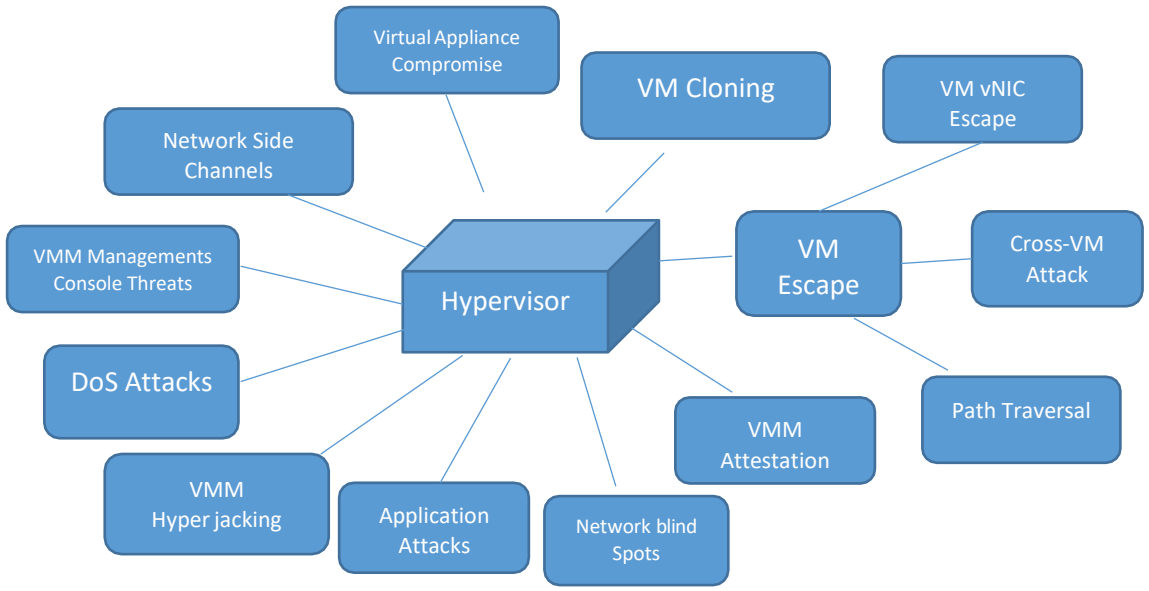


Figure 1-5 Threats to Hypervisor (VMM)

Although the attacks depicted in Figure 1-5 above are aimed at the hypervisor, these attacks still pose great threats to the virtual machines because once the VMM is compromised the VMs will automatically become vulnerable to the same attacks. There are a number of attacks aimed specifically at the VMs as illustrated in Figure 1-6 below. Attacks like DoS attacks can affect both the VMM and VM.

Threats to the Virtual Machine

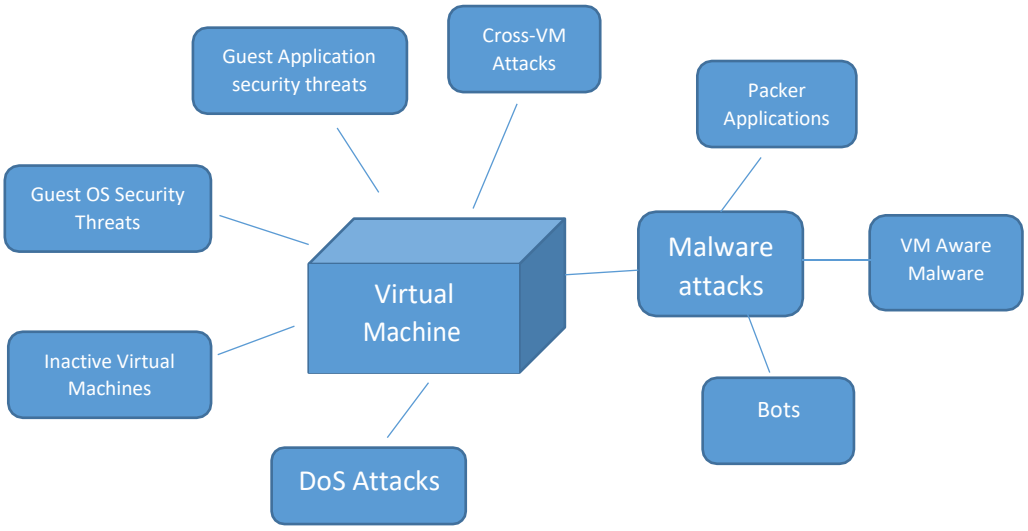


Figure 1-6 Threats to the Virtual Machines

Attacks on the hypervisor can be either through the host operating system or through a guest operating system as illustrated in the section below.

Attacks on the Hypervisor through the Guest Operating System

Attacks on the hypervisor through a guest operating system can be used to gain unauthorized access to other virtual machines or the hypervisor. These attacks are also known as VM escapes or jailbreak attacks as the attacker "escapes" the confinement of the virtual machines into layers that are unknown to the virtual machine. This is the most plausible attack on the hypervisor; because the attacker can only compromise a virtual machine remotely as the underlying host operating system is invisible (Chung et al., 2013, Luo et al., 2011, McDaniel and Nance, 2013, Neal, 2013, Saroha, 2014). Since many virtual machines share the same physical resources, if the attacker can find how virtual machines' virtual resources map to the physical resources, then attacks can be made directly onto the physical resources. Modifying the virtual memory in a way that exploits how the physical resources are mapped to each virtual machine, the

attacker can affect all the virtual machines, the hypervisor, and potentially other programs on that machine (Chung et al., 2013, Luo et al., 2011, McDaniel and Nance, 2013, Neal, 2013). This type of attack is illustrated in figure 1-7 below:

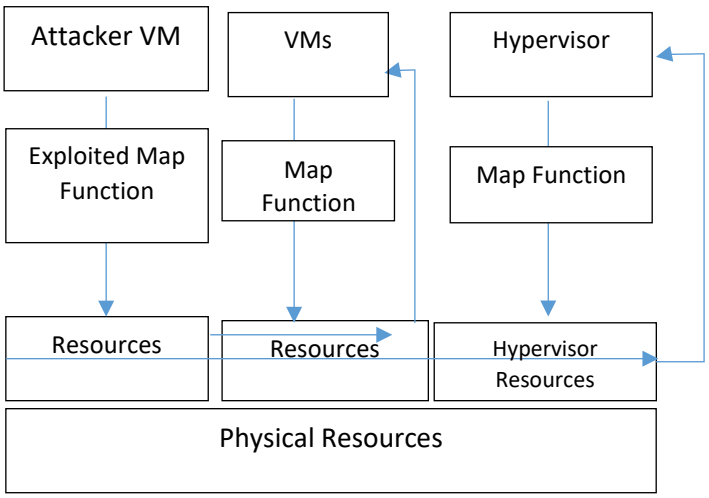


Figure 1-7 Attacks on the hypervisor through the guest operating system

Figure 1-7 shows that the attacker exploits the vulnerable VM, which provides access to the system's resources through the exploited map function and then moves through the resources of the physical system to gain access to the other VMs as well as the hypervisor.

Attacks on the Hypervisor through the Host Operating System

The vulnerabilities and security holes in most operating systems can be used to gain control of the host operating system. Since the hypervisor is simply a layer running on top of the host operating system, once the attacker has gained control of the host operating system, the hypervisor is essentially compromised. The administrative privileges of the hypervisor will enable the attacker to perform malicious activities on any of the virtual machines hosted by the hypervisor as illustrated in the figure 1-8 below:

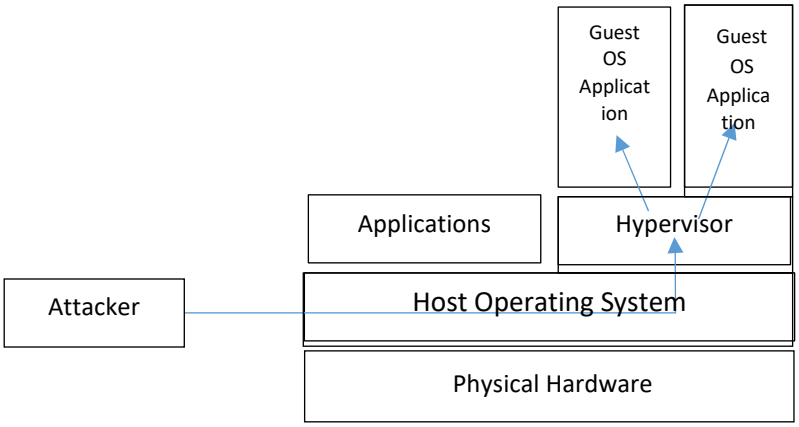


Figure 1-8 Attacks on the hypervisor through the host operating system

These security issues have led to the design and implementation of various security defence mechanisms aimed specifically at combating attacks in various components of virtualisation such as but not limited to those highlighted in Table1-2 below.

Table 1-2 Attacks on Virtualisation Components and Available Defence Mechanism

Attack Vector/Component	Attacks	Defence Mechanism
Hypervisor-based attacks	VM Escape attacks Hyper-Jacking VM Sprawl Rootkits Inter-VM communications	Designing Secure Hypervisors (Dildar et al., 2017, Vasudevan et al., 2013) Protecting Hypervisor integrity and Reducing hypervisor attack surface (Szefer et al., 2011) Intrusion Detection System (Azmandian et al., 2011, Taj et al., 2020, Kumara and Jaidhar, 2015) Proper Configuration of interactions between VM and Host (Rueda et al., 2008)
VM-based attacks	Inside-VM attacks Malware Denial of Service (DoS) attacks Cross-VM side-channel attacks Idle VM attacks Software vulnerabilities VM Footprint	VM Introspection (Lee and Yu, 2014a) Virtual Machine-based Intrusion Detection (Azmandian et al., 2011, Taj et al., 2020) Co-Residency Detection in Cloud via Side Channel Analysis (Zhang et al., 2011) A Security-Aware Scheduler for Virtual machines on IaaS Clouds (Khan et al., 2016) Using anti-virus, anti-spyware programs in virtual machines to detect suspicious activity

Despite the many solutions for virtualisation security, challenges like monitoring, visibility and infrastructure still pose security threats to virtualised systems or environments (Brooks et al., 2012, Sahoo et al., 2010) :

- **Monitoring** is the ability for data centres and cloud to log trustworthy data on activities in virtual machines or the hosts (Brooks et al., 2012, Sahoo et al., 2010). Abstraction, being one of the main benefits of virtualisation, prevents important information that might help to determine potential threats, from being visible as it provides insufficient data to see if a threat has occurred.
- **Visibility** is how much intrusion detection and prevention systems can see into a virtualised network (Brohi et al., 2012, Lee and Yu, 2014b, Padhy et al., 2011, Pearce et al., 2013) . This normally works hand in hand with monitoring, in that, if it is difficult to monitor activities of the virtualised environment, there will be no detection of malicious activities and prevention therefore, will not be necessary. This characteristic however, also causes the visibility on the host's operating systems and the virtual networks to lower, making it harder to detect infected Virtual Machines and to prevent malicious intrusions detection. Again, there currently lacks a balanced solution between visibility and inherent security for virtualisation**.**
- **Infrastructure** is the way virtualisation is set up in a data centre or cloud (Brohi et al., 2012, Lee and Yu, 2014b, Padhy et al., 2011, Pearce et al., 2013). This is a challenge in virtualisation because of the incompatibility and misconfigurations of virtual machines and hosts between various cloud platforms or datacentres. These significant security risks that result from monitoring, communication and modifications of the guest operating system and unique to virtualisation in this context, refer to the security risks that might compromise confidentiality, integrity and availability of resources in cloud computing.

A VMM, as the properties state in the first paragraph of the chapter, only intercepts sensitive instructions, which could interfere with the operation of the VMM itself. All the other instructions execute directly onto the hardware where possible without the interruption or interference of the VMM. To understand the threats posed by virtualisation in cloud, designing and implementing a threat model is key in virtualisation as it is in the design and implementation of any security for information systems and applications. The threat model such as that illustrated in Figure 1-6, aims at depicting the security or threat model designed for virtualisation security, with an emphasis on sensitive information and the possible definitions of the security attacks. A generalised threat model that attempts to address all types of attacks, penetrations, and leakages concerning every possible configuration of virtual platforms, utilizing any type of supporting hardware and/or software, and protecting every type of asset is very likely to be too general to be of significant value. It is therefore important to develop a model that is specific to the operational needs and scenarios of the security evaluation process available.

Figure 1-9 shows an example of a generic threat model process:

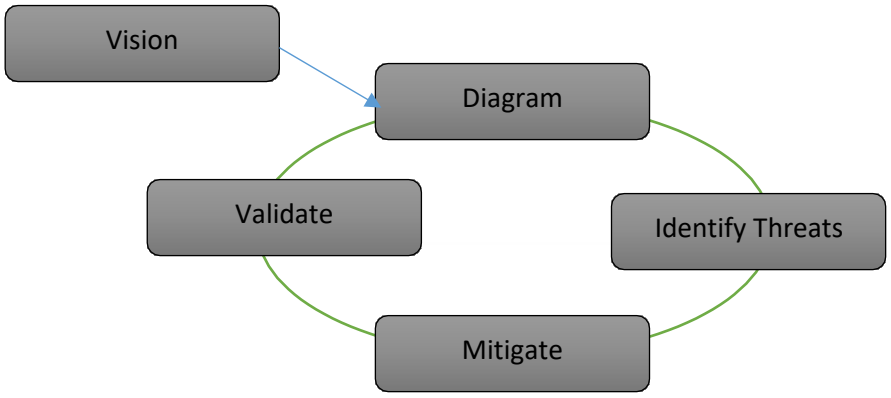


Figure 1-9 Security Threat Modelling Process(Amini and Jamil, 2018)

The threat model process in Figure 1-9 above helps to define the necessary threat model that is tailored to a particular scenario which in this case is virtualisation. A successful threat model follows a systematic approach that will:

- Define the security requirements and risks of the system under development
- Communicate the security design aspects of both the VMM and VM system
- Analyse various system components for potential security threats using a proven validation methodology
- Discover and mitigate issues that could put VM infrastructures at risk, as well as prioritize and plan efforts to address significant threats that arise from system backups, failover, misallocation of resources and malfunctions caused by malicious software used to Identify the threats
- Manage and mitigate available security threats

The strong properties of virtualisation discussed earlier in the chapter can either enhance or compromise the security of virtualised environments. The benefits that virtualisation offers to cloud computing can be very dangerous to the virtualised environment if the VMM used is not trusted. This usually occurs because of the VMM trust model, the transparent nature of an ideal VMM, and the introspection capabilities that the resource control requirement allows. The need to trust a VMM in a virtualised environment is a big vulnerability since the VMM is a single point of failure because a malicious, compromised, or even problematic VMM is most likely to observe and/or intervene with the VM's processes or operations. Secure virtualisation relies upon the authenticity and integrity of the VMM, and in most cases on the security and the type of the underlying hardware (Lombardi and Di Pietro, 2011, Yang and Fung, 2016). The use of a trust model can offer attestation of the integrity and state of the system components that ensures that applications and operating systems can only run in an environment that has an attestation chain to a trusted root. Attestation in virtualisation offers a degree of verification,

authenticity and integrity of the VMM. Attestation methods have been used as a means to assess the security of virtualisation, in this case the VMM, the hardware, and the guest OS, and all the applications. Attestation can be done locally via internal I/O channels, or remotely through a network (Pearce et al., 2013). Attestation of the integrity, authenticity, and state of guest software is important for applications and data security not only for guest OSs, but also for the applications. This is due to the fact that security requirements can depend not only on the integrity of core system components or the application itself, but also on the presence and state of additional software inside the guest OS. This includes the status of security measures such as logging, antivirus, or intrusion detection systems available. A VM is a container with various software and data components such as VM settings, a virtual disk image, and captured VM states (Wojtczuk and Rutkowska, 2009, Wojtczuk and Rutkowska, 2011). Attestation alone does not remove the security threats surrounding virtualisation because even an attested component is vulnerable to attacks or even used as an attack vector.

1.5 Thesis Statement

This thesis explores the possibility of implementing predictive analytics into information systems by implementing it as a security management tool in virtualised information systems. To do this, we explore various types of virtualisation, virtualisation architectures as well as the security threats and attacks which virtualisation is vulnerable. The thesis also reviews existing solutions in relation to the topic in question by exploring the various techniques used to help address the various security issues that virtualisation faces. To help understand and answer the question of the project, an in-depth exploration of what predictive analytics is, how it can be adopted into empirical information systems and the possibility of using predictive analytics as a security management tool for virtualised systems is carried out. Since the main aim of the project is to use empirical predictive analytics in information systems such as virtualised systems in this case as opposed to techniques that use explanatory or exploratory data analytics

only, a further discussion on what predictive analytics is and how it differs from explanatory analytics is covered. This gives an in-depth example of how explanatory analytics works in comparison to predictive analytics as well as how it is used to feed into predictive analytics as one of PA's initial processes. The thesis presents experimental results of the attack simulation process that is used to generate the data required for exploratory data analytics and thus predictive analytics and modelling. The steps taken to build, validate, evaluate and test predictive analytics using various algorithms is presented and the results that prove the use of predictive analytics modelling in information systems and virtualisation in particular are shown and explained. The thesis demonstrates that empirical predictive analytics can be adopted as a security management tool in virtualised systems, illustrated through the partial application of the built predictive models on new data of the attack simulation in real time.

1.6 Research Aim and Objectives

As security continues to be the focus in virtualised environments, the demand for adequate solutions has also increased. A number of contributions towards the security issues of virtualisation have been made but none so far has provided the so much needed solution of dealing with attacks in real to near real time using predictive analytics. The use of predictive analytics has become popular in other disciplines such as medicine, economics, and e-commerce to mention but a few, and its impact has made significant difference to the success of various components of these disciplines. The adoption of predictive analytics into empirical information technologies such as cloud computing which uses multi-tenant infrastructures is very important especially with the many benefits predictive analytics offers.

The main aim of this research is to design and implement a predictive analytics modelling framework that will enable the modelling and use of predictive analytics as a security management tool within a fully virtualised information system environment (Kapasa et al., 2015). The achievement of the use of empirical predictive analytics as a security management

tool in virtualised systems will illustrate and potentially show that predictive analytics can be used in information systems such as cloud computing as successfully as other disciplines that are using predictive analytics to make informed decisions.

To achieve and successfully implement empirical predictive analytics in a fully virtualised system, a list of objectives established for the implementation of predictive analytics as a security management solution for virtualised environment are to. :

1. Design a predictive analytics modelling framework that will be used to design, implement and deploy the security management predictive model in question
2. Explore and evaluate the concept of predictive analytics, how it works and how it is being used in other disciplines as well as how it can be adopted as a cybersecurity tool for cloud multi-tenant infrastructures.
3. Analyse and assess the risks associated with virtualised systems together with a thorough analysis and comparison of the existing controls that are currently in place to mitigate these risks.
4. Identify and evaluate the various types of tools and techniques used in predictive analytics as opposed to exploratory analytics and how these tools and techniques can be combined to help build, implement and deploy the predictive analytics model as a security tool in a virtualised environment.
5. Set up an experimental environment for the simulation of attacks for data collection, explanatory data analysis and the designing of the predictive analytics model in question and then use the results of the attack simulation to define the predictive goals of the project.
6. Build the predictive analytics model and then test and validate the built predictive model to measure the predictive accuracy of the built model using the sample

assessment of the cross-validation method and then modify the predictive model accordingly until the desired results are reached.

7. Implement and deploy the built predictive model into the setup cloud computing system.

1.7 Contribution

The incorporation of virtualisation into cloud computing is very useful especially where security and utilisation of computing resources are concerned. The evolving technology together with the new characteristics of virtualisation has also brought about new security threats and risks. It is very important for security professionals and researchers to continue finding solutions to the never-ending security threats and risks affecting virtualisation. The speed at which these security threats and risks need to be mitigated for both the cloud providers and the clients, is very important. The literature reviewed on the current or existing solutions still does not answer the questions of real time analysis and mitigation of the security threats and risks. Although many researchers start with predictive analytics in mind, most of them end up with explanatory solutions that use data mining techniques to analyse security threats and risks after the fact. These solutions are usually based on activities that have already happened. The speed of identifying potential threats and risks before they happen, as well as finding solutions to mitigate these risks in real time, is very important.

In this research we have designed and implemented a predictive analytics modelling framework (PAMF) as a novel contribution to the building and implementation of a novel predictive analytics model for the management of security in virtualised information systems. These two novel contributions add to the existing security frameworks and security used in implementing security solutions for virtualisation security identified and implemented by many researchers and security professionals. Successful achievement of the project also contributes to the theory of using predictive analytics in empirical information systems suggested by (Shmueli and

Koppius, 2011) through the successful design and implementation of the same predictive analytics model into cloud computing.

Successful achievement and implementation of the stated objectives:

- Demonstrates that the designed PAMF was followed to successfully build, validate, implement and deploy the predictive analytics model in question.
- Demonstrates that security threats such as DoS attacks are analysed and mitigated in real to near real time using predictive analytics as opposed to those using explanatory data analytics such based on data mining techniques.
- Highlights that the built model although tested on DoS attacks can used widely on various attacks in real time to identify patterns in the data that might lead to possible attacks unlike the Home Alone framework, which concentrates only on the risks of isolation.
- Highlights the various risks of using virtualisation and then test the controls available for mitigating these risks by using the laid down capabilities of the Home Alone framework to incorporate better classifiers to improve on how the risk of isolation is being mitigated. It then extends this to all the other aspects of virtualisation using predictive analytics rather than mere statistical analytics which Home alone is currently using.
- Contributes to Shmueli et al's idea on how empirical predictive analytics is designed and implemented in information systems such as cloud systems in this case and then demonstrates how this can be adopted in virtualised systems as a risk management strategy for virtualisation security. The aim of the implemented work is to illustrate the accuracy level in predicting new data especially in virtualised systems.
- Contributes to the existing theories on the use of predictive analytics and possibly improves on the existing theories that are currently based on explanatory predictive

analytics and using exploratory statistical models to develop new measures by comparing different operationalization of constructs.

Illustrates the possibilities of using predictive analytics as a risk management strategy in real time as compared to those predictions based on explanatory predictions done on static data. A number of predictions use data mining and statistical algorithms only to analyse risks or to see if an attack has occurred so that a solution to remove the risk can be found. Mainly the designed frameworks use detection methods to eradicate the problems. Although many attempts (Low et al., 2012, Mishra and Sahoo, 2011, Zhang et al., 2010) have been made in trying to mitigate risks by using predictive analytics, most of them normally end up as just explanatory. The what-if analysis framework for example started with predictive analytics goals in sight but ended up with a system in which, instead of using real time predictions, static data was collected and then analysed. This research goes beyond explanatory predictions or data mining techniques in that, predictions are run in real time and risks are identified and mitigated before they actually happen

1.8 Summary

Virtualisation is described as the encapsulation of the VMM that surrounds the operating system and provides the same processes as those of the physical system. The chapter besides providing the definition of what virtualisation is, has also given an overview of the different architectural types of virtualisation commonly known as Type I or bare metal which is installed directly onto the hardware and Type II also referred to as hosted which is installed right on top of the host operating system of the physical hardware. Section 2.2 of the chapter demonstrates how virtualisation can be applied to various components of the computer system such as the server, applications, desktop and the network. As virtualisation does not only bring security benefits as highlighted in the introductory part of chapter it also brings about security issues caused by the strong properties virtualisation offers to the cloud. The security issues identified

in section 2.3 are caused by transparency, VMM insertion, and hyper-jacking, introspection and intervention of nonlinear VM operations, monotonicity and decoupling of both the hardware and the software. Since attacks on virtualised systems come through various vectors or platforms, the chapter also highlights the many ways different components of virtualisation such as the hypervisor, the guest operating system or the applications can be attacked and the channels used for different types of attacks. The security threat model illustrated in figure 2.5 in section 2.3 defines the security requirements of the system, proposes the security design for both the VMM and the VM, analyses potential security risks of various components, highlights security issues relating to the VM infrastructure and provides solutions on how to mitigate the identified threats. The chapter concludes by discussing the various threats that arise from control, data and software together with the security vulnerabilities and threats of both the VM and VMM.

As security remains the main concern for virtualised systems in cloud computing, it is imperative for security professionals and researchers to come up with effective security solutions. The following chapter reviews existing security solutions for virtualisation.

2. Existing Security Solutions for Virtualisation in Cloud and Related Work

2.1 Introduction

Solutions to address the security issues unique to virtualisation have been suggested, designed and implemented by many researchers and security professionals. Analysis of malicious packets and process logs has been a problem in information systems such as virtualised systems as attackers are continuously changing their methods and tactics of launching attacks. The use of big data analytics to detect and mitigate attacks in information systems has become key in computer security. Taking advantage of the cloud-scale techniques and storage mechanisms, big data can be used to collect and analyse large volumes of data processes and network flows. The scale and granularity of using big data can result in accuracy in stopping attacks more than in legacy data-limited defence mechanisms. Although big data analytics offers a sense of security in cloud computing, the need to stay ahead of attacks especially in multi-tenant infrastructures such as virtualisation security is key. Virtualisation as discussed in chapter 2 increases the efficiency and elasticity of modern computing infrastructures. The need to stay a step ahead of the threats and attackers has led to the design and implementation of the predictive model as a security management tool for virtualised systems using predictive analytics. Researchers such as (Won et al., 2013, Almutairy et al., 2019, Armstrong et al., 2008) have come up with solutions on how to manage risks in information systems using different types of techniques. Other existing solutions for virtualisation security issues are illustrated in table 2-1.

Table 2-1 Virtualisation Security risks proposed solutions and the impact to in Cloud systems

Virtualisation Security issue	Impact on Cloud	Counter Measure/Possible Solution	Focus of the Work
Data Leakage and Data Remanence	Exposure of Confidential Data	Hypervisor-based virtualisation, VM Security and Reliability monitoring (Payne et al., 2008, Sabahi, 2012, Zou et al., 2013, Mishra et al., 2017a)	Virtualisation technology to Reduce the work load and Distributed Security System
Breaches due to Software Defects	Compromises guest VMs	Virtualisation introspection system (Lee and Yu, 2014b)	To detect and prevent malicious attacks and to protect both the VMs and the Host
Malicious Network traffic	Spoofing Attacks	A robust and learning based security approach Network Packet Detection (MNPD) (Guizani and Ghafoor, 2020) (Mishra et al., 2017b)	To detect network intrusions in cloud
Hypervisor attack	Allows the attacker to gain access and authorisation over the hypervisor allowing them to modify or remove systems files	Virtual Machine and Hypervisor Intrusion Detection System (Kumara and Jaidhar, 2015)	To determine competent approaches for defending the hypervisor attacks in cloud computing
VM escape	Attacker can gain control of	Penetration testing	To test the security of the virtualised environment
VM Sprawl, Idle VMs Resource Control, Unauthorised access and Configuration issues	Compromises confidentiality, integrity and availability of resources. Configuration problems can proliferate and consume resources causing performance problems	Effective controls Data Encryption Develop policies to handle VM imaging and snapshots	Identification and management of security risks specific to virtualisation technologies

A number of virtualisation security solutions using predictive analytics have been proposed as explained in the following section, which compares and evaluates some of these existing approaches and solutions to virtualisation security in cloud computing.

2.2 Related Work

2.2.1 Home Alone: Co-Residence in the Cloud via Side-Channel Analysis

Zhang et al for example have designed and implemented a framework called Home Alone: Co-Residence in the Cloud via Side-Channel Analysis. The purpose of the framework is to be able to verify remotely that the tenants' virtual machines are physically isolated and that a tenant has exclusive use of the physical machine. The frameworks idea is to invert the usual application of side channels and use it as a defensive detection tool that analyses cache usage during periods where virtual machines coordinate, in order to avoid portions of the cache of a tenant using Home Alone, that can detect the activities of a co-resident foe (Won et al., 2013). The idea of the Home Alone is to allow the tenant with the Home Alone to coordinate its VMs in order to silence their activities in a selected cache region for a period. The tenant will then measure the cache usage during the silenced time to check for unexpected activities if any. These activities if found will suggest the presence of a malicious VM. The Home Alone approach however requires a more complicated solution than simple silencing of VMs and listening for malicious activities of the cache because the cache in a virtualised environment is never completely quiet and measuring of these activities can be done with a lot of errors due to available noise. The other challenge of the Home Alone framework is that there is need for the construction of effective classifiers that can distinguish normal cache activities from malicious activities.

Even though Zhang et al have claimed to manage the full implementation of the Home Alone framework, they have also indicated the need for better classifiers for cache timing behaviours to help improve their framework (Won et al., 2013).

2.2.2 Network Intrusion Detection and Countermeasures Election (NICE) framework.

Chung et al have designed and implemented a Network Intrusion Detection and Countermeasures Election (NICE) framework. The framework is aimed at identifying malicious VMs on the network or VMs that are vulnerable and prone to attack in virtual network systems by establishing a defence-in-depth intrusion detection framework which incorporates attack graph analytical procedures into detection processing (Zou et al., 2013). NICE is intended to help detect and mitigate collaborative attacks in cloud virtual networking environment (Chung et al., 2013). NICE however only investigates the network IDS approach to counter idle virtual machines explorative attacks, and has indicated the need to improve accuracy by encouraging host-based IDS solutions (Zou et al., 2013).

2.2.3 Distributed Graph Lab

Low et al on the other hand have implemented a different framework for machine learning and data mining in the cloud model known as the Distributed Graph Lab. This is an extension of the MapReduce applications (Elkan, 2013, Low et al., 2012). MapReduce is one of the most important programming paradigms that facilitates and provides parallelisation, distribution and fault tolerance within the framework as well as facilitating the cloud computing environment to process big data (Elkan, 2013, Low et al., 2012). The distributed Graph Lab or machine learning data mining (MLDM) framework as named by Low et al is a graph structured, asynchronous and iterative computational framework aimed at modelling data dependences and updating large sets of parameters (Elkan, 2013, Low et al., 2012). Low et al like the others suggest that the Distributed Graph Labs abstraction and runtime need to be extended in order to support the dynamically evolving graphs and external storage in graph databases, which will enable Distributed Graph Lab to continuously store and process time evolving data (Elkan, 2013, Low et al., 2012).

The explored existing solutions above have indicated both time and big data analytics, which the security management tool for virtualised systems proposes, as restrictions or areas of improvement to their systems. With the ever evolving, large volumes of data that need to be processed, analysed and reported there is need for real time processing and analysis of big data to help address the security issues of virtualisation. As indicated in the virtualisation security solutions above, the existing solutions use data mining techniques or statistical algorithms to monitor, analyse and mitigate risks. Data mining and statistical algorithms however, have been surpassed in demand by predictive analytics (Low et al., 2012).

Researchers such as (Li et al., 2012) have identified the ideas of how risks in information systems, specifically virtual systems, can be identified, analysed and mitigated with the use of predictive analytics. Yu Si et al for example, investigate the threats that break isolation and propose a virtual machine co-resident detection scheme via cache-based side channel attacks to determine the location of the specified virtual machine (Luo et al., 2011). Side channel attacks are malicious activities that malicious users can use to steal private information from other users by analysing responses of third party sharing resources in the cloud (Brooks et al., 2012). Singh et al has also suggested an analytical modelling tool that tries to address what-if analysis in complex cloud computing applications. This is a workload-based what-if system that takes hypothetical changes in the applications workload or environment to estimate its impact on performance (Singh et al., 2013).

Both Yu Si et al and Singh et al are still using data mining techniques to address the security issue of virtualisation. These solutions or approaches are still handling risks after the fact and are not trying to address the issue of handling big data analytics in real time to predict the security threats unique to virtualisation as illustrated in this project.

Although big data analytics uses machine-learning tools and techniques such as data mining to determine whether or not an attack has occurred there is still need to stay ahead of the threats and attacks. An increase in the adoption and use of predictive analytics has also led to high demand for big data analytics. Big data analytics is the collection and analysis of large and complex datasets that the traditional data management tools are failing to process. Predictive analytics together with big data analytics has improved the way data is collected, processed and analysed and has led to finding solutions to the various questions that have stayed unanswered. The amount of data available, the range of the data types and the speed at which the data arrives has also increased.

The diagram below shows how predictive analytics and big data work together.

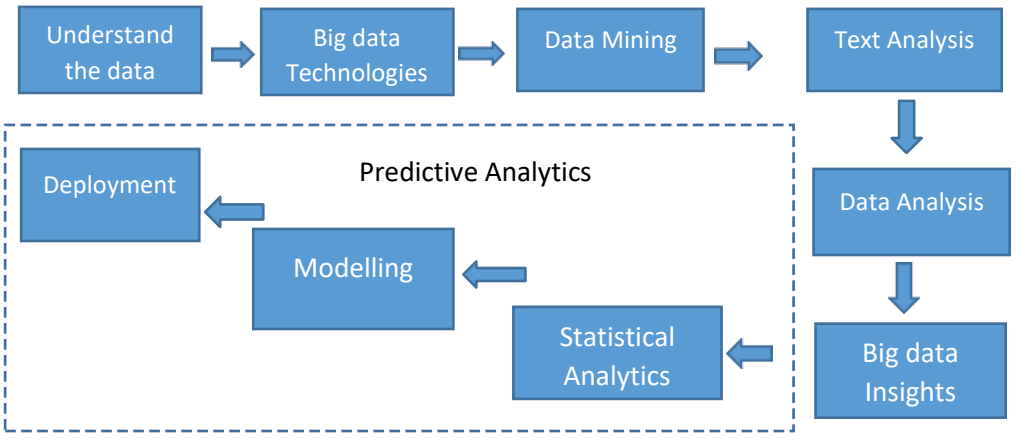


Figure 2-1 Predictive Analytics and Big Data Analytics (Gandomi and Haider, 2015)

In order for big data predictive analytics to be effective, an iterative process should be followed as indicated in Figure 2-1 above. Predictive analytics and data mining are sometimes viewed as the same even though in reality they are separate interacting processes. Predictive analytics uses confirmed relationships between explanatory and criterion variables from past occurrences to predict future outcomes whereas data mining is an automated process that uses sophisticated mathematical algorithms to search through enormous sets of data.

2.3 Predictive Analytics

Predictive analytics is the practice or method of extracting information from existing data sets in order to determine patterns and predict future outcomes and trends. This is to say that predictive analytics will not tell you what is going to happen in future, but it forecasts on what might happen in the future with an acceptable level of reliability, and includes the what-if situations and risk assessment. Predictive analytics is used to forecast the future probabilities.

Predictive analytics is also described as the art and science of using data to help uncover hidden patterns and relationships in the data that can help predict what will happen in the future as well as providing valuable actionable insights for confident decision-making (Fuentes, 2018). Predictive analytics has grown from building a predictive model that can influence a decision to incorporating these models into day-to-day operations (Won et al., 2013, Gulati, 2015). In the past years, theory was used to develop hypotheses that where then tested; nowadays data is used to find relationships that can be used to develop a hypothesis which then leads to testing the developed hypothesis, building a model and then validating the built model. Predictive analytics is the broad term that describes a variety of statistical and analytical techniques that are used to develop models that predict future events and behaviours (Eckerson, 2007, Gulati, 2015, Shmueli and Koppius, 2011, Fuentes, 2018). These techniques are usually divided into three different categories: Predictive models, Descriptive models and Decision models.

Predictive models look for certain relationships and patterns between explanatory variables and dependant variables that usually lead to certain behaviour. If the explanatory variables can be determined then the outcome of the dependent variables can be predicted. Predictive modelling is a process that uses data mining and probability algorithms to forecast outcomes. Each model is made up of a number of predictors, which are variables that are likely to influence future results. The model may employ a simple linear equation, or it may be a complex neural network, mapped out by sophisticated software.

Descriptive models aim at finding clusters of data elements with similar characteristics and then create segments, which are often used for classification of variables and identifications of relationships. Descriptive modelling is a mathematical process that describes real-world events and the relationships between factors responsible for them. It is more of the knowledge of what happened in the past that can be used in the here and now to predict what could happen in future.

Decision models use optimisation techniques to find the most certain outcome for a specific predictive decision.

Predictive analytics is mainly used for

1. New theory development - This is very important because the fast changing environments, the large amounts of data sets made up of various types (like text and digital) and the high speed at which this data is being received all bring about the need for effective new theory development. Predictive analytics helps in identifying complex relationships and patterns that are hard to draw hypotheses from, especially the ones with theories that exclude newly measurable concepts by detecting new patterns and behaviour that can help uncover new casual mechanisms and lead to new theory development.
2. Construct operationalization – This is a more specific aspect of predictive analytics where new theory development works hand in hand with the development of new measures. It compares the different operational constraints of various models using assessment methods to provide construct validation of the compared models.
3. Comparing existing theories – Predictive analytics uses explanatory models to compare leading theories, which are normally difficult to compare statistically for predictive accuracy.

4. Improving existing explanatory models – Predictive analytics improves existing explanatory theories by capturing complex patterns and relationships of the models and turning them into predictive values.
5. Assessing predictability of existing models – Predictive analytics can be used to quantify the predictability level of measurable phenomena by creating a benchmark for predictive accuracy, which may lead to development of new measures, data collection, new empirical approaches and evaluation of the difference in predictive power of existing models.

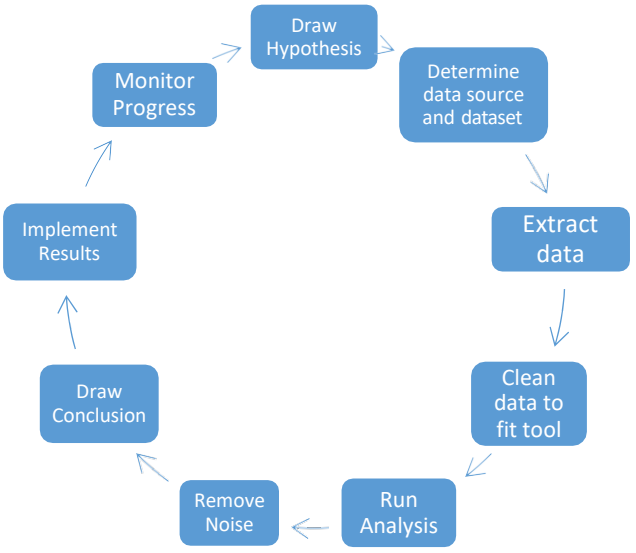


Figure 2-2 Predictive Analytics Process (Larose, 2015)

Figure 3-1 above presents an iterative predictive analytics process that starts with a hypothesis or an assumption to test. This leads to the identification and extraction of data from the sources and using a predictive analytics tool, data is processed, a conclusion is drawn, the results are implemented and the model is monitored for performance accuracy.

Finding the best predictive modelling technique and algorithm to leverage data for insightful decisions is key to finding the best predictive model for the management of security issues in virtualised systems implemented in this project. Predictive modelling in machine learning is a

process that uses data and statistics to predict an outcome using data models. Predictive analytics refers to data science and involves making predictions about future events by using different algorithms like linear regression, an algorithm that analyses past data to make predictions about the future.

The main advantage of predictive modelling is that it gives accurate insights into any question and allows users to create forecasts. It is important to have an insight into future events and outcomes that challenge key assumptions, especially where security is concerned. There are a number of predictive models available but only a few listed below are created in this project to demonstrate predictive analytics modelling for attacks using the generated dataset and will be represented in the same order as listed below.

- Linear Regression Models
- Naïve Bayes
- Decision Trees Models
- Logistic Regression Models
- Time Series predictive Models

Each model has a particular use in this project and answers specific questions or deals with a certain type of dataset. For example, to build a linear predictive model, supervised learning is used on the supplied training dataset to perform regression tasks that model a target prediction value based on the independent variables that try to find the relationship between variables and then forecasting. Naïve Bayes on the other hand is a probabilistic classifier that uses Bayes theorem strong independent assumptions between the features. It is used in this case to validate the linear regression model that has been built. Despite the different methodological and mathematical ability, all of them try to predict the future outcomes based on data from the past occurrences. For predictive modelling to be successful, data intended for building the different types of models should be properly labelled and categorised appropriately for machine learning

to be successful. Sufficient sample size data is very important for statistical methods to be consistently successful. Without sufficient data, the models can be produced with the influence of a lot of noise in the data that is used. There are three types of machine learning techniques that are used for building intelligent machines that transform data into knowledge. These machine-learning techniques are briefly explained in Table 3-1 below.

Table 2-2 Types of Machine Learning for Predictive Modelling

Supervised Learning	➢ Labelled Data ➢ Direct Feedback ➢ Predict Outcome
Unsupervised Learning	➢ No labels or targets ➢ No Feedback ➢ Finds Hidden Structure in Data
Reinforcement Learning	➢ Decision Process ➢ Reward System ➢ Learn Series of outcomes

Supervised learning is used on well-labelled data where the algorithm learns from the labelled training data to predict the outcome of unforeseen data. Unsupervised learning is a technique that allows the model to work on unlabelled data to discover hidden information. Reinforcement learning is a goal-oriented technique that allows the agent to learn how to attain complex objectives in an interactive environment using the trial and error method using feedback from its own actions and experiences. To build predictive models the dataset needs to be split into two distinctive datasets called training data and testing data for each model and for each predictive algorithm used. A simple process for building machine-learning systems for predictive models in question was followed. An example of a ML process is illustrated in Figure 3-2.

Process for building Machine Learning Models.

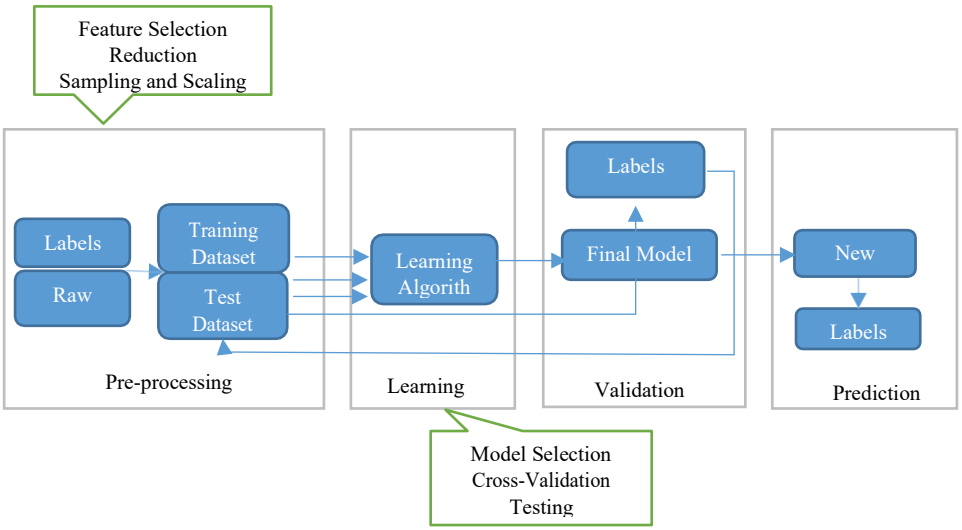


Figure 2-3 Process for Building Machine Learning Models (Liu et al., 2017)

2.3.1 Predictive Analytics Framework

A simple predictive analytics framework is illustrated in the Figure 3.3 below:

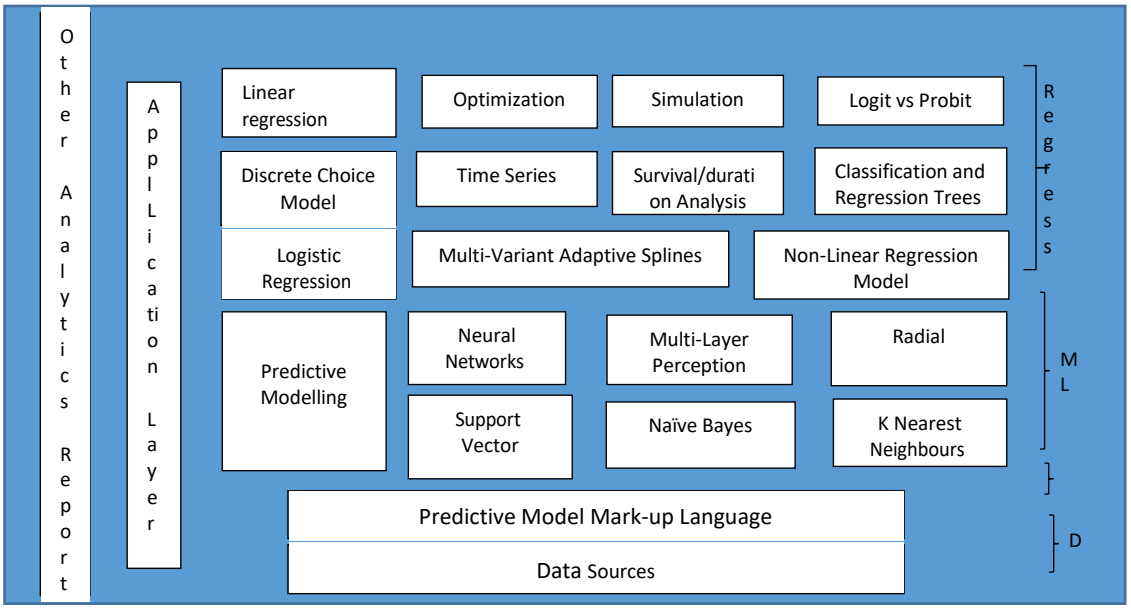


Figure 2-4 Predictive Analytics Framework

2.3.2 Predictive Analytics Concepts

The implementation of predictive analytics as a security management tool in virtualised systems has been the focus of this project. Virtualisation being the heart of cloud technologies

raises many security issues that need to be analysed and mitigated in real time. The use of predictive analytics in this context offers a solution to the many security challenges virtualisation poses to cloud computing. Successful implementation of predictive analytics requires the use and combination of various tools and techniques like big data analytics, predictive modelling, machine learning and data mining techniques to analyse current and historical facts to make relevant predictions about unknown events. The ability to predict attacks in a virtualised environment will help to minimise the risks that come with security breaches like denial of service attacks, side channel attacks, VM escape, malware-based attacks and many more threats. A combination of predictive analytics tools and techniques are used in order to achieve the predictive value in question. Techniques used in predictive analytics are divided between regression techniques and machine learning techniques.

Although there are a number of listed predictive analytics techniques, the main concepts relevant to this project are identified and described below.

2.3.3 Predictive Analytics Tools and Techniques

1. Regression techniques

Regression refers to a linear relationship between the input and output variables. A predictive model with linear functions requires a predictor or feature in order to predict the output or outcome. Regression analysis is therefore a form of predictive technique that is used in investigating the relationship between a target variable also known as a dependent and predictor, independent variable. Regression techniques are used for forecasting, time series modelling and finding the casual effect relationship between the same variables.

Regression modelling is considered the core of predictive analytics because of its focus on mathematical equations that are used to represent interactions between different variables. Regression analysis is a tool for modelling and analysing data. It provides the ability to investigate the relationship between a target variable and a predictor by showing the significant

relationships between the two variables and the strength of the impact of multiple predictors on the target variables. Regression analysis also allows the effects of variables measured on different scales to be compared and in return provides the best set of variables for building a predictive model. In order for regression techniques to be successful, three metrics are considered; number of predictors, the types of target variables and the shape of the regression line. An overview of the regression techniques is illustrated in Figure 3-4 below.

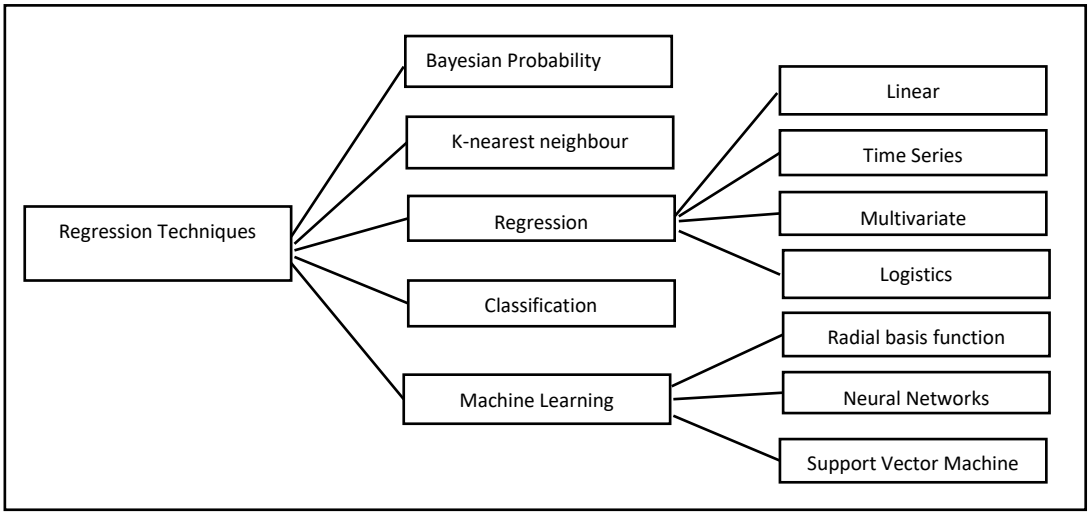


Figure 2-5 Regression Techniques

Although there are a number of techniques that are covered in regression modelling, only a few discussed below are selected and used in this project:

- ***Linear regression***

  Logistic regression is another statistical technique used in machine learning. It is used for binary classification problems as opposed to regression problems of linear regression modelling. The name logistic regression comes from the function of its core method the logistic function, which is also referred to as a sigmoid function. Logistic regression, like linear regression also uses an equation where the input values x are

linearly combined using weights or coefficient values to predict an output value y. The main difference from linear regression is that in logistic regression the output value is a binary value of either 0 or 1 rather than a numeric value. An example of the equation used in logistics regression is:

$$y = e\hat{}(b0 + b1 * x)/(1 + e\hat{}(b0 + b1 * x))$$

Where y is the predicted output, $b0$ is the bias and $b1$ is the coefficient for the single input value x. Linear regression is in most cases the first step in regression analysis. In this type of analysis, the target variables, also referred to as the dependent variables, are continuous, the predictor variables also known as the independent variables are continuous and the regression line is linear. Linear regression establishes a relationship between the target variable (Y) and the predictor (X) by using a straight line called regression line. Linear regression is represented by the equation

$$(Y=a + b*X +e)$$

Where "a" is the intercept, "b" is the slope of the line and "e" is the error term. The equation is used to predict the value of the target variable based on the predictor(s). Linear regression is usually represented as either simple linear regression where the predictor has just one variable or as multiple linear regression where the predictor has multiple variables.

Simple linear regression predicts the score of the second variable called the predictor "X". In simple linear regression there is only one predictor variable "Y" which when plotted as a function of "X" forms a straight line. Multiple linear regression or multi-variant linear regression where multiple variables are predicted, rather than just one, can also be used.

- ***Logistic regression***

Logistic regression is a technique borrowed by machine learning from the statistical way of modelling a binomial outcome with one or more explanatory variables. This is achieved

by measuring the relationship between the categorical dependent variable and one or more independent variables which estimates the probabilities using a logistic function, which is the cumulative logistic distribution. Logistic regression is a classification and not a logistic algorithm, which is used to estimate discrete values like 0/1, Yes/No and True/false depending on a given set of independent variables. Logistic regression predicts the probability of an event occurring by fitting data to a logit function.

A logistic function curve is an "S" shape represented by the equation:

$$f(x) = \frac{L}{1+e-k(x-x_0)}$$

Where:

e = the natural logarithm base

x_0 = the x-value of the curve's midpoint

l = the curve's maximum value

k = the steepness of the curve

The general logistic function with ($k = 1, x_0 = 0, l = 1$) parameters produces the following:

$$f(x) = \frac{1}{1+e^{-x}} = \frac{ex}{1+e^{x}} = \frac{1}{2} + \frac{1}{2tanh(\frac{x}{2})}$$

Due to the exponential function e^{-x} the logistic function for x can be computed over a small range of real numbers as represented in the curve above. Exponential function carries the form $f(x) = b^x$.

Exponential functions in mathematics carry the format of $f(x) = b^{x+c}$ where the input variable (x) occurs as an exponent and c as a constant. The symmetry property of the logistic function then will be:

$$1 - f(x) = f(-x)$$

Showing $x - f(x) - \frac{1}{2}$ as an odd function and $f(x) = \frac{1}{2} + \frac{1}{2} tanh\left(\frac{x}{2}\right)$ as an offset or scaled hyperbolic tangent function, which follows from

$$\boldsymbol{tanh(x) = \frac{e^x - e^x}{e^x + e^x} = \frac{e^x.(1 - e^{-2x})}{e^x.1 + e^{-2x}} = f(sx) - \frac{e - 2x}{1 + e^{-2x}} = f(2x) - \frac{e^{-2x} + 1 + 1}{1 + e^{-2x}} = 2f(2x) - 1}$$

This can be easily derived as:

$$\boldsymbol{f(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{1 + e^x} \frac{d}{dx} f(x) = \frac{e^x.(1 + e^x) - e^x.e^x}{(1 + e^x)2} = f(x)(1 - f(x))}$$

The derivation of this function therefore has $\frac{d}{dx} f(x) = \frac{d}{dx} f(-x)$ properties.

- ***Time series***

  Time series regression is a statistical method that can be used to predict a future response based on the response history and the transfer of the dynamics from relevant predictors. It helps to understand and predict the behaviour of dynamic systems from experimental and observational data. Time series is a series of data points that are either indexed, listed or graphed in time order. It comprises of methods for analysing data in order to extract meaningful data for statistical purposes.

- ***Classification and regression trees.***

  Classification trees are tree models where the target variable can take a discrete set of values. In these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. The decision trees where the target variable can take continuous values and real numbers are called regression trees.

2. Machine learning techniques

Machine learning techniques are artificial intelligence self-adaptive algorithms that increasingly get better analysis and patterns with experience or with new added data; the computational algorithm built into a computer model will process all transactions happening

on the digital platform, find patterns in the data set and point out any anomaly detected by the pattern.

Machine learning techniques, being a part of artificial intelligence, consist of advanced statistical methods for regression and classification that are used to develop techniques that enable computers to learn. Machine learning techniques emulate human knowledge and understanding to learn from training samples to predict future events. Although there are a number of machine learning techniques available, the following are relevant to this project:

- *Naïve Bayes.*

  Is a simple technique for constructing classifiers: models that assign a class label to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. All naïve Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. One of the advantages of Naïve Bayes is that it only requires a small number of training data to estimate the parameters necessary for classification.

2.4 Motivation and Problem Statement

Predictive analytics, as explained by Shmueli et al, includes statistical models and other empirical methods that are aimed at creating empirical predictions, as opposed to predictions that follow theory only, as well as methods for assessing the quality of those predictions in practice (Shmueli, 2010, Shmueli and Koppius, 2011). Shmueli et al observe that although predictive analytics is a core scientific activity, empirical modelling in information systems has been dominated by casual explanatory statistical modelling, where statistical inference is used to test casual hypotheses and to evaluate explanatory power underlying casual models (Shmueli, 2010, Shmueli and Koppius, 2011).

The increasing nature of threats to security requires that anticipatory reasoning become an everyday activity (Sanfilippo, 2010). Research shows that there is a difference between empirical predictive modelling (Sanfilippo, 2010) and explanatory statistical modelling (Luo et al., 2011, Pearce et al., 2013, Shmueli and Koppius, 2011). Empirical predictive modelling includes statistical models, data mining algorithms and methods for evaluating predictive accuracy whereas explanatory statistical modelling includes statistical models that are used to test hypotheses and methods for testing these statistics (Luo et al., 2011, Pearce et al., 2013, Sahoo et al., 2010, Shmueli and Koppius, 2011).

Table 2-2 below illustrates the differences between explanatory analytics and predictive analytics:

Table 2-3 Explanatory Analytics vs Predictive Analytics

Plan	**Explanatory Analytics**	**Predictive Analytics**
Goal	Used for testing casual Hypothesis	Used for predicting new observations and assessing predictability levels
Set Variables	Used only to study underlying conceptual constructs and relationships between the set variables	Focuses on observations and measurable variables.
Modelling Optimized Function	Focuses on minimizing model bias	Focuses on minimizing the combined bias and variance
Modelling Constrains	Empirical model must be interpretable and must support the statistical testing of the hypothesis in question and model must adhere to the theoretical model.	Uses variables that are available at the time of deployment.
Model Evaluation	Explanatory power of the model is measured by how well the model fits the measures and tests.	Predictive power is measured by how accurate the model is with out-of-sample predictions

Predictive analytics is becoming increasingly mainstream with most organisations using it for improving customer engagement, managing risks, reducing fraud or optimising the supply chain (Elkan, 2013, Shmueli and Koppius, 2011).

The main motivation for the project is to design a predictive analytics modelling framework as well as the ability to use empirical predictive analytics for risk analysis and security management in virtualised systems.

The ability to adopt empirical predictive analytics into virtualisation as a security management solution, as opposed to the solutions that follow theory only, is the driver of this research.

2.5 Summary

Security attacks on information systems are becoming more prevalent as cyber attackers aim to exploit information systems like virtualised systems in cloud computing for personal and financial gain. Theft of private and sensitive organisational data and/or the destruction of infrastructure are just a few motives resulting from espionage among many others. Since cyber attackers are aware of the existing security controls and have access to a wide range of tools and techniques to bypass traditional security mechanisms, the time taken from the initial attack to when the attack is actually detected can be measured in days or even weeks. Attacks like zero day exploits, malware infection frameworks, rootkits and browser exploit packs are readily available for purchase on unauthorised markets. This makes security breaches inevitable, as the cyber attackers seem to be getting one step ahead. Early detection or even prediction are the best defence to surviving an attack. Detection and prevention methodologies that help reduce the risks of security breaches or attacks and those that help to mitigate or even prevent them, need to be put in place. Predictive analytics provides important navigational tools to organisations, companies and individuals to successfully reach or attain their destinations through forecasting what is about to happen so that they can respond accordingly to stay on the

most accurate, safe repeatable, profitable and efficient course. The use of predictive analytics is already transforming the way we are interacting with our environment, especially in the computer world, as the quantity of data increases and the quality improves, facilitated by the availability of the cost-efficient processing power, predictive analytics is bound to be even more pervasive than it is at present. Virtualisation, despite its many benefits to cloud computing still raises a number of security issues that concern both the service providers and the end users. These security issues prevent end users from fully embracing cloud computing and the adoption of virtualisation as a whole. In the bid to secure its resources, cloud service providers in conjunction with researchers and security professionals are on the race to find adequate security solutions. As predictive analytics has worked so well by improving the various processes in other disciplines like medicine and e-commerce, the adoption or use of this powerful tool in information systems, more especially as a security management tool for virtualised environments, can add another level to security and other IT processes that need improving. Successful implementation of empirical predictive analytics model as a security management tool in virtualised environments demonstrates a novel contribution to the so much needed virtualisation security solutions among others. To implement the predictive analytics model for the management of security in virtualised information systems an experimental environment used for emulating attacks and to generate data for predictive modelling was setup as presented in chapter 3.

3. Framework for Predictive Analytics Modelling and Methodology

3.1 Introduction

Predictive analytics brings together advanced analytics capabilities ranging from statistical analysis, predictive modelling, data mining, optimisation, real time scoring and machine learning. In order for predictive analytics to be successful, a number of iterative processes need to be followed as shown in the figure 3-1.

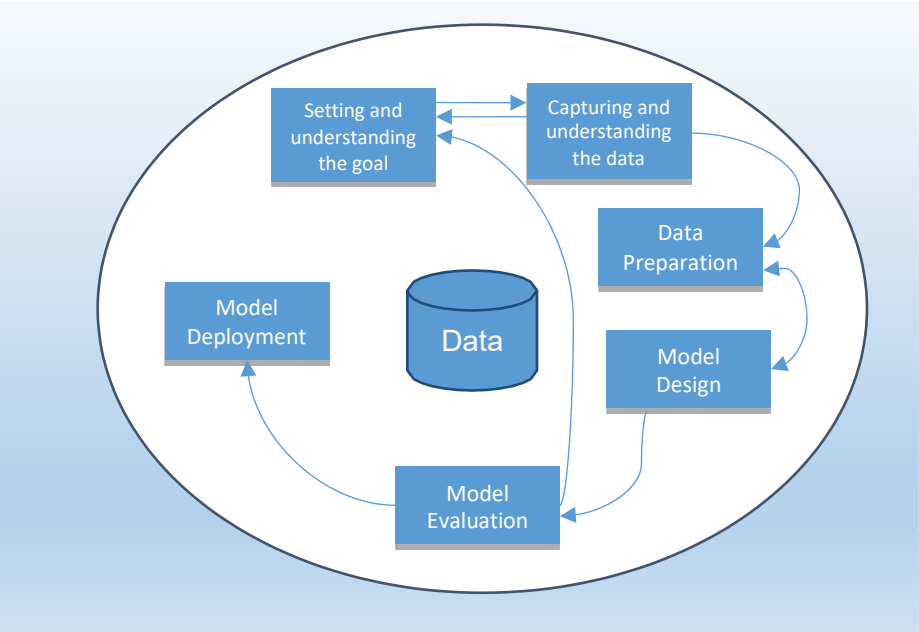


Figure 3-1 Predictive Analytics Process

For predictive analytics to be successfully implemented, each phase of the process in Figure 3-1 needs to be accomplished before moving on to the next stage. To achieve the stages of the process, a predictive analytics modelling framework presented in figure 3-2 was designed and used to successfully build, implement and deploy the predictive analytics model security risk analysis and management of threats for virtualised systems in cloud computing.

3.2 Predictive Analytics Modelling Framework

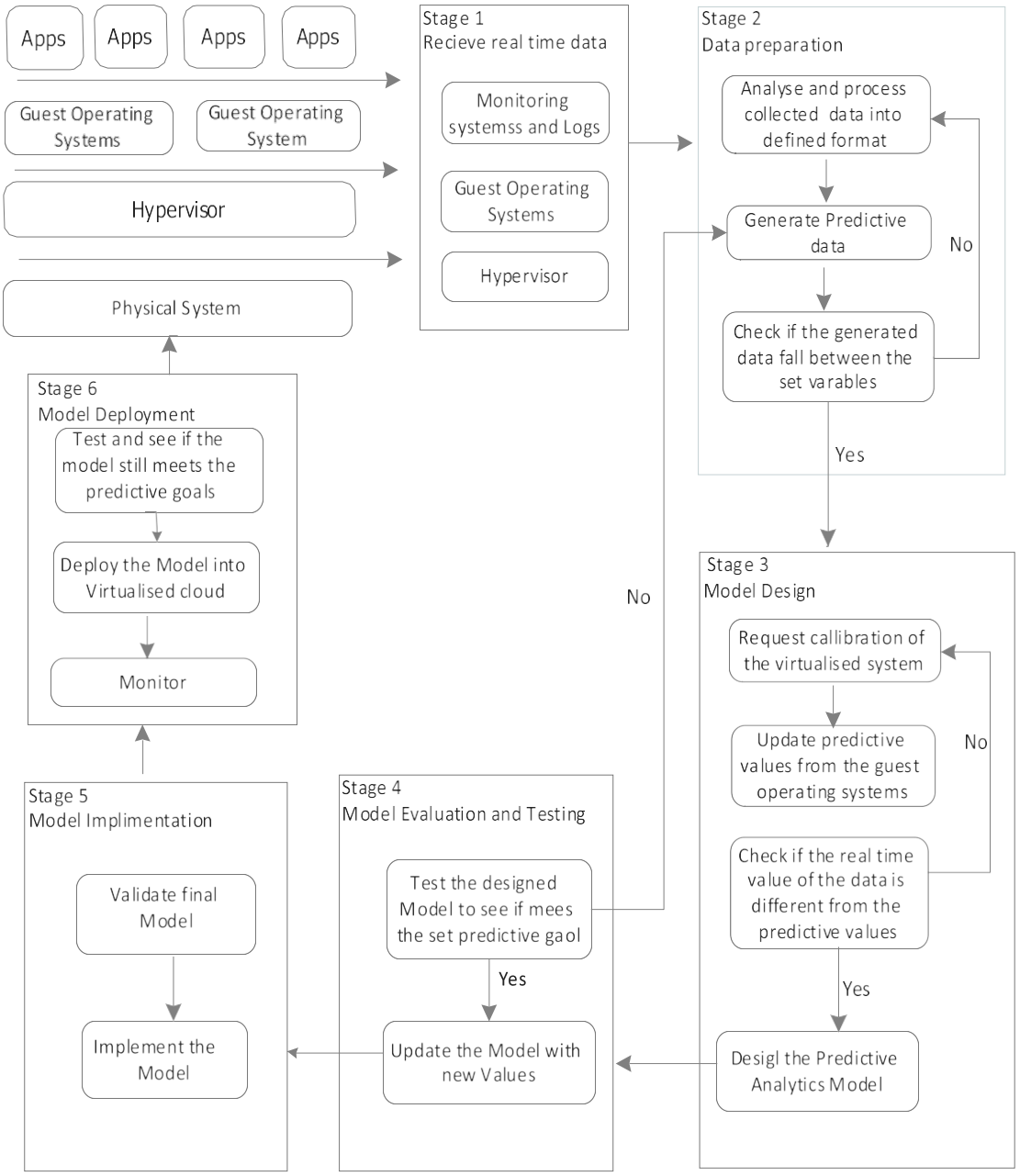


Figure 3-2 Predictive Analytics Modelling Framework

The predictive analytics modelling framework is divided into distinctive stages with mini processes at each stage. The designed framework follows an iterative process which ensures that all the processes vital to the successful achievement of the set predictive values and goals of each stage are successfully achieved before moving onto the next stage as each stage needs

to be checked and validated against the set predictive goals. The process at each stage are discussed in the section that follows.

Stage 1 Receive Real Time Data

The first stage of the framework is to collect data from the specified sources which in this case, are the logs of the virtual system deployed on the host of the test environment. The collected data is analysed and then measured for predictive quality to see if it matches the initial predictive goals. Supervised learning and data driven algorithms are used to help understand the relationships and patterns of the collected data that exist in both input and valid output data to generate the predictive value used to measure against model design and development. The methods of collecting data may vary according to the discipline. The emphasis on ensuring accurate and honest collection remains the same in all situations. Many scholars agree with this opinion (Sapsford and Jupp, 1996a, Sapsford and Jupp, 1996b); they put it that a formal data collection process is important as it ensures that the gathered data is both defined and accurate and that subsequent decisions on arguments embodied in the findings are valid. The figure illustrates the process of ensuring that the correct data is collected before passing it on to stage 2 for processing.

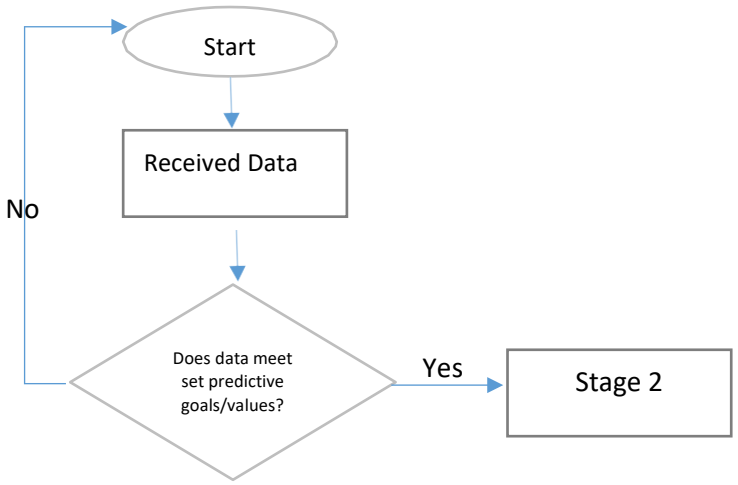


Figure 3-3 Data Collection Process

Stage 2 Data Preparation

The data preparation process at this stage is used to transform the data collected in stage 1into meaningful data that can be used for predictive modelling. Data preparation process is the gathering, cleaning, and integrating of the collected data into a single file or data table, with the purpose of using it in both exploratory and predictive analysis. The results from the first stage of the predictive analytics modelling process are used to prepare and transform the data into the relevant format for analysis. This stage of the framework is used in handling unstructured, inconsistent, and/or unstandardized data by combining data from multiple sources by using data analysis algorithms such a classification and regression to check the data for missing values and creating an intuitive workflow, validation, transformation and backflow of cleaned data. Variables, and other elements relevant to meet the predictive goals are either added or removed from the dataset. The data is finally checked to ensure it falls between the set predictive values. This determines whether the missing values are added to the predicted observations or are reserved for model training. The final data is partitioned into three parts; the first for the training the model, the second dataset is used to evaluate performance of the final predictive model and the last dataset is used in the validation stage for model testing.

Figure 3-3 shows the relationships between the data collection and data preparation processes. The first two processes are carried out in an interactive way until the relevant data is extracted and the predictive value determined.

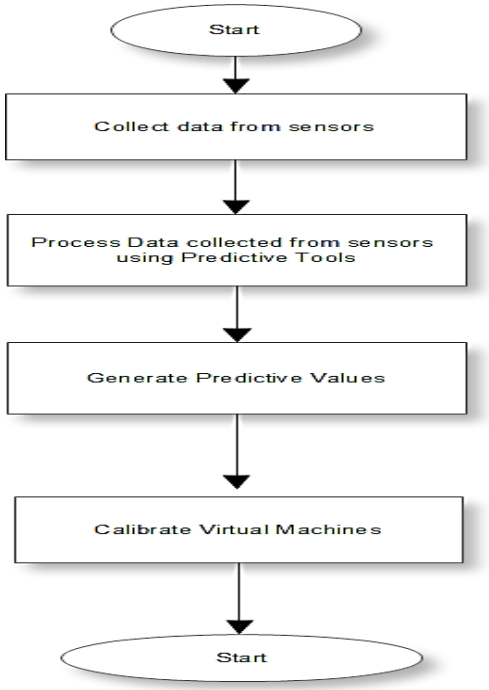


Figure 3-4 Data collection and Data Preparation Processes

Stage 3 Model Design

At this stage, the prepared data set is analysed further, validated and tested to ensure it meets the set predictive value. The data is calibrated with the virtualised system by comparing real time values of the data is different from the set predictive values if so the predictive valuesare updated according before building the model if not the data needs to be rechecked against the time data of the virtualised system. Machine learning algorithms like classification and regression algorithms are used to transform the results into knowledge. Input and output predictive values for supervised learning are set. Supervised learning techniques are applied at this stage where the set input values are fed into the training model and the results measured against the set predictive output values. Graphs such as box plots and scatter plots are used to illustrate the results. The model is then built using predictive analytics modelling techniques such as linear regression which is used to build the predictive model and logistic and Naïve

Bayes used to test and validate the built model. Time series is used to determine the time value of the built predictive model.

Figure 3-5 below illustrates the data analysis and modelling processes highlighted above.

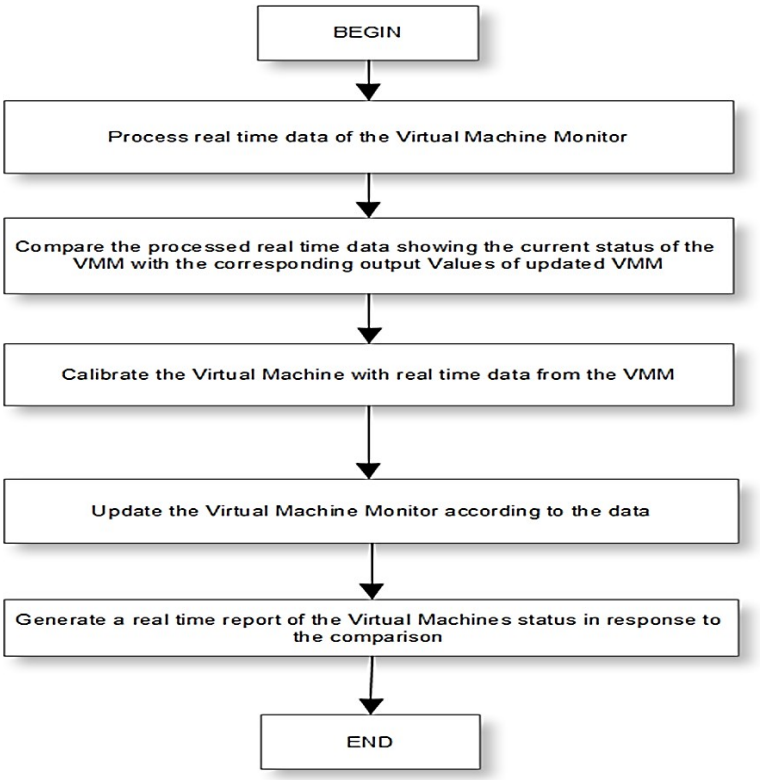


Figure 3-5 Data analysis and Model Design

Stage 4 Model Evaluation and Testing

Model Evaluation is an integral part of the model development process. Model evaluation at this stage helps to find and choose the best model that represents the available data and how the best-chosen model will work in future. At this stage two methods (Holdout and Cross-Validation methods) of evaluating the model are followed.

In the Holdout method, the large dataset is indiscriminately divided into three subsets

1. Training set that is used to build the predictive model.

2. Validation set, which is used to assess the performance of the model, built in the training stage. This provides a test platform to fine-tune the built model and its parameters to help select the best values for model.
3. The last set is the test set which is also known as the unseen example. This dataset is used to assess the likely future performance of a model. Model Evaluation is divided into two sections: Classification Evaluation and Regression Evaluation (Mavrogiorgou et al., 2017).

The predictive accuracy is measured using the third set of data obtained in the data preparation stage to validate the model and to test the predictive value of the model. The model is then assessed for overfitting by comparing the training data with the output data obtained in the early stages of the predictive modelling. The early stages of the framework is continuously revisited and predictive tests carried out to ensure that the predictive value initially set, has been met. The final model, depending on its predictive accuracy, is then implemented.

Figure 3-6 below illustrates the model evaluation and testing process of the predictive analytics modelling process.

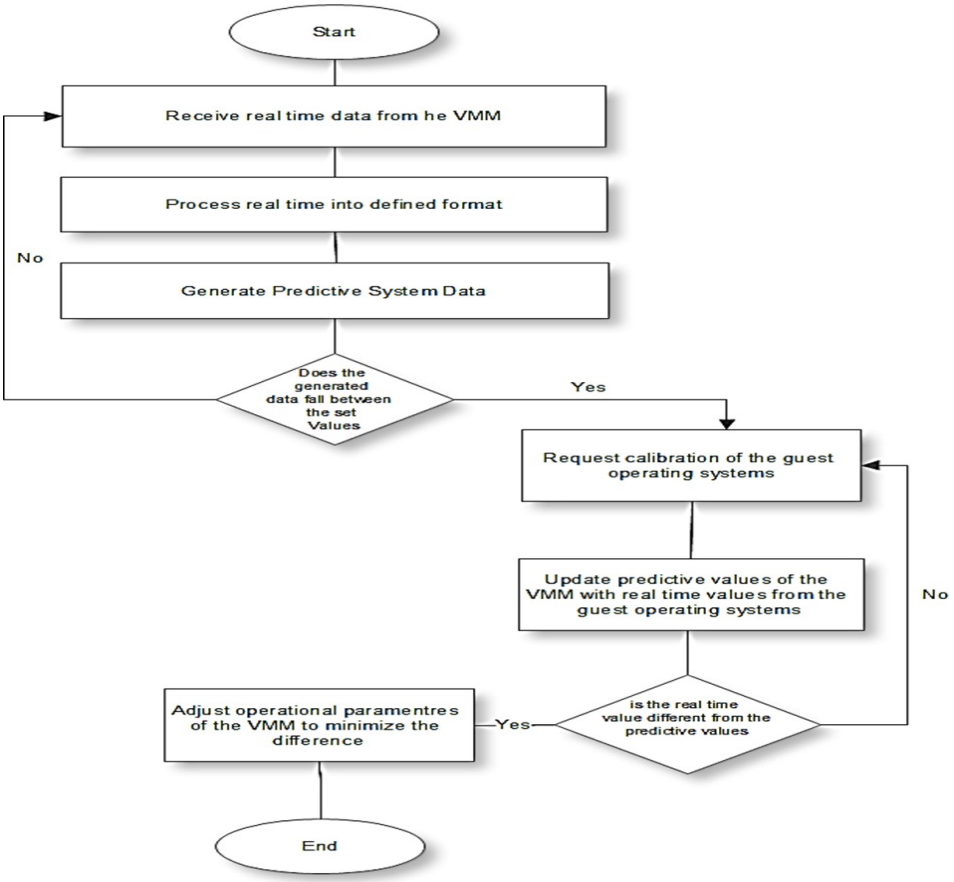


Figure 3-6 Model Evaluation and Testing

Stage 5 Implementation

Stage 5 is the most important phase of the predictive modelling framework as it determines whether the designed predictive analytics model meets the set predictive goals of the project before it is finally deployed into a live virtualised system. There are so three main implementation methods available but pilot implementation method is used. This allows the predictive model to be implemented on the setup experimental environment and then tested for predictive accuracy before it is actually goes live. This ensures that the predictive model is built and implemented correctly and the results meet the set predictive goals which is accurate attack prediction in his case. The process followed implemented in this stage is illustrated in figure 3-7 below.

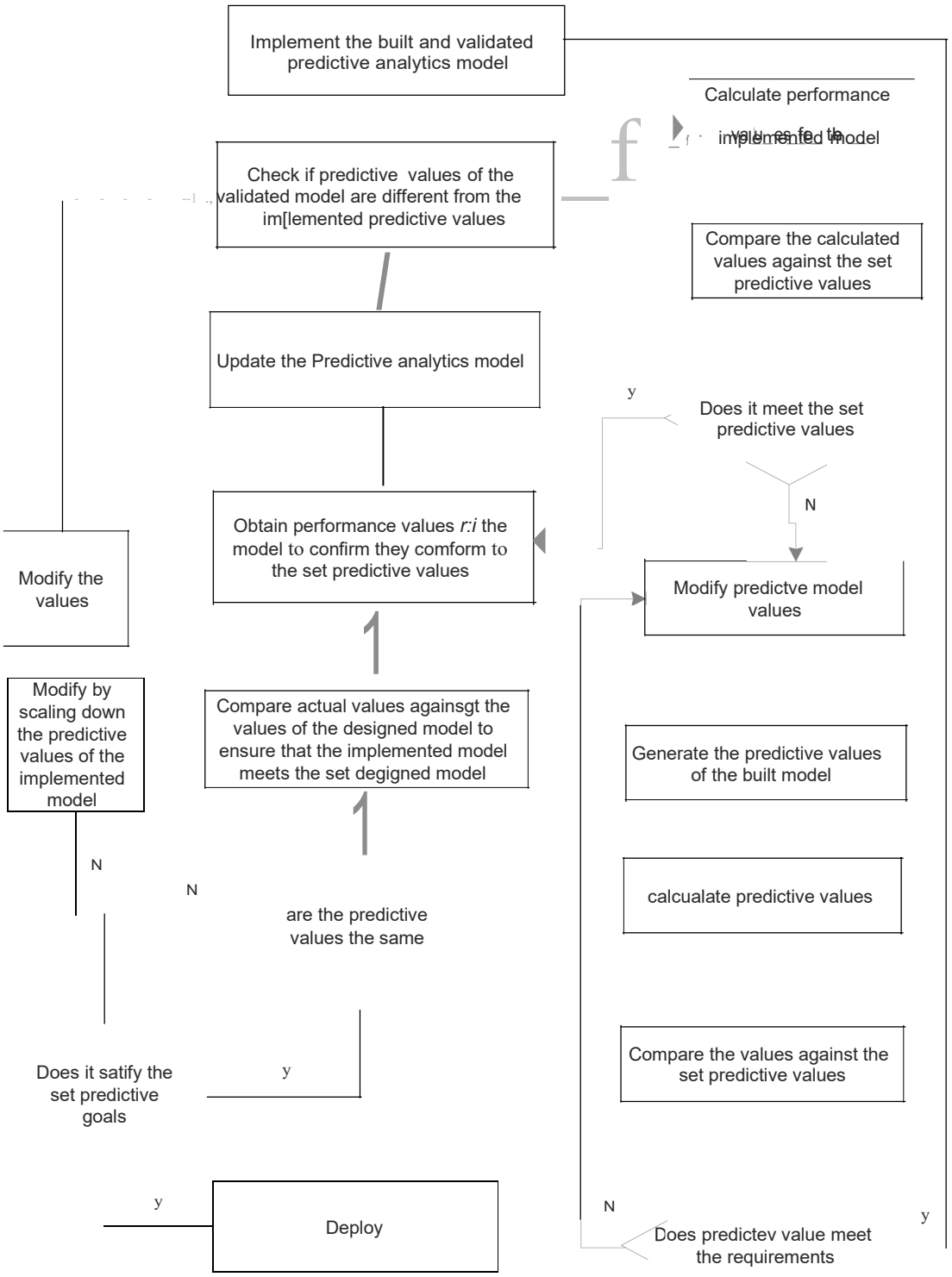


Figure 3-7 Implementation Process

The tested against the set predictive values in Once predicve deployed into the live real time virtualised system in cloud computing and then monitored for performance.

Model Deployment

Model Deployment provides the possibility to position the analytical results in the everyday decision-making process for insightful decision making, reports and outcomes by automating the decisions based on the modelling. The developed model will be deployed on the virtual system that will be created to test the hypothesis. This will include monitoring and testing the model for predictive accuracy from the data collection in stage one where data is collected from the virtual environment and then passed on for preparation in stage two. In stage two, simple analytics are carried out to prepare the data for predictive analytics and tested to ensure the data meets the required values for predictive modelling before passing on to the third stage where the actual predictive modelling is carried out. The model is designed and calibrated with the virtualised environment to ensure the correct data is used for the predictive value in question, if not, the data is updated and the model is built accordingly. In stage four, the validation of the built model is carried out. The model is tested against new data to ensure the correct values are being predicted and new values are updated and validated accordingly. The final stage five is where the model is deployed into real life operations of the virtual environment. Deployment is the most difficult and time-consuming part of the process, as the model needs to be reconfigured and reprogrammed if programming is used to build and deploy the model. This can cause some functions of the model to stop working properly. The process continues in an iterative manner until the results of the predictive model are satisfactory.

To help follow through and accomplish the stages of the framework effectively, a data flow diagram was designed as shown in Figure 4-4.

3.2.1 Data Flow Diagram

The processes of the data flow diagram are explained in the subsection that follows. This highlights the tools and techniques used at each stage of the process that will lead to the

successful achievement of the proposed project's goal. Data collected from the IDS, logs and monitoring tool is then analysed and defined into relevant formats in order to generate the predictive data. This data is then used to design a model, which is then evaluated, tested and deployed.

Data Flow Diagram

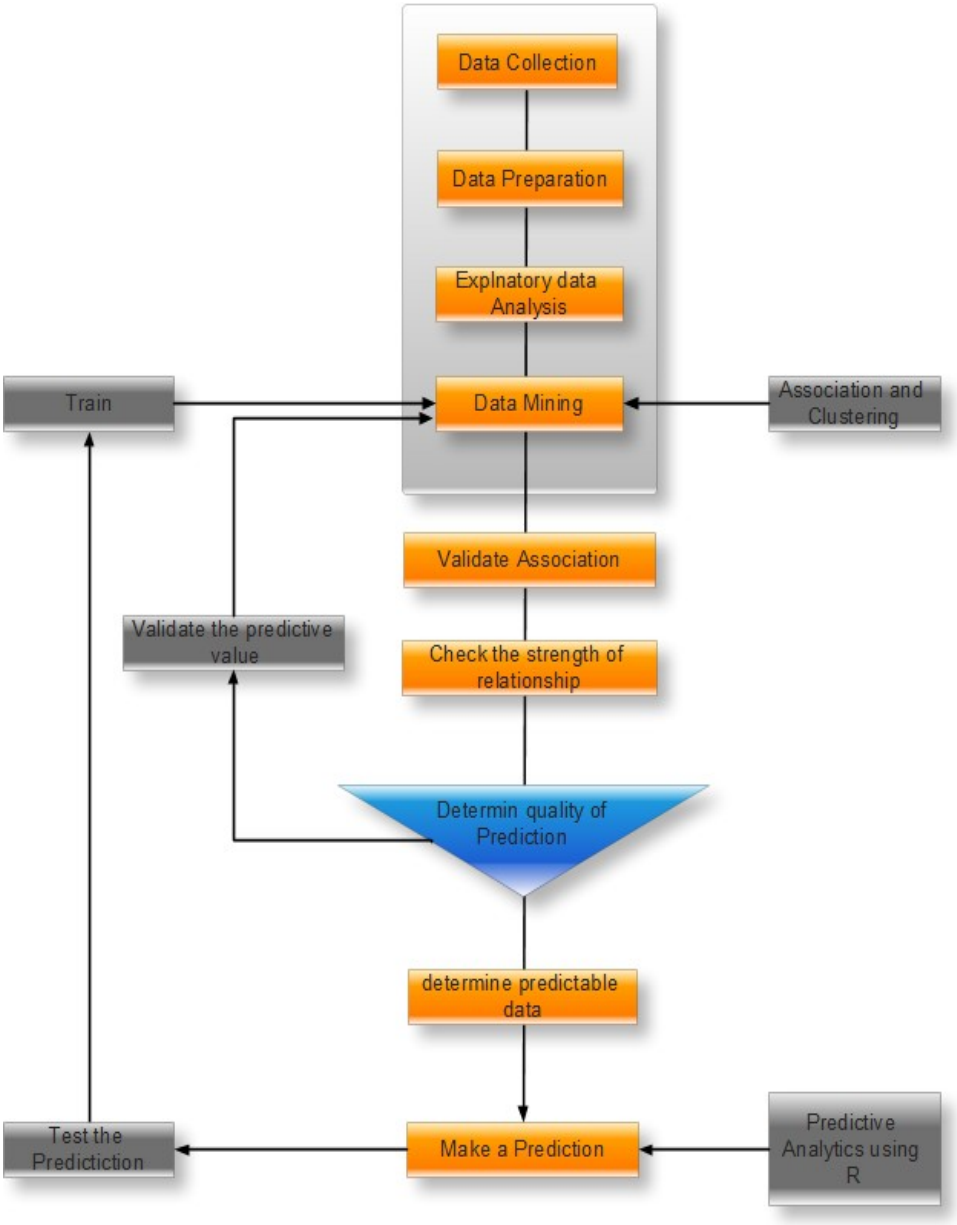


Figure 3-8 Data Flow Diagram

3.3 Choice of platform for Predictive Analytics

Predictive analytics, being the focus of this project, requires a platform that is not only efficient and effective but also indispensable for efficient deployment of the predictive model in question. In order to achieve the goal of the project a number of tools and techniques are used in this project. For predictive analytics a combination of data science technologies like Weka, R and Python together with its tools were used for data collection, preparation processing and analysis.

3.4 Choice of Virtualisation

Virtualisation comes in so many forms and platforms. For this project, Virtual Box implemented on Ubuntu was used as the choice and platform for virtualisation. Virtual Box is a free and open source hypervisor that can be installed on a number of host operating systems.

3.5 Experimental Environment for Predictive Analytics Modelling

To successfully design, build and implement the security predictive model in question, an experimental environment for the simulation of attacks, was set up as illustrated in Figure4-1. The diagram provides an overview of the architecture with the various tools and techniques that were put together to assist with the achievement of the project in question. The attack detection process, the detection infrastructure and finally the computational process of the detection process where also established as highlighted and discussed later in the chapter.

3.5.1 Architecture Overview

By Architecture Overview, we mean the fundamental organisation of a system, concretised in its components and their relationships to each other and the nature and rules that govern its design and evolution (Platt, 2002).

For the project, we implement and provide the architecture in Figure 3-6, which illustrates an overview of the set-up environment for the simulation of experiments, data collection and reporting.

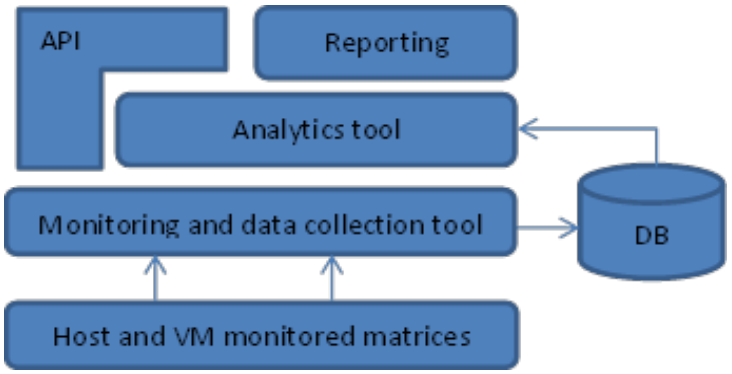


Figure 3-9 Architecture Overview

3.5.2 Application Programming interface

The main purpose of the application-programming interface (API) is to provide a platform that allows for robust tailoring of the defined process specifications. The API is made up of defined methods and processes that provide a set of definitions, protocols and tools to help build the predictive model. The API used in this case is Python because of the vast selection of libraries and tools that the project needed for successful completion.

3.5.3 Reporting

The reporting tool is used for presenting and visualising the findings of the simulated experiments. Reporting tools enable users to represent the results of any analysis using various graphical representation such as plots, graphs or even charts. The reporting tools used in this project are R, Weka and Zabbix, which is the external monitoring tool have been used to help visualise the results for predictive modelling. Both R and Weka have been chosen as choices for reporting because they both provide an environment for statistical computing like linear and non-linear modelling, time series analysis, classification and clustering which is relevant

to the project. Zabbix has been chosen for its real time monitoring and visualisation capabilities.

3.5.4 Analytics Tool

An analytics tool provides the user the ability to exploit the collected data for visualisation and reporting. An analytics tool is used to analyse data for actionable insights. Analytics tools can be used for experiments and experimental design. They are techniques used to validate strategic hypotheses, spot patterns, determine and measure relationships between two separate variables. Analytics tools can also be used for regression analysis, scenario analysis and time series analysis. Weka together with R, due to their ability to do all the above are the choices of analytics tools used in this project.

3.5.5 Monitoring and Data collection tool

Monitoring of the system is an essential part of this project because it provides an overall visual representation of what is happening in the set up virtualised environment. Monitoring logs the activities of the virtualised systems, which are used to compare with the results from the analytical tool to help verify the predictive goal. Data collection is the process of logging the raw data of the system's activities, which is used for analysis, reporting and visualisation. Data collection involves the process of identifying and extracting log files from the virtual environment. This is done using several tools like database, through the monitoring tool and through the API where data is pre-processed into a format that can be used by machine learning algorithms.

3.5.6 Database

The database is used for keeping the collected data logs and matrices of the simulated experiments. Two databases are used in this case; one database is from the monitoring tool side

and the other one is installed on the attack simulation tool. The use of two databases helps compare the two logs of data through analysis for validation purposes.

3.5.7 Host and VM Monitored Matrices

Ubuntu is used as the host for the virtualised environment and virtual box as the virtualisation. The matrices being monitored are the incoming network traffic that is the number of packets in and out of the network as well as the performance of the system in general most specifically the web application services.

3.6 Summary

To sum up, the chapter has presented the architectural overview of the set-up environment for the intended experiments. The components of the architecture have been defined and the purpose for each used component has been highlighted. The chapter also discusses the methodology followed in this project that helped in the achievement, implementation and adoption of predictive analytics as a security tool for managing security in virtualised environments. The choice of platform used for predictive analytics together with the choice of virtualisation used are also discussed this chapter.

To prepare for predictive analytics, data was collected from the set-up environment through the simulation of attacks as presented in chapter 5, which discusses the attack simulation process for data collection and the generation of the required dataset for predictive modelling.

4. Simulation of Attacks for Data Collection.

4.1 Introduction

Attack simulation as the name suggest is the assimilation of real-life cyber-attacks on virtualised systems using a test environment. Effective design, implementation, testing and analysis of the simulated attack environment require a combination of components and an assortment of tools and techniques. The tools and techniques highlighted and reviewed in

Chapter 4 are used to demonstrate the development and simulation of attack scenarios in this project. Simulation of attacks is carried out on a test lab set-up on Ubuntu as a host and virtual box as a platform for virtualisation. A number of virtual machines are installed onto the platform and these are used to simulate attacks. Kali Linux, because of the vast collection of tools and techniques it offers, is used to carry out various attacks on the installed virtual machines as well as for intrusion detection and prevention techniques. A successful attack simulation requires knowledge and understanding of the target system, together with the knowledge of the network, the platform and the methods to handle imperfect knowledge of the target systems and networks. Creating a simulation with a high probability of success for example, demands knowledge of the target system's IP address, network topology or/and domain names if the attack is against the known network, or user names, architecture type services running on a system and ports if the target is the host.

Attack simulation has two main processes; vulnerability assessment, which is a step by step process of assessing system risks and vulnerabilities and penetration testing, which is the actual process of performing attacks on the vulnerable system. Knowledge of these vulnerabilities can be achieved through foot printing and/or scanning of passive and/or active virtual machines which leads to intelligence gathering. Intelligence gathering allows the attacker to learn more about the target system. Running attack simulations in a virtualised environment requires the system to be able to reset and reconfigure the simulation environment rapidly. It also requires a list of configurable options for various effects, a set timeline for activities together with their execution, knowledge of vulnerabilities of the target network or system and the ability to test the simulation.

- Resetting and Reconfiguring – this is the situation where the simulation environment has the ability to be reset or reconfigured as quickly as possible after an attack has been

simulated. This should allow components to be added, removed or modified with new parameters and the results to be saved and captured for each participating simulation.

- List-of-Configurable – this provides a list of modules to be used in the simulation. New tools can be developed to supplement old tools to provide for enhanced simulations.
- Timelines – being able to see the timeline of different activities of the simulation is key. Attack simulation behaviours and activities can be added to the module.
- Knowledge of Vulnerabilities – attacks are usually focused on networks and systems that are vulnerable. In order for the attacker to conduct a successful attack, knowledge of the target system is required.

The experiment environment was set up on Ubuntu as illustrated in Figure 5-1 below. The environment has three main components identified as the virtualised systems, which represents all the virtual machines installed for the simulation of attacks; the data processing, analytics and management, which involves all the attack simulation processes from data collection, data preparation, classification, analysis and management; and lastly results visualisation, presentation and reports. The host gateway is the virtual network of the host system and the installed virtual machines. This network is important because it provides the necessary network topology needed for the simulation of the distributed denial of service attacks that the project focuses on.

Set up Attack simulation Testbed

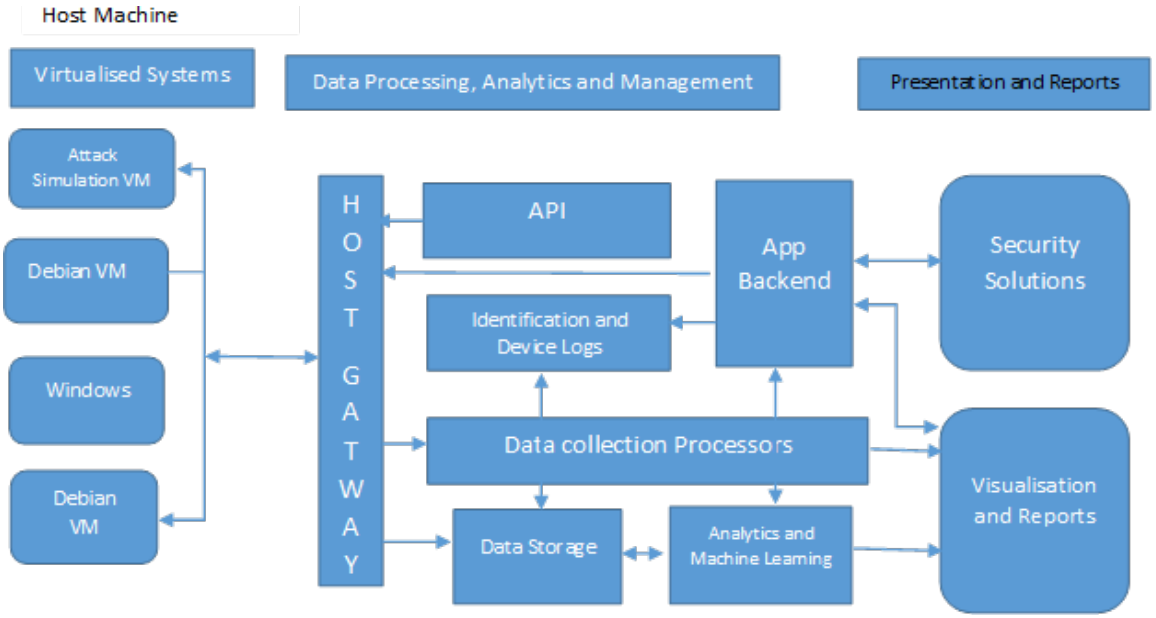


Figure 4-1 Experiment Environment

Attack Simulation Framework

There are a number of attack simulation approaches and frameworks available. These are mainly based on the attacks being simulated and the environments where these attacks are simulated. The proposed attack simulation framework (ASF) has been benchmarked with the other standard frameworks. There are three main steps in this framework as illustrated in Figure 5-2.

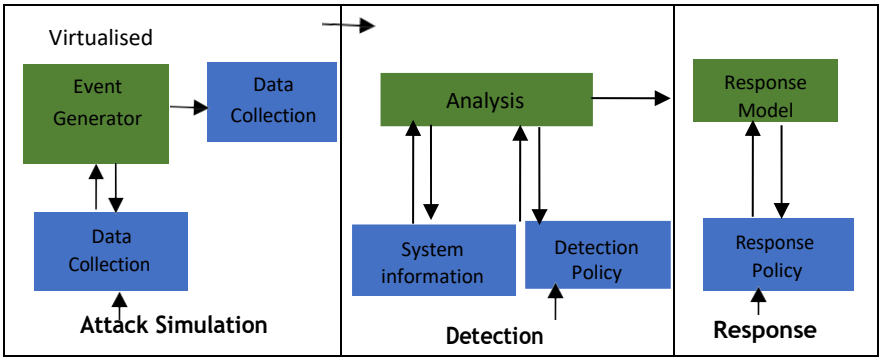


Figure 4-2 Computational Process

The three steps (Attack Simulation, Detection and Response) of the framework in figure 4-2 are:

1. Attack simulation - which is where the initial steps of attack are coined. At this stage, the target system is identified by foot printing and scanning the existing virtual network. The methods and policies for data collection are set and the type of intended attacks for the target system are identified by the attacker. The main purpose of this stage of the project is to help generate relevant data logs needed for analysis and visualisation in preparation for predictive analytics.
2. Detection – this is where an attack or exploit of the system is discovered on the target system side. Identification or detection of an attack is done with the help of an intrusion detection system (IDS) which uses analytical methods to scan through the collected data to determine whether an attack occurred or has been successful. This stage of the framework is important especially for this project because it helps to prove that the intended attack or exploit has actually been successful.
3. Response – this is the final stage of the framework where a possible response to the attack, depending on the set policies, is developed. The response to the attack comes in various forms like stetting the security measures of the systems, for example the use of IDS to help sniff out the attacks, the use of an antivirus depending on the kind of attack. This stage is where predictive analytics would come in, in terms of this project. If achieved, predictive analytics can be used to predict attacks using the vulnerabilities and network activities of the target system.

The three main stages of the simulation framework are applied in the form of experiments and results are highlighted in the next part of the section that follows. Figure 4-3 breaks down the activities of the simulation framework.

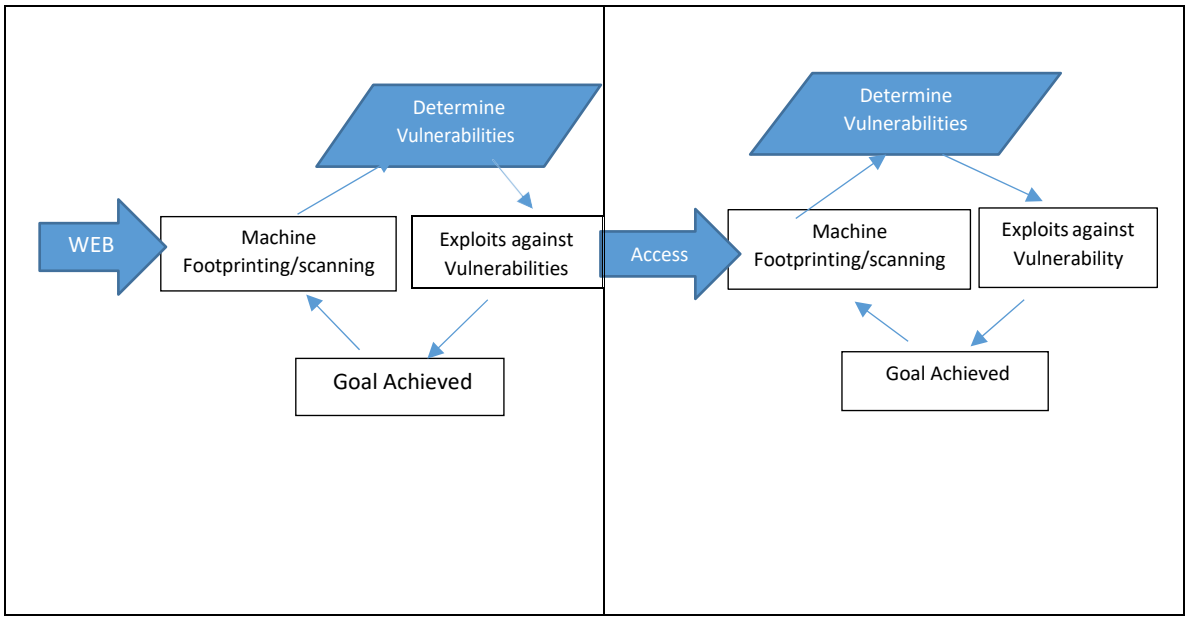


Figure 4-3 Simulation Framework Activities

4.2 Attack Simulation

Attack simulation, due to the nature of the project scope, is carried out on the experimental testbed set up in the lab. The test environment is set up on Ubuntu machine as the host with virtual box as the choice of virtualisation. A number of virtual machines used mainly with different operating systems are also installed on virtual box. Although a number of VMs are installed for experimental and data generation purposes, only Kali Linux and Metasploitable are used to the simulation of attacks.

Kali Linux is used as an attacking system because of the various tools it offers. There are a number of tools available, only a few explained below were used to achieve this project.

1. **Information gathering** – these are tools that gather various information about the target system or virtual machine. There are a number of tools for information gathering.
    - **Hping3 Package Description** is a command line TCP/IP packet assembler and analyser. The interface is inspired by the ping(8) Unix command although hping3 does not only send ICMP echo requests it also supports TCP, UDP,

ICMP and RAW_IP protocols and has a traceroute mode with the ability to send files between covered channels. Hping3 can be used for firewall testing, advanced port scanning and remote operating system fingerprinting among others.

- **Nmap** known as network mapper is used for network discovery and security auditing.
- **dnmap Package Description** is a framework that distributes nmap scans among several clients.
- **Amap Package Description** which is the next generation scanning tool that attempts to identify applications running on various ports other than the normal port. It also identifies non-ascii applications by sending trigger packets and listening for responses in the list of response strings; an example of this is shown in the figure below:

2. **Vulnerability Analysis**

- **Nmap** is not only used for network discovery and security auditing it can also be used to determine hosts available on the network, what services the hosts are running, the host's operating system as well as the packet filters that the hosts are in using.

3. **Exploitation tools**

- Metasploit framework is a platform that provides the infrastructure, content and tools to enable the user to exploit and validate vulnerabilities.
- Armitage is a scriptable collaboration tool for metasploit that visualises targets, recommends exploits and exposes the post-exploitation features in the framework

4. **Forensic tools**

5. **Sniffing and Spoofing**

- Wireshark is the network protocol analyser that provides an insight of what is happening on the network at the lowest level. Wireshark performs inspections of the protocols while adding on more. It performs live capture of the network activities and allows for offline analysis. It also has rich VoIP analysis.

6. **Reporting Tools**

Metasploitable with all its vulnerabilities is the target system.

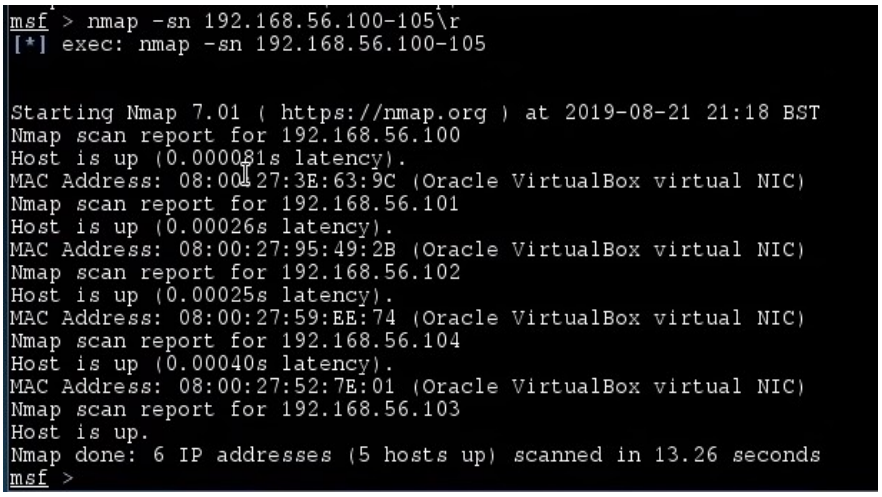
4.3 Information Gathering

Information gathering is the first stage of the project. Information gathering can be achieved by using different tools like Nmap or Hping3 as explained in the earlier chapter. Nmap as described is used to find available hosts on the network. Hping3 on the other hand is used as an advanced ping tool that has the ability to bypass any firewall to use TCP, UDP, ICMP and RAW-IP protocols. For this project both Nmap and Hping3 are used for information gathering. Before any attack is planned or launched, it is essential to find out more information of the existing systems on the set-up virtual network. Information gathering helps to identify possible vulnerabilities of existing systems, which helps to identify the type of attack for each vulnerability. To do this, scans of the network are carried out to determine the IP address of the existing systems. Metasploit framework a pre-installed tool in Kali Linux is used for information gathering.

Using Nmap to scan all available systems on the network, the following command is issued

```
Nmap –sn 192.168.56.100-105
```

The results are displayed below

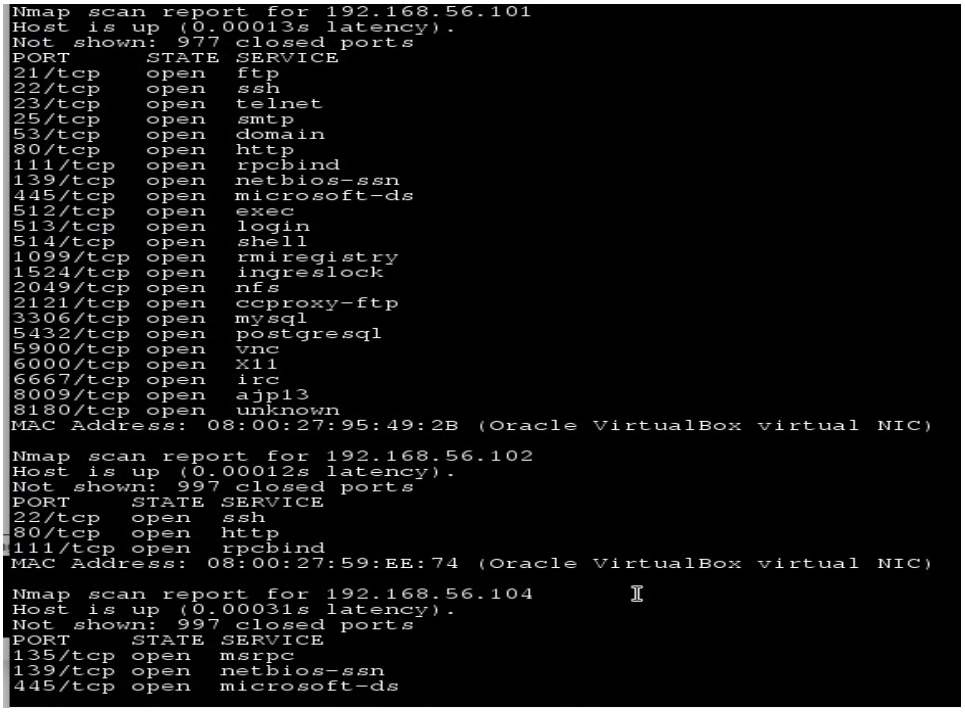


Listing 4-1 Scan Results for available Systems on the Network

These results in Listing 5-1 give an insight of the systems that are running on the virtual network. Having identified the IP addresses, a vulnerability scan is carried out to help identify vulnerable machines on the system.

A full vulnerability scan for all systems is carried out using the command below:

Nmap 192.168.56.100-105



Listing 4-2 Results for full Network Vulnerability Scan

Listing 5-2 above shows the results of a vulnerability scan of three IP address 192.168.56.101, 192.168.56.102 and 192.168.56.104. The system with IP address 192.168.56.101 shows many open ports as compared to IP addresses 192.168.56.102 and 192.168.56.104. This shows that the system with IP address 192.168.56.101 is vulnerable and therefore is the target virtual machine.

To find out more about the ports and the services running on the target system an aggressive scan of the targets TCP ports using Nmap is carried out. The following Nmap command can be issued.

Nmap –p- -A 192.168.56.101

Where “p” is the parameter that indicates all the TCP ports that need to be scanned which is all ports in this case. A is for the aggressive scan of the target to find out all the open ports for the running services.

```
Host is up (0.00037s latency).
Not shown: 65504 closed ports
PORT      STATE SERVICE     VERSION
21/tcp    open  ftp         vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
22/tcp    open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_  2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
23/tcp    open  telnet      Linux telnetd
25/tcp    open  smtp        Postfix smtpd
|_smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTL
ES, 8BITMIME, DSN,
| ssl-cert: Subject: commonName=ubuntu804-base.localdomain/organizationName=OCOSA/stateOrPr
no such thing outside US/countryName=XX
| Not valid before: 2010-03-17T14:07:45
|_Not valid after:  2010-04-16T14:07:45
|_ssl-date: 2019-08-22T13:31:52+00:00; +1s from scanner time.
53/tcp    open  domain      ISC BIND 9.4.2
| dns-nsid:
|_  bind.version: 9.4.2
80/tcp    open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
|_http-title: Metasploitable2 - Linux
111/tcp   open  rpcbind     2 (RPC #100000)
| rpcinfo:
|   program version   port/proto  service
|   100000  2            111/tcp  rpcbind
|   100000  2            111/udp  rpcbind
|   100003  2,3,4       2049/tcp  nfs
|   100003  2,3,4       2049/udp  nfs
|   100005  1,2,3      42648/udp  mountd
|   100005  1,2,3      59475/tcp  mountd
|   100021  1,3,4      46979/tcp  nlockmgr
|   100021  1,3,4      47652/udp  nlockmgr
|   100024  1          35795/udp  status
|_  100024  1          47809/tcp  status
139/tcp   open  netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
512/tcp   open  exec        netkit-rsh rexecd
513/tcp   open  login
514/tcp   open  tcpwrapped
1099/tcp  open  java-rmi    Java RMI Registry
1524/tcp  open  shell       Metasploitable root shell
2049/tcp  open  nfs         2-4 (RPC #100003)
```

Listing 4-3 Aggressive scan Results of the Target System 1

Listings 5-3 above and 5-4 below show results of a more aggressive scan of the target system. The two results show more details of the target system with the open ports and services running on each port for example port 513 shows the authentication port is open and vulnerable for escalation of privilege attacks and port 2121 and 3306 on the other diagram shows an open ftp and an open port for MySQL database.

```
|   100024  1          35795/udp  status
|_  100024  1          47809/tcp  status
2121/tcp  open  ftp          ProFTPD 1.3.1
3306/tcp  open  mysql        MySQL 5.0.51a-3ubuntu5
| mysql-info:
|   Protocol: 53
|   Version: .0.51a-3ubuntu5
|   Thread ID: 8
|   Capabilities flags: 43564
|   Some Capabilities: LongColumnFlag, SupportsTransactions, Support41Auth, Speaks41Protoco
erHandshake, ConnectWithDatabase, SupportsCompression
|   Status: Autocommit
|_  Salt: M1HZj'"p2FJ!bL5bq2D?
3632/tcp  open  distccd      distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
5432/tcp  open  postgresql   PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc          VNC (protocol 3.3)
| vnc-info:
|   Protocol version: 3.3
|   Security types:
|_    Unknown security type (33554432)
6000/tcp  open  X11          (access denied)
6667/tcp  open  irc          Unreal ircd
6697/tcp  open  irc          Unreal ircd
8009/tcp  open  ajp13        Apache Jserv (Protocol v1.3)
|_ajp-methods: Failed to get a valid response for the OPTION request
8180/tcp  open  http         Apache Tomcat/Coyote JSP engine 1.1
|_http-favicon: Apache Tomcat
|_http-server-header: Apache-Coyote/1.1
|_http-title: Apache Tomcat/5.5
8787/tcp  open  drb          Ruby DRb RMI (Ruby 1.8; path /usr/lib/ruby/1.8/drb)
10050/tcp open  tcpwrapped
46979/tcp open  nlockmgr     1-4 (RPC #100021)
| rpcinfo:
|   program version   port/proto  service
|   100000  2           111/tcp  rpcbind
|   100000  2           111/udp  rpcbind
|   100003  2,3,4      2049/tcp  nfs
|   100003  2,3,4      2049/udp  nfs
|   100005  1,2,3     42648/udp  mountd
|   100005  1,2,3     59475/tcp  mountd
|   100021  1,3,4     46979/tcp  nlockmgr
|   100021  1,3,4     47652/udp  nlockmgr
|   100024  1         35795/udp  status
|_  100024  1         47809/tcp  status
47809/tcp open  status       1 (RPC #100024)
```

Listing 4-4 Aggressive scan Results of the Target System 2

Equipped with the information of the target system, different attacks targeting different vulnerabilities can now be launched. There are four main attack categories under consideration in this project. These types of attacks encapsulate a general overview of the attacks identified for the target system. A brief description in the following section helps understand the various types of attack.

4.3.1 Types of attack

There are a number of attack categories available in the computing world. These attacks can be classified in different ways based upon the methods, capabilities and attack vectors that can be used. These types of attacks can be grouped into four main categories named Foot printing, Access, Denial-of-Service and Data Manipulation. These attacks are directed at all the components of the system and in turn affect the benefits that virtualisation promises which are confidentiality, integrity and availability.

1. **Foot printing –** these attacks focus on information gathering. Information gathering in these kinds of attack does not directly compromise the targeted system but creates a situation or a platform for other attacks. The aim of this kind of attack is to gather intelligent information that leverages available information such as network information for the target system. Another way of gathering inelegant information in this type of attack is by using network and port scanning techniques such as Nmap. Data gathered with this kind of techniques includes connectivity of host systems, server status together with their security settings and computer architecture employed. For example, running an Nmap command together with the IP address of the target system in Kali Linux will provide enough information of that system for the attacker to assess if the system is vulnerable or not.
2. **Access -** This type of attack aims at gaining access to an unauthorised system or elevating privileges to computer resources. Access attacks use password-based methods such as brute force techniques among others to try to determine user passwords. Phishing techniques are common methods used to gain privileged access to a target system.
3. **Denial of Service -** The common method of Denial of service (DoS) attacks is to flood the targeted system with unnecessary requests so that it cannot be reached by legitimate users. These kinds of attacks compromise the availability of computer systems to the intended users. It makes it difficult for the end users. Multiple computers can also be used in a more aggressive form of denial of service. This form of aggression is called distributed denial of service. There is another type of denial of service called buffer overflow. This exploits software flaws that allow input of a field or a parameter into an application to make it accept large volumes of data causing it to overwrite legitimate

system data called the execution stack with invalid inputs and malicious code. The idea is to deny the proper end users the services that they need.

4. **Data Manipulation -** These types of attacks exploit network communication protocols by compromising the integrity of the system's data. This attack manipulates the needed data. Techniques used in this type of attack are IP spoofing, man-in-the-middle and session hyper-jacking attacks among many others. In IP spoofing, the attacker impersonates the host system to deceive the defensive packet filtering schemas. With Man-in-the-middle attacks, the attacker places themselves in between the source and the destination of the network communication. This makes it difficult or impossible for the end users to access the services that they need.

All the four attack categories explained herein are used to help generate the needed dataset for predictive analytics. Considering the results obtained in information gathering all the described attacks can be launched onto the target machine, but for the sake of demonstration purposes only a few described below are illustrated and presented in this document. The results together with the data generated from these experiments are also shown at the end of the section.

List of Executed attacks and Duration

Attack	Tools Used	Duration of Attack	Attacker	Target
DoS attack	Hping and SYN Flood	24 hours	Kali Linux	Metasploitable
Web Application attack	Slowloris	48 hours	Kali Linux	Debian Web Server
BruteForce attack	FTP-Patator	24 hours	Kali Linux	Metasploitable
Infiltration	Nmap and Hping	24 hours	Kali Linux	Metasploitable

Table 4-1 List of Executed Attacks

Denial of Service (DOS) attacks: For DoS attack scenario, SYN Flood auxiliary using metasploit and Hping3, both tools in Kali Linux are used. SYN floods are a type of attack that sends huge amounts of Sync to consume all the available resources of the target system. SYN is known as half-open scan because it does not complete the TCP three-way handshake. In this kind of attack, an attacker sends a SYN packet to the target system and waits for a response; if a SYN/ACK frame is received it is assumed that the connection to the target is complete and the port is listening as illustrated in the Figure 5-4.

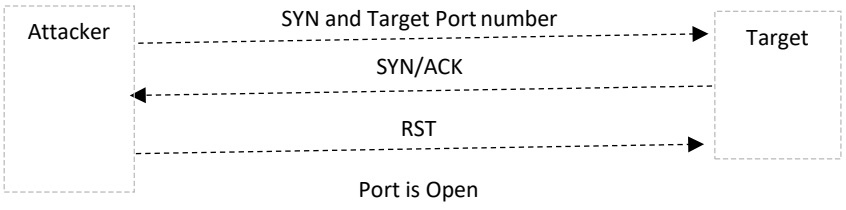


Figure 4-4 Complete SYN/ACK Connection to Target

If it receives an RST then the target port is assumed closed or not active as illustrated in Figure 5-5

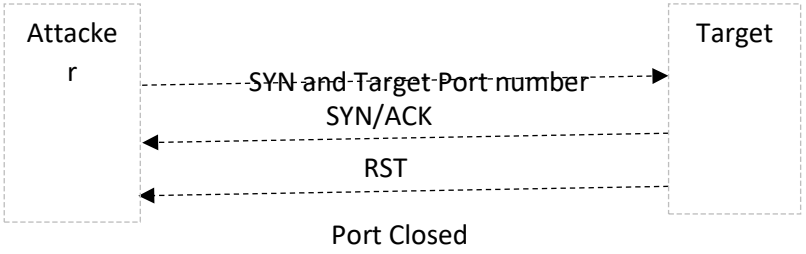


Figure 4-5 Incomplete SNY/ACK Connection from Target

Hping3 on the other hand is a packet generator and analyser for the TCP/IP protocol.

Brute Force attacks is another form of attack that is used to obtain private user information such as usernames, passwords and personal identifications. There are a number of tools available for brute force attacks like metasploit, Nmap scripts, Hydra to name but a few. For this attack, Kali Linux is used as the attacker is aiming its attacks at the FTP and SHH of the target system Metasploitable.

Infiltration attacks in this attack scenario a vulnerable application is exploited. The target system is sent a malicious document through email and using the metasploit framework a backdoor is opened and executed on the target system. The open backdoor allows the target to be attacked using IP sweep, full port scan and service enumerations using Nmap.

4.3.2 Preparing for the Simulation of attacks

Attack simulation as explained in earlier chapters is done on a localised virtual environment set up in the lab. Ubuntu platform is used as the host and virtual box for virtualisation. Due to the nature of the project, the attacks simulated are those that are unique to virtualisation as the main aim of the project is to design and implement a predictive model for these attacks. To carry out an attack therefore, a quick scan of the vulnerable or target system is done using an

Nmap command to help determine the open ports that are needed to carry out the various attacks. Take DoS attack for example, the TCP port 80, which is an HTTP port, is what is required for the attack.

```
Nmap 192.168.56.101
```

```
Starting Nmap 7.01 ( https://nmap.org ) at 2019-08-22 21:44 BST
Nmap scan report for 192.168.56.101
Host is up (0.00011s latency).
Not shown: 977 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown
MAC Address: 08:00:27:95:49:2B (Oracle VirtualBox virtual NIC)
```

Listing 4-5 Targeted Port Scan

4.3.3 Attack Simulation

There are a number of attacks unique to virtualisation that are highlighted here, but the most relevant and the ones demonstrated in this project are DoS attacks. Due to the nature of the project, it is important to note that only a few attacks like DoS, Infiltration, botnet, application and web attacks are discussed but only DoS attacks are simulated. The rest are beyond the scope of the project. Simulation of these attacks is carried out in a systematic form to help generate the data required for **explanatory data analysis (EDA**) that leads to predictive analytics and then to the design and implementation of the predictive model in question.

There are a number of tools (hping3, auxiliary synflood and slowloris to name but a few) available on Kali Linux to assist with DoS attacks. DoS attack simulation using auxiliary synflood is demonstrated in the section that follows.

1. DoS Using Auxiliary Synflood

To launch a DoS attack using auxiliary synflood: A synflood attack is a DoS attack used to send huge amounts of sync to consume all resources of the target system. To do this, the target system using the IP address is set as the host and the target port, which is port 80, in this case is set in metasploit as shown in the figure below.

```
msf > set RHOST 192.168.56.101\r
RHOST => 192.168.56.101
msf > set RPORT\r0
RPORT => 80
```

Listing 4-6 Preparing the Target Host and Port for Attack

To ensure that the target system and the port are set, a quick show option is done in auxiliary mode as shown in the Listing 5-7 below.

```
msf auxiliary(synflood) > show options\r

Module options (auxiliary/dos/tcp/synflood):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   INTERFACE                   no        The name of the interface
   NUM                         no        Number of SYNs to send (else unlimited)
   RHOST      192.168.56.101   yes       The target address
   RPORT      80               yes       The target port
   SHOST                       no        The spoofable source address (else rand
omizes)
   SNAPLEN    65535            yes       The number of bytes to capture
   SPORT                       no        The source port (else randomizes)
   TIMEOUT    500              yes       The number of seconds to wait for new d
ata

msf auxiliary(synflood) >
```

Listing 4-7 SynFlood attack Environment

Listing 5-7 shows that the target host represented by RHOST with the IP address 192.168.56.101 is active and the target port represented by RPORT with port 80, both confirmed with a "YES" is active and ready for attack. Once the confirmation was positive

DoS attacks were then launched using the methods highlighted above to generate data for the dataset. DoS attacks are achieved using SYN flood tools available in Kali Linux.

In order to generate enough and meaningful data for evaluation, an Intrusion detection system (IDS) together with an Intrusion prevention system (IPS) both available tools in Kali Linux were also used. The data generated mimics real life attacks, which target various IP addresses and ports. Due to lack of diversity in traffic-data and lack of volume representing attacks, several attacks (DoS, Brute Force, Web Application, Infiltration and Botnet) explained in the beginning of the chapter were simulated and results collected and stored. The final dataset was put together by combining several results of the experiments collected from the IDS and IPS.

4.4 Data Collection

Data collection is a systematic approach of gathering and measuring information collected from various sources like IDS and IPS and activity logs. Data collection enables relevant questions to be answered (Müller et al., 2016) for example the data collected in the simulation will allow for exploratory data analysis and big data analysis to be carried out which in turn will lead to building the prediction analytics in question to be answered. Data collection for this process involves analysing, testing and evaluating packets captured by Wireshark as well as logs of the intrusion detection and prevention systems with a focus on network anomaly detection. The objective of this process is to develop a systematic approach in generating a diverse dataset for attack simulation based on the available profiles that contain abstract representation of the simulations and behaviours of the network during simulation. Various simulation results, each with unique features, are combined to form a dataset. The dataset focuses on DDoS attack scenarios with one attacking system and two active target systems one of which has many vulnerabilities and the main target for the simulation of DDoS attacks. The generated dataset contains details of attacks for applications, protocols and lower level network entities. The features are distributions of packets sizes, the flow of packets, patterns of payloads and the

request time of packet distribution. Simulated protocols are HTTP, FTP and SHH with the majority of traffic being HTTP.

Table 6-1 below shows the simulated attacks, the target and attacker IP addresses.

Attacker IP	Target IP	Activity
192.168.56.103	192.168.56.101	Port Scan DDoS
192.168.56.103	192.168.56.102 192.168.56.105	Port Scan

Table 4-2 Simulated Attacks with Attacker and Target IP address

The attack simulation process, due to the limitation of resources, was carried out within a 24-hour period and this involved launching DDoS attacks at intervals and a continuous scan of various or available ports. Several datasets where generated although for demonstration purposes only results for a day's simulation are shown and used in the building of predictive models including time series modelling. A sample of the generated dataset exported into an Excel spreadsheet is shown in appendix i. The raw data collected is converted into a CSV file format that can be imported into an analytical tool. A sample of the csv data file is shown in Figure 6.3 below. The full dataset is shown in appendix ii.

Sample Data imported into an analytical tool

Relation: DDoSdataset

No.	1: Label	2: Source IP	3: Source Port	4: Destination IP	5: Destination Port	6: Protocol	7: Timestamp	8: Flow Duration	9: Total Fwd Packets	10: Total Backward Packets	11: Total Length of Fwd Packets	12: Total Length of Bwd Packets	13: Fwd Packet Length Max	14: Fwd Packet Length Min	15: Bwd Packet Length Max	16:
	Nominal	Nominal	Numeric	Nominal	Numeric	Numeric	Nominal	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	Numeric	
...	Benign	192.168....	38579.0	192.168.56.1...	53.0	17.0	29/10/2019...	165.0	2.0	2.0	88.0	188.0	44.0	44.0	94.0	
...	Benign	192.168....	41505.0	192.168.56.1...	53.0	17.0	29/10/2019...	217.0	2.0	2.0	102.0	224.0	51.0	51.0	112.0	
...	Benign	192.168....	55421.0	192.168.56.1...	53.0	17.0	29/10/2019...	186.0	2.0	2.0	88.0	188.0	44.0	44.0	94.0	
...	Benign	192.168....	16092.0	192.168.56.1...	53.0	17.0	29/10/2019...	195.0	2.0	2.0	88.0	188.0	44.0	44.0	94.0	
...	Benign	192.168....	55956.0	192.168.56.1...	88.0	17.0	29/10/2019...	1002.0	2.0	2.0	2944.0	578.0	1472.0	1472.0	289.0	
...	Benign	192.168....	58250.0	192.168.56.1...	88.0	6.0	29/10/2019...	107.0	1.0	4.0	0.0	12.0	0.0	0.0	6.0	
...	Benign	192.168....	1439.0	192.168.56.1...	53.0	17.0	29/10/2019...	361.0	2.0	2.0	80.0	180.0	40.0	40.0	90.0	
...	Benign	192.168....	17310.0	192.168.56.1...	53.0	17.0	29/10/2019...	182.0	2.0	2.0	64.0	158.0	32.0	32.0	79.0	
...	Benign	192.168....	10513.0	192.168.56.1...	53.0	17.0	29/10/2019...	199.0	2.0	2.0	102.0	224.0	51.0	51.0	112.0	
...	Benign	192.168....	44022.0	192.168.56.1...	53.0	17.0	29/10/2019...	180.0	2.0	2.0	64.0	96.0	32.0	32.0	48.0	
...	Benign	192.168....	33520.0	192.168.56.1...	53.0	17.0	29/10/2019...	207.0	2.0	2.0	64.0	96.0	32.0	32.0	48.0	
...	Benign	192.168....	64871.0	192.168.56.1...	53.0	17.0	29/10/2019...	219.0	2.0	2.0	102.0	224.0	51.0	51.0	112.0	
...	Benign	192.168....	61169.0	192.168.56.1...	53.0	17.0	29/10/2019...	213.0	2.0	2.0	88.0	188.0	44.0	44.0	94.0	
...	Benign	192.168....	30471.0	192.168.56.1...	53.0	17.0	29/10/2019...	256.0	2.0	2.0	64.0	158.0	32.0	32.0	79.0	
...	Benign	192.168....	49157.0	192.168.56.1...	53.0	17.0	29/10/2019...	186.0	2.0	2.0	88.0	188.0	44.0	44.0	94.0	
...	Benign	192.168....	60233.0	192.168.56.1...	53.0	17.0	29/10/2019...	270.0	2.0	2.0	64.0	158.0	32.0	32.0	79.0	
...	Benign	192.168....	4259.0	192.168.56.1...	53.0	17.0	29/10/2019...	239.0	2.0	2.0	64.0	158.0	32.0	32.0	79.0	
...	Benign	192.168....	36368.0	192.168.56.1...	88.0	17.0	29/10/2019...	475.0	2.0	2.0	356.0	358.0	178.0	178.0	179.0	
...	Benign	192.168....	34422.0	192.168.56.1...	53.0	17.0	29/10/2019...	283.0	2.0	2.0	76.0	176.0	38.0	38.0	88.0	
...	Benign	192.168....	43483.0	192.168.56.1...	53.0	17.0	29/10/2019...	220.0	2.0	2.0	138.0	238.0	69.0	69.0	119.0	
...	Benign	192.168....	50453.0	192.168.56.1...	88.0	17.0	29/10/2019...	753.0	2.0	2.0	504.0	2884.0	252.0	252.0	1442.0	
...	Benign	192.168....	64905.0	192.168.56.1...	53.0	17.0	29/10/2019...	184.0	2.0	2.0	88.0	188.0	44.0	44.0	94.0	
...	DDoS	192.168....	49790.0	192.168.56.1...	80.0	6.0	29/10/2019...	9650764.0	4.0	0.0	24.0	0.0	6.0	6.0	0.0	
...	DDoS	192.168....	49791.0	192.168.56.1...	80.0	6.0	29/10/2019...	82462.0	3.0	6.0	26.0	11607.0	20.0	0.0	5840.0	
...	DDoS	192.168....	49791.0	192.168.56.1...	80.0	6.0	29/10/2019...	9648776.0	4.0	0.0	24.0	0.0	6.0	6.0	0.0	
...	DDoS	192.168....	49792.0	192.168.56.1...	80.0	6.0	29/10/2019...	12364.0	3.0	5.0	26.0	11601.0	20.0	0.0	7300.0	
...	DDoS	192.168....	49792.0	192.168.56.1...	80.0	6.0	29/10/2019...	9633740.0	5.0	0.0	30.0	0.0	6.0	6.0	0.0	
...	DDoS	192.168....	49793.0	192.168.56.1...	80.0	6.0	29/10/2019...	13199.0	3.0	5.0	26.0	11601.0	20.0	0.0	5840.0	
...	DDoS	192.168....	49793.0	192.168.56.1...	80.0	6.0	29/10/2019...	9628804.0	4.0	0.0	24.0	0.0	6.0	6.0	0.0	
...	DDoS	192.168....	49794.0	192.168.56.1...	80.0	6.0	29/10/2019...	12814.0	3.0	6.0	26.0	11601.0	20.0	0.0	4380.0	
...	DDoS	192.168....	49794.0	192.168.56.1...	80.0	6.0	29/10/2019...	9629835.0	4.0	0.0	24.0	0.0	6.0	6.0	0.0	
...	DDoS	192.168....	49795.0	192.168.56.1...	80.0	6.0	29/10/2019...	12742.0	3.0	5.0	26.0	11601.0	20.0	0.0	5840.0	
...	DDoS	192.168....	49795.0	192.168.56.1...	80.0	6.0	29/10/2019...	9629928.0	4.0	0.0	24.0	0.0	6.0	6.0	0.0	
...	DDoS	192.168....	49796.0	192.168.56.1...	80.0	6.0	29/10/2019...	13154.0	3.0	4.0	26.0	11601.0	20.0	0.0	8760.0	
...	DDoS	192.168....	49796.0	192.168.56.1...	80.0	6.0	29/10/2019...	9627768.0	4.0	0.0	24.0	0.0	6.0	6.0	0.0	
...	DDoS	192.168....	49797.0	192.168.56.1...	80.0	6.0	29/10/2019...	13111.0	3.0	4.0	26.0	11601.0	20.0	0.0	8675.0	
...	DDoS	192.168....	49797.0	192.168.56.1...	80.0	6.0	29/10/2019...	9626765.0	5.0	0.0	30.0	0.0	6.0	6.0	0.0	
...	DDoS	192.168....	49798.0	192.168.56.1...	80.0	6.0	29/10/2019...	8772780.0	5.0	0.0	30.0	0.0	6.0	6.0	0.0	
...	DDoS	192.168....	49799.0	192.168.56.1...	80.0	6.0	29/10/2019...	854432.0	3.0	5.0	26.0	11601.0	20.0	0.0	5840.0	
...	DDoS	192.168....	49799.0	192.168.56.1...	80.0	6.0	29/10/2019...	8770794.0	4.0	0.0	24.0	0.0	6.0	6.0	0.0	
...	DDoS	192.168....	49800.0	192.168.56.1...	80.0	6.0	29/10/2019...	854435.0	3.0	5.0	26.0	11601.0	20.0	0.0	7300.0	
...	DDoS	192.168....	49800.0	192.168.56.1...	80.0	6.0	29/10/2019...	8773787.0	5.0	0.0	30.0	0.0	6.0	6.0	0.0	
...	DDoS	192.168....	49801.0	192.168.56.1...	80.0	6.0	29/10/2019...	854454.0	3.0	5.0	26.0	11601.0	20.0	0.0	5840.0	
...	DDoS	192.168....	49801.0	192.168.56.1...	80.0	6.0	29/10/2019...	8771788.0	5.0	0.0	30.0	0.0	6.0	6.0	0.0	
...	DDoS	192.168....	49802.0	192.168.56.1...	80.0	6.0	29/10/2019...	840118.0	3.0	6.0	26.0	11607.0	20.0	0.0	7300.0	
...	DDoS	192.168....	49802.0	192.168.56.1...	80.0	6.0	29/10/2019...	8770487.0	4.0	0.0	24.0	0.0	6.0	6.0	0.0	

Figure 4-6 Sample of the Dataset imported into Weka

The csv file in Figure 6-3 is imported into an analytical tool in this case WEKA and a sample of the snapshot of the imported data is shown in Figure 6-4 below. The imported dataset illustrated in figure 6-4 has 31 attributes with 159,958 instances. There are two nominal values or labels identified as DDoS and benign values. The DDoS values represent the attacks being simulated whereas the benign values represent all the other traffic going through the network. The two labels or attribute values are represented as red for DDoS and blue for benign values.

Figure 6-4 shows all the attributes together with all the instances of these attributes based on the activities of the class attributes.

Dataset imported in WEKA represented as a CSV file.

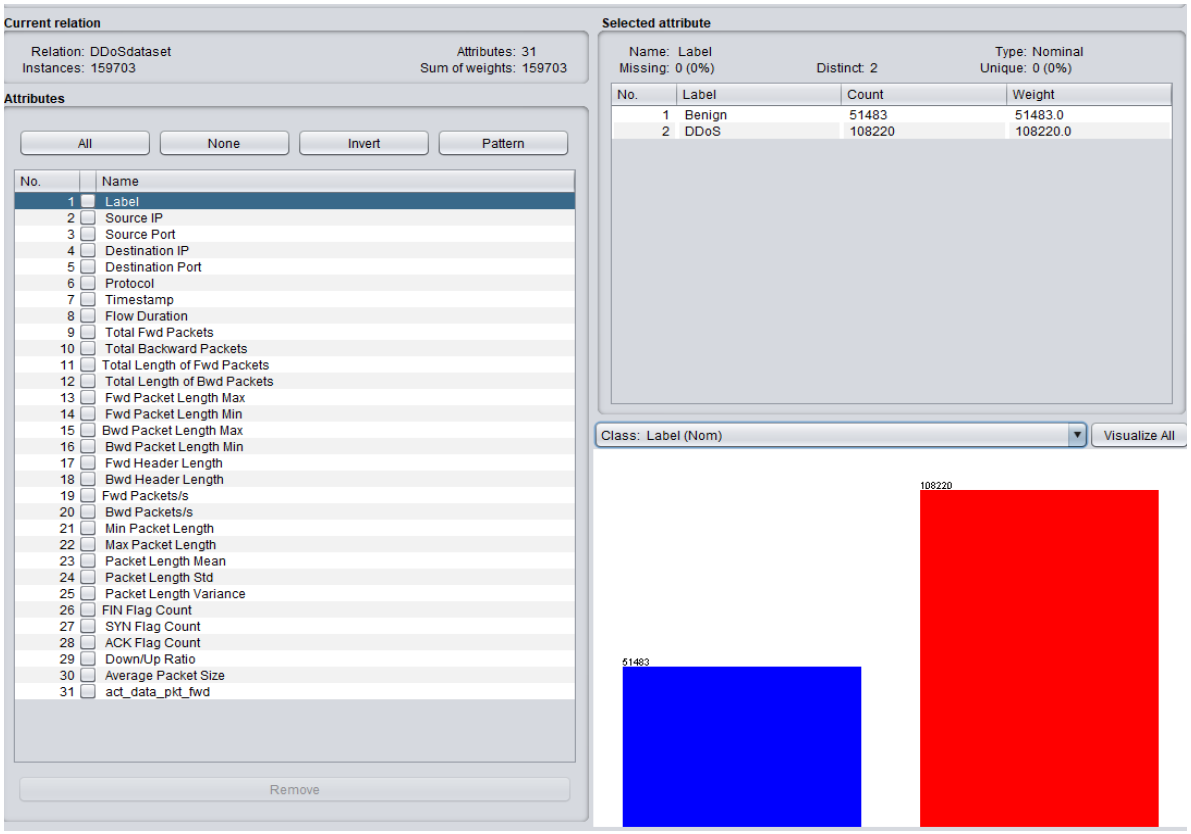


Figure 4-7 Overview of the Data loaded for Analysis

The first step once the data is read into the analytical tool is to understand how the data is represented in the dataset as well as to prepare the data for predictive analytics. It is important to know the type of features that are in the dataset. To do this, the data preparation process is carried out as described in the following section.

4.5 Data Preparation

The data preparation process is the most important stage of the EDA stage as it provides an insight of the data and lays a foundation for advanced analytics such as predictive analytics. The results in Figure 6-4 show that the attributes in the dataset have a combination of values and the class label has two values known as DDoS and benign. A review of the attributes shows that some of the attributes are numeric and others are nominal values with differing scales. The

class attribute is nominal and has two output values meaning that this is a two-class classification problem and not one as they would normally be. To prepare and understand the relationship and interaction of the attributes in the dataset features need to be selected and visualised.

4.5.1 Feature Selection

Feature selection (FS) in predictive analytics, refers to the process of identifying the most important variables that help in predicting the outcome of an activity. FS is a process of selecting a subset of the produced attributes in order to reduce the feature space according to the criterion. The goal of FS is to reduce the number of features to the relevant features required for predictive analytics. To select the features, we evaluate the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them using the best first method. This searches the space of attributes' subsets by greedy high climbing augmented with a backtracking facility. The forward direction search method and supervised learning was used with a subset evaluator, 192 subsets were selected including the locally predictive values as illustrated in Listing 6-1.

```
Evaluation mode:    evaluate on all training data


=== Attribute Selection on all input data ===

Search Method:
        Best first.
        Start set: no attributes
        Search direction: forward
        Stale search after 5 node expansions
        Total number of subsets evaluated: 191
        Merit of best subset found:    0.838

Attribute Subset Evaluator (supervised, Class (nominal): 1  Label):
        CFS Subset Evaluator
        Including locally predictive attributes

Selected attributes: 5,11,12 : 3
                      Destination Port
                     Total Length of Fwd Packets
                      Total Length of Bwd Packets
```

Listing 4-8 Feature Selection Criteria

The results showing attributes representing the selected features are illustrated in Listing 6-2 below.

```
=== Run information ===

Evaluator:    weka.attributeSelection.CfsSubsetEval -P 1 -E 1
Search:       weka.attributeSelection.BestFirst -D 1 -N 5
Relation:     DDoSdataset
Instances:    159703
Attributes:   31
               Label
               Source IP
               Source Port
               Destination IP
               Destination Port
               Protocol
               Timestamp
               Flow Duration
               Total Fwd Packets
               Total Backward Packets
              Total Length of Fwd Packets
               Total Length of Bwd Packets
               Fwd Packet Length Max
               Fwd Packet Length Min
              Bwd Packet Length Max
               Bwd Packet Length Min
               Fwd Header Length
               Bwd Header Length
              Fwd Packets/s
               Bwd Packets/s
               Min Packet Length
               Max Packet Length
               Packet Length Mean
               Packet Length Std
               Packet Length Variance
              FIN Flag Count
               SYN Flag Count
               ACK Flag Count
               Down/Up Ratio
               Average Packet Size
               act_data_pkt_fwd
Evaluation mode:    evaluate on all training data
```

Listing 4-9 Feature Selection

The selected attributes are collectively related to the attack that is simulated, for example, the source IP and source port represent the IP address where the attack is originating from, and the destination IP and the destination port represent the target IP address and port for the attack. The protocol is the TCP protocol under attack. The timestamp and flow duration is the time of the attack and the duration of the attack before the system shuts down also known as denial of service. Total forward and backwards packets shows the total number of packets from the attacking system to the target and the response back from the target to the attacker respectively. The total length of both the forward and backwards packets is the total number of the packets; and the maximum and minimum length of the packets represent the maximum number of packets forwarded to the target and back to the attacker respectively. The maximum and minimum packet length on the other hand represents the largest and the smallest packet size.

The packet length is the mean of packets, the packet length standard is the standard length of the packet and the packet length variance is the variance of the packet. Final flag count is the total number of packets forwarded as a whole. The SYN and ACK flag are the numbers of SYN floods forwarded to the target and the total number of acknowledgements of the attacks received back from the target to attacker. Down and up ration is the time the system stayed up during the attack against the time the target was denied services.

To understand this a brief description of what each attribute represents, is presented in table 6-2 below.

The selected features in Figure 6-2 are described in the Table 6-2 below.

Feature Name	Description
Label	Attacks simulated
Source IP	Attackers IP Address
Source Port	Attackers Port
Destination IP	Targets IP Address
Destination Port	Targets Vulnerable Port
Protocol	HTTP, FTP
Timestamp	Time of Attack
Flow Duration	Duration flow of Packets
Total Fwd Packets	Total number of packets forwarded to the target
Total Backward Packets	Total number of packets received back from target to attacker
Total Length of Fwd Packets	Length of Packets sent to target
Total Length of Bwd Packets	Length of Packets received back
Fwd Packet Length Max	Maximum number of Packets sent to target
Fwd Packet Length Min	Minimum number of Packets sent
Bwd Packet Length Max	Maximum number of Packets received
Bwd Packet Length Min	Minimum number of Packets received
Fwd Packets/s	Number of Packets sent per Second
Bwd Packets/s	Number of Packets received back per second
Min Packet Length	Size of the smallest packets sent
Max Packet Length	Size of the smallest packets received back
Packet Length Mean	Mean of packets
Packet Length Std	Standard length of Packets
Packet Length Variance	Variance of the packets
FIN Flag Count	Final flag count of the attacks
SYN Flag Count	Total amount of SYNFlood attacks forwarded
ACK Flag	Total amount of acknowledged SYN attacks
Down/Up Ration	Total time the target was down against the Up time
Average Packet Size	Average size of a forward/backward Packet
Act_data_pkt_fwd	Total acknowledged packets

Table 4-3 Selected Features Explained

Given the attributes selected in Listings 6-1 and 6-2, it is difficult to understand how the attributes are associated with the each other or even with the class labels without understanding how the data is represented in the dataset. To have a clear view of how the data is associated and how it relates with the class objects, further processing and visualisation of the data is necessary

4.5.2 Attack Detection Component

The main aim of the detection component is to find attacks and most possibly repel them. Detecting attacks normally depends on the number of appropriate actions as shown in the Figure 5-6 below. Prevention of attacks requires a well-selected combination of tools aimed at investigating both baiting and trapping threats. Both the host system and the trapping system are monitored and data generated through the monitoring system is carefully examined.

Attack Detection Process

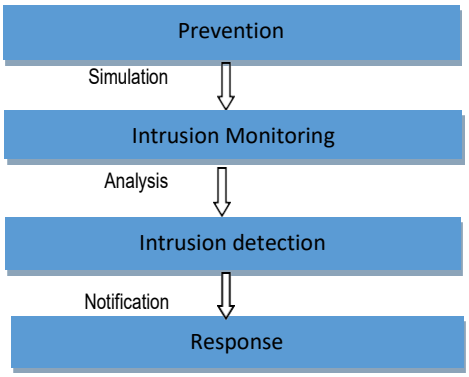


Figure 4-8 Detection Component

4.5.3 Attack detection infrastructure

The core of the detection is the analysis model that is responsible for detecting attacks. The model contains decision-making mechanisms for the attack. The model receives raw data from the sensors.

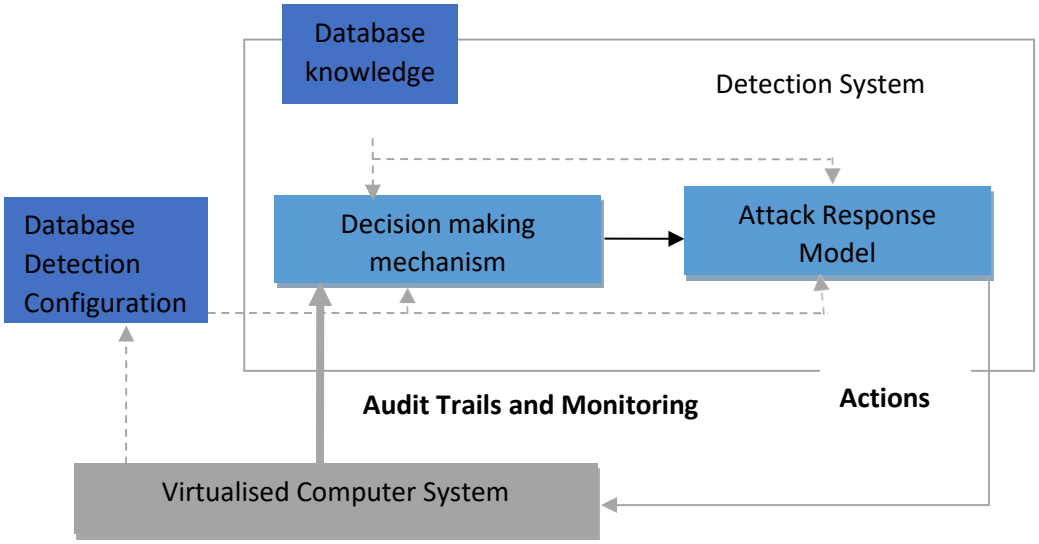


Figure 4-9 Attack Detection Component

Due to the nature of the work involved and the amount of data needed, attack simulations were carried out over a period in short simulation bursts. The dataset contains information about the kind of attack, source of the attack and target/destination IPs, attack and target ports and

information about the network traffic based on HTTP, FTP and SSH protocols. In order to generate a meaningful and usable dataset the following criteria were considered:

1. **Network configuration** by looking at the type of virtual network being used including the switches and the guest operating systems such as windows or Ubuntu.
2. **Traffic –** by profiling the target systems and the attackers in the network and using the monitoring system to monitor activities.
3. **Interaction –** by ensuring that there is complete interaction between the virtual machines or systems in the network
4. **Capture –** by ensuring that all necessary traffic was captured from the target system during attack simulation and dataset was correctly labelled

4.5.4 System Monitoring

Monitoring in this project involves the regular observation and recording of activities of the virtual environment both in its normal state and/or during attack simulation. A monitoring system is a piece of software that collects data from several sources, analyses the data and gives visualised results of the data (Zachar, Mehrotra et al., 2011). The main purpose of monitoring the activities of the virtual environment is to gain an insight of the system's behaviour before and after attack simulation and to gather information for analysis and feedback. There are a number of tools and techniques used to monitor events of information systems, detect attacks and provide identification to any problems or unauthorised use of the system. For this project, the Zabbix monitoring tool is used for monitoring the activities of the set-up virtual system, reporting and partial data collection through a MySQL database attached to the monitoring tool. Zabbix is an effective open source-monitoring tool for both networks and applications with three main components:

- The database server which provides data storage for all collected data and configurations
- The Zabbix server, which performs the actual monitoring, then collects data, and stores it onto the database.
- The webserver, which is a graphical user interface accessible through the HTTP.

Data collection from the tool can be done through the Zabbix agent installed on all systems being monitored. The Zabbix agent is a piece of software that runs on the monitored host that collects data and then sends it to the server through the Zabbix protocol.

In addition to the monitoring tool, IDS together with the IPS both tools of Kali Linux, are used for passive monitoring of traffic at packet level and prevention where problematic patterns that might lead to direct action are addressed respectively. Monitoring of the virtual environment is a continuous process that is integrated with the predictive analytics life cycle to determine whether the built and deployed predictive model continues to meet the set predictive goals of the model. A selected sample of the monitoring results for both the server and the implemented virtual environment before and after the attack simulation are illustrated in Figures 5-8 and 5-9 for activities of the server before the attack and Figures 5-10 all the way to Figure 5-17 for activities after attack simulation for both the server and the virtual machines of the simulation environment.

Monitoring results before attacks

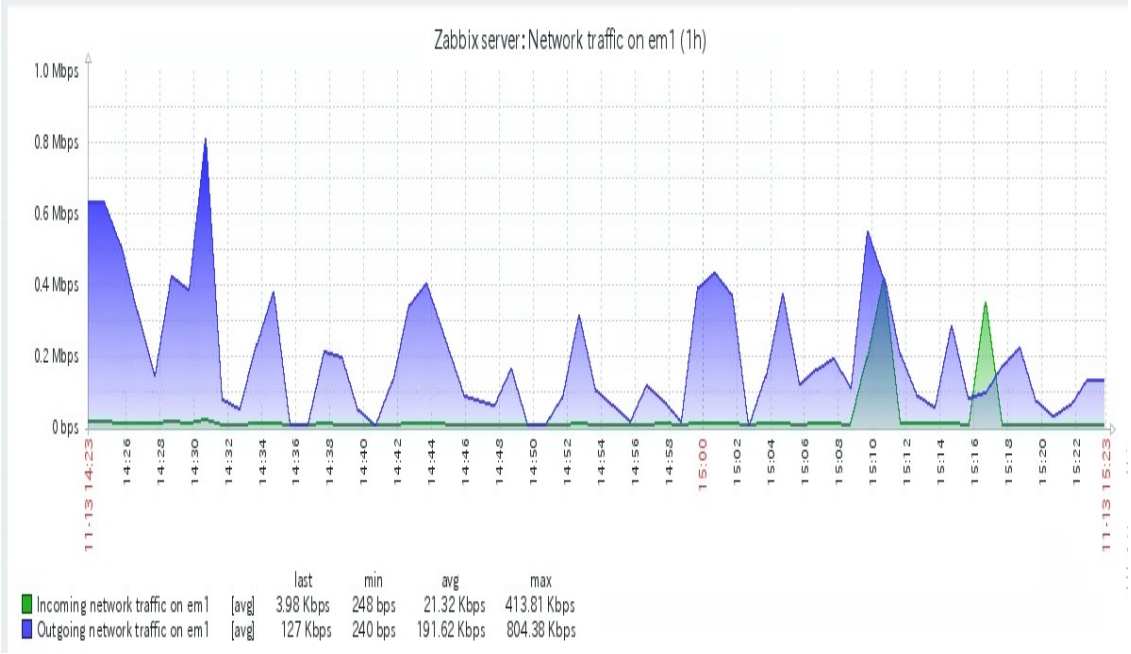


Figure 4-10 Network Traffic Results before attacks

Figure 5-8 shows uninterrupted network traffic flow between the server and the monitored network system and Figure 5-9 illustrates the internal processes of the server before attacks.

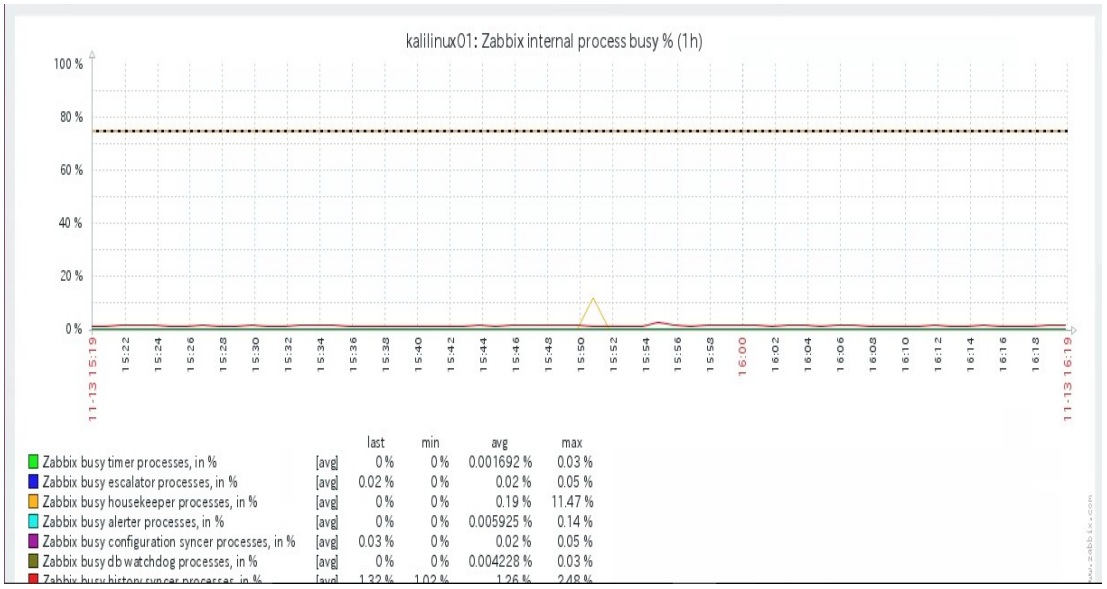


Figure 4-11 Attacker internal processes results before attacks

Monitoring Results after attacks

The results for both the server and the monitored virtual environment after attack simulation are illustrated in the graphs below.

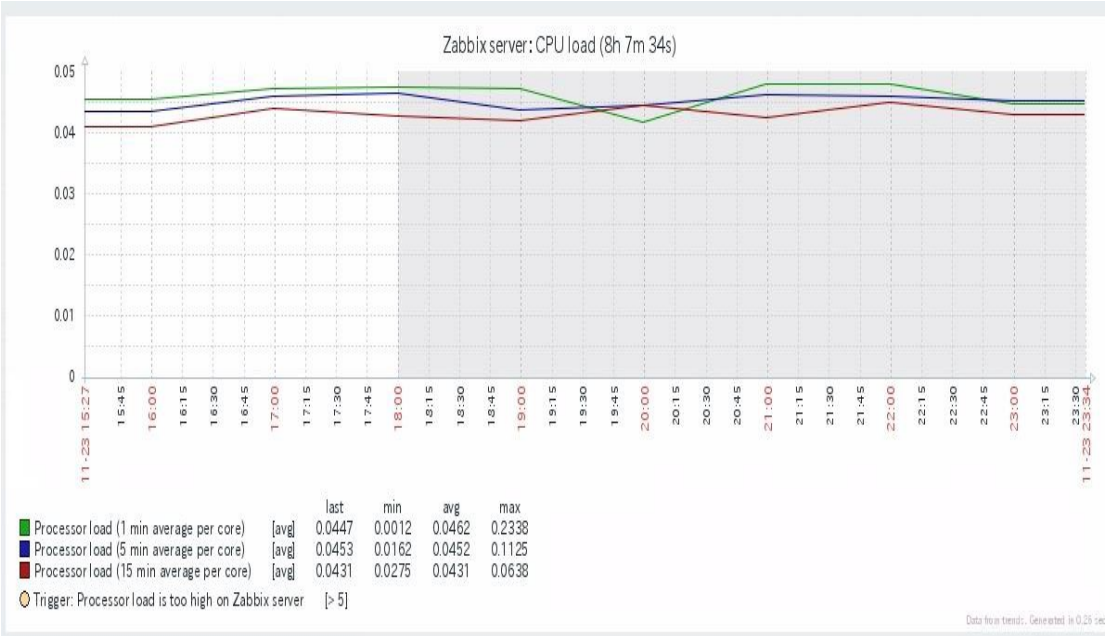


Figure 4-12 Server CPU load results

Figure 5-10 illustrates the results of the server processor load after the attacks were simulated. The results show the flow of processes at different time intervals to show how the CPU load changes during the attack of the system. Figure 5-11 on the other hand illustrates the data gathering process of the server during the attack simulation. The results show that data is being collected from various internal activities of the system until the system reaches an unreachable state causing DoS.

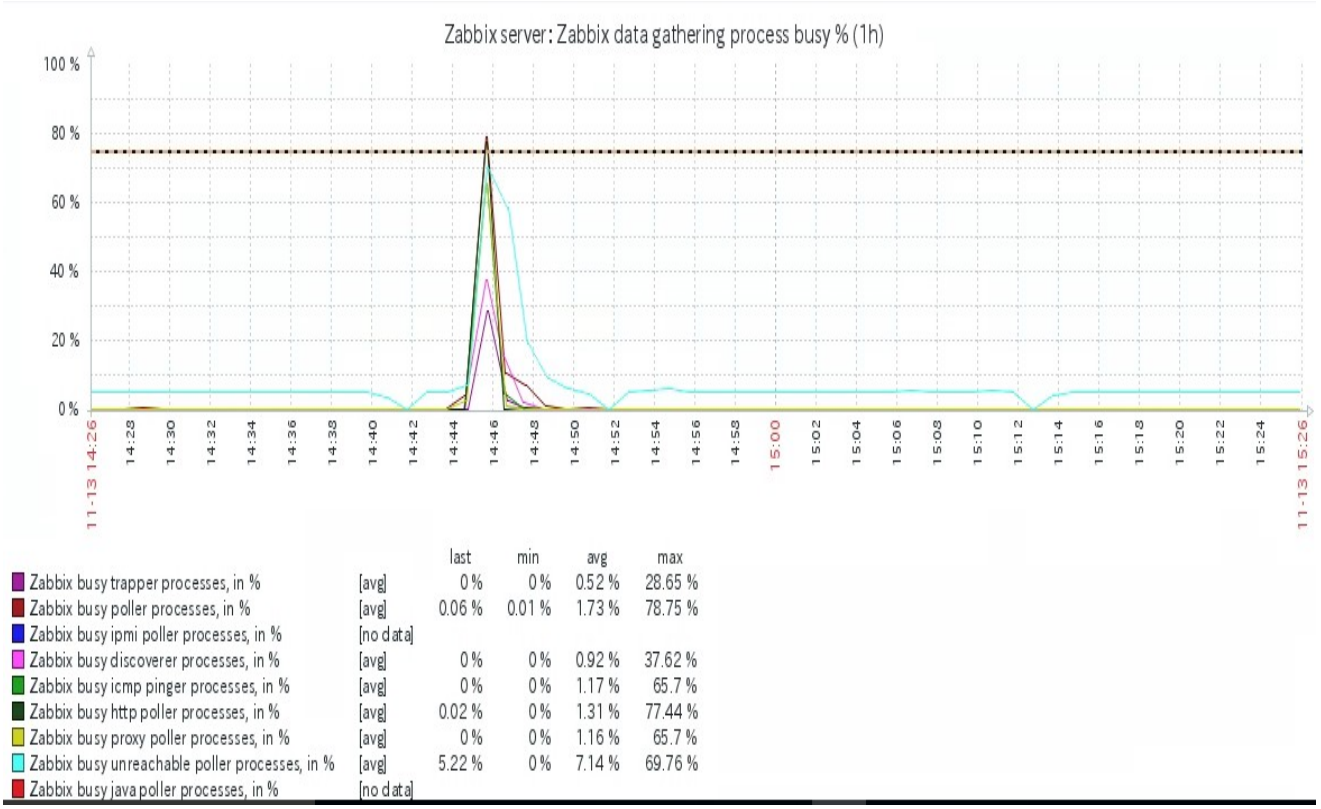


Figure 4-13 Server data gathering after attacks

Figure 5-12 illustrates the flow of both the incoming and outgoing traffic of the monitored virtual system after the attack is launched. The results show that the network is terminated causing an interruption in the network service represented by the gap in the graph. The network traffic continues after the attack is eradicated as illustrated in the same graph.

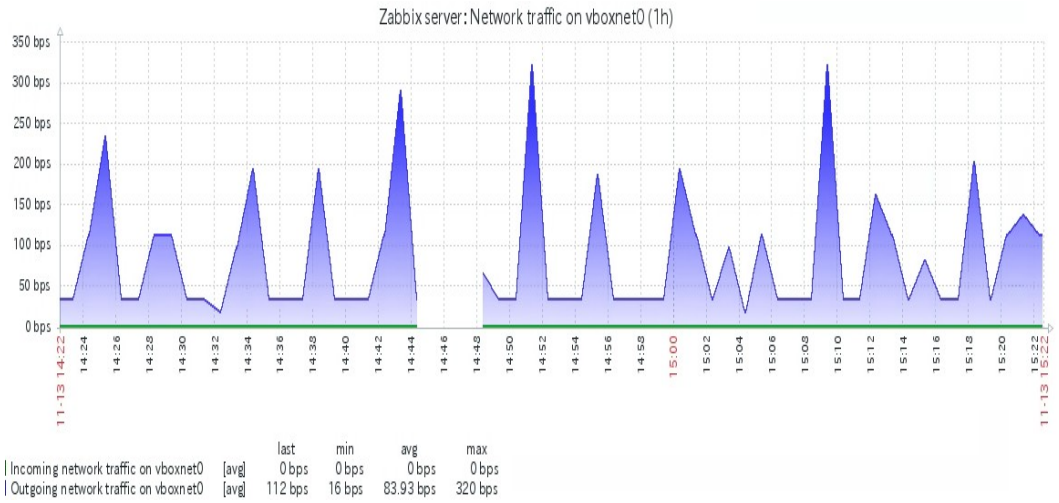


Figure 4-14 Server Network Traffic after attacks

Data gathering from the attacked virtual machine is illustrated in Figure 5-13 which shows the discoverer process reaching maximum during the attack. The results shows both the http and the poller processes at almost maximum which triggers the attack just before the DoS attack is fully achieved.

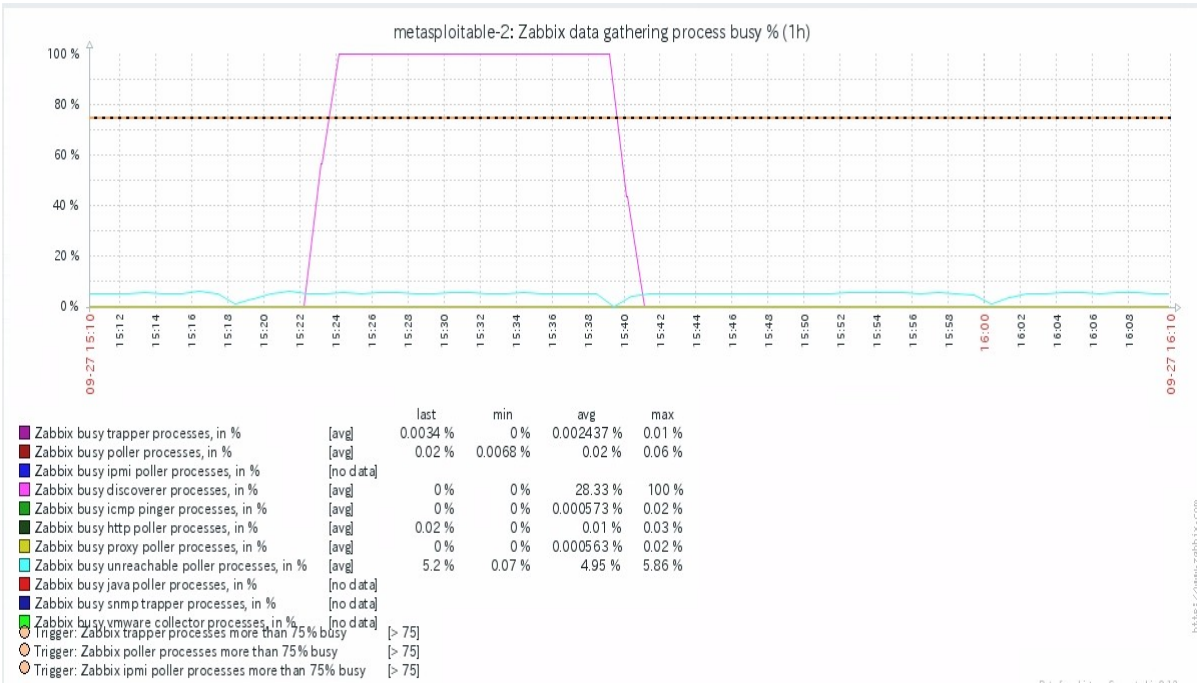


Figure 4-15 Target 1 data gathering process

Figure 5-14 illustrates the results of the attacked virtual systems self-monitoring processes during the attack and triggers the attack when the packets received reaches 75%. Figure 5-15 of the other hand shows the results of the http poller processes as busy during the attack

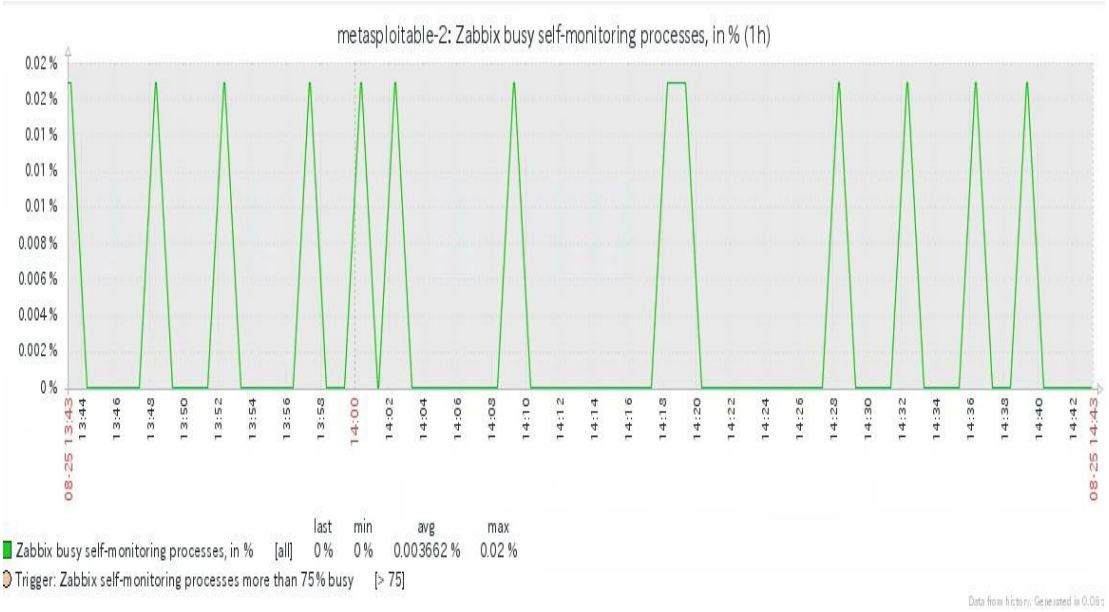


Figure 4-16 Target 1 self-monitoring process

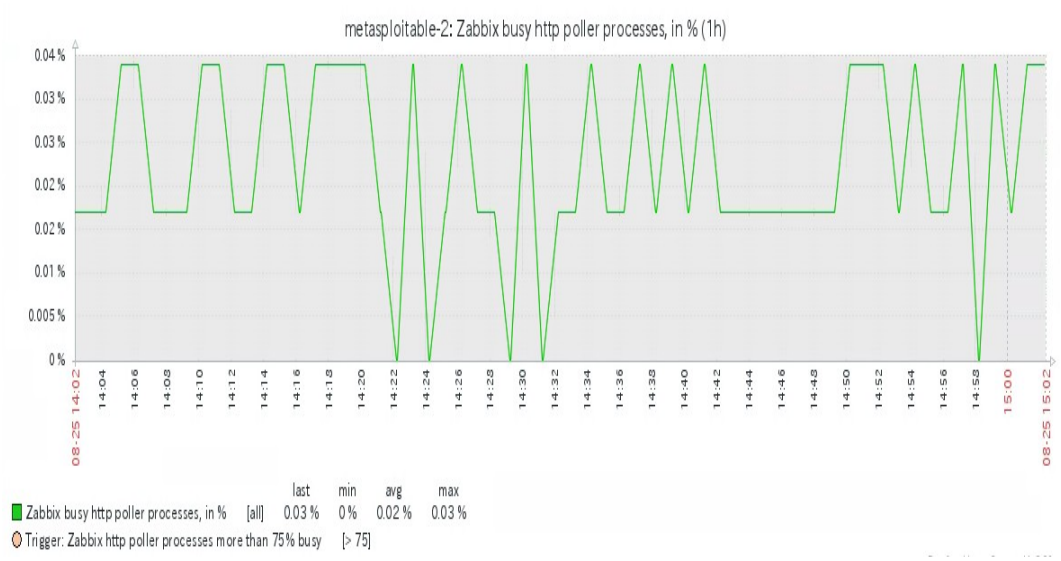


Figure 4-17 Target 1 Http Poller Process

Figure 5-16 illustrates results of the packets sent by the attacking system. The results show a pick in the packets when maximum packets are processed or sent to the target system causing DoS on the target system. Figure 5-17 on the other hand illustrates the internal processes of the attacking system. The results show the time set for the attack, the escalation of the attacks, the intrusion detection of the attacks, the data collection of the activities of the system together with the history of the attacks in form of logs.

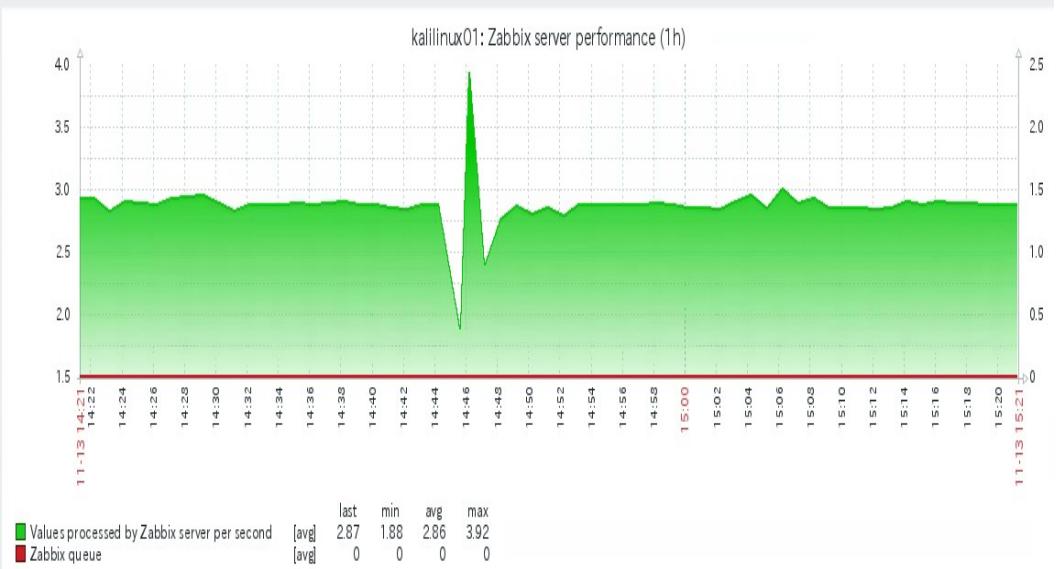


Figure 4-18 Server Performance after attacks

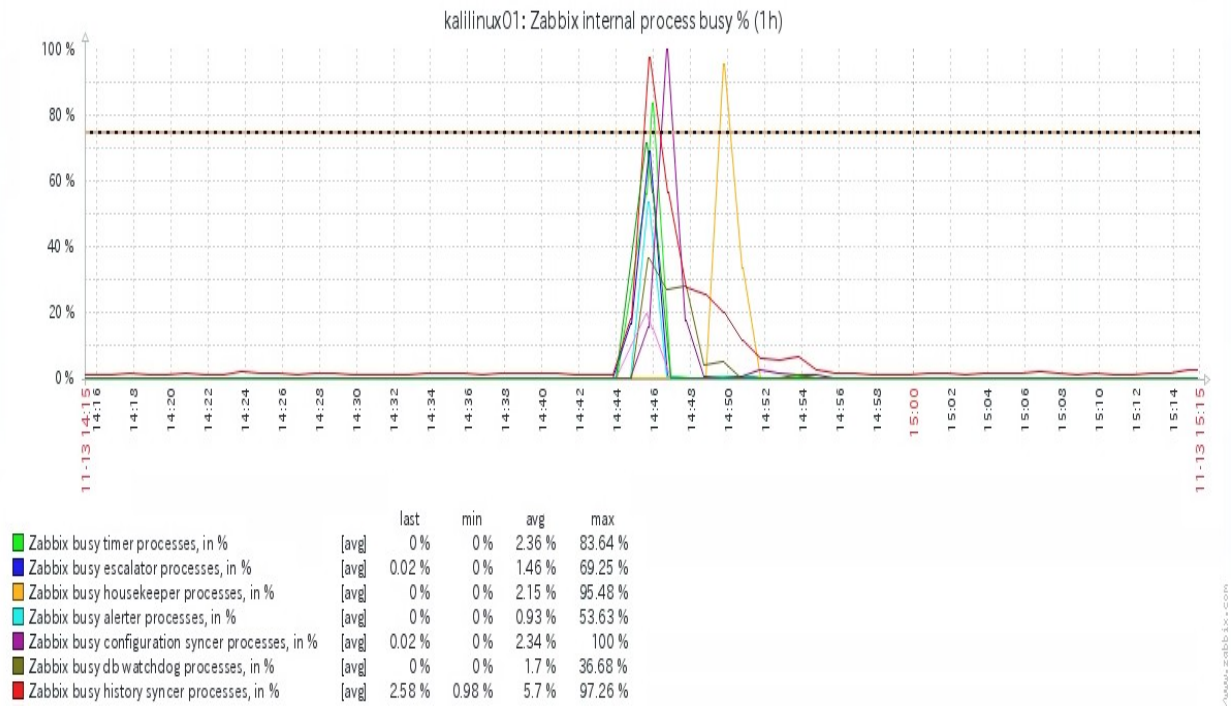


Figure 4-19 Attacker internal processes after attack

4.5.5 Summary

The chapter provides an introduction of the attack simulation process and provides a simulation attack framework with a decomposed explanation of the framework. This chapter has demonstrated the process for simulating attacks through the experimental results as well as the process for generating the required dataset. The dataset generated highlights the attributes that are relevant to the data analysis and classification processes for building and validating the predictive analytics model in question. The chapter identifies and describes the simulation tools used for data collection required for the predictive analytics Modelling (PAM). The types of attacks being simulated together with the results obtained from the simulation are also presented in this chapter through the various screen shots of the attacks in motion. The data of the simulation activities that has been produced from the intrusion detection system (IDS) and intrusion preventive system (IPS) both tools in Kali Linux, is imported into a spreadsheet. Since monitoring has been an integral part of the process, the results from the monitoring tool are

presented in this chapter, in the form of graphs for both before and after the attack simulation.. The data collected during the attack simulation process is manipulated by exploring, analysing and visualising it in the opening part of the predictive analytics application process in the explanatory data analysis of chapter 6, which is the initial stage for predictive analytics and modelling. The end of the chapter also briefly describes the monitoring process and presents the results through graphs of monitoring tools.

The satisfactory completion of the data collection process leads to the application of predictive analytics as presented in chapter 6 that follows. In chapter 6 an explanation of prediction analytics and the processes involved are discussed and presented. The chapter also provides an exploratory process that is used to prepare and pre-process the collected dataset into useful information for predictive modelling.

5. Predictive Analytics Modelling

5.1 Introduction

Predictive analytics as explained in chapter 3 is an analytical process that encompasses statistical techniques from data mining, machine learning and artificial intelligence to predict what will happen in the future. Prediction in machine language refers to the output of an algorithm after it has been trained on a historical dataset and applied to new data when trying to forecast the likelihood of a particular outcome (Kelleher et al., 2015) such as future attacks in this case. The algorithm generates probable values for unknown variables for each item in the new data, which can be identified as the most likely values (Kelleher et al., 2015). There are five common predictive analytics models

1. Classification models – These are the simplest of several predictive analytics models. These types of models separate data into categories based on what they learn from historical data. Their breadth of possibilities and the ease of use by which they can be trained with new data means that they can be applied to many cases in various fields including information technology.
2. Clustering Models – These types of models focus on identifying groups of similar records and labels the records according to the group to which they belong using unsupervised learning algorithms. There are different types of clustering algorithms (K-Means, Mean-Shifting, Expectant-Maximization using Gaussian-Mixture-Models (GMM) and Agglomerative hierarchical clustering to name but a few) that are available.
3. Forecast Models – The most used predictive analytics models that deal with metric value prediction by estimating numeric values for new data based on historical data.
4. Outlier Models – These types of models deal with anomalous data entries within the dataset.

5. Time Series Models – These comprise of a sequence of data points captured using time as an input parameter. It uses the last date of the historical data to predict the outcome of the next using the same metric.

The models described above can all be used in the building of the predictive model in question and can be deployed using different predictive modelling mark-up language (PMML) as illustrated in the stages of chapter 6. Due to the work involved, only classification, clustering, forecast and time series models are built and illustrated. As the main aim of the project is to use predictive analytics as a security management tool in virtualised systems, a number of steps leading up to this stage have been taken as highlighted in chapter 5 and 6. The application of predictive analytics in this project follows through the systematic process of

- Project Definition discussed in the research aims and projects in chapter 3. This section defines the project outcomes, deliverables, scope and the expected overall outcome of the project.
- Data Collection demonstrated in chapter 5, which provides the basis for data mining for EDA and predictive analytics is also covered. Data collection is achieved by continuous simulation of attacks to generate enough data for EDA, predictive analytics and modelling.
- Data Analysis, which is the preliminary process of inspecting, cleaning and data modelling, covered in EDA in the beginning of chapter 6. The objective of EDA is to help in discovering useful information about the generated dataset that leads to the successful conclusion of this project, which is predictive analytics. The collected data is pre-processed, visualised and prepared for predictive modelling.
- Statistical analysis carries on from EDA as it enables the validation of assumptions, hypothesis and then allows to test them using statistical models such as clustering.

- Modelling process that provides the ability to build accurate predictive models such as regression and time series models that are validated and deployed to help in predicting attacks in virtualised systems.
- Predictive Model Deployment provides the option to deploy the created model into the virtual environment for attack prediction.

To achieve PA, a simple process illustrated in Figure 6-1 below is followed in the EDA stage of PA.

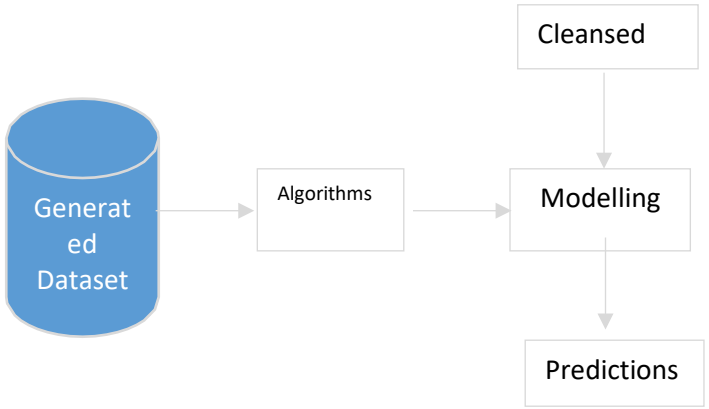


Figure 5-1 Implemented Predictive Analytics Process

The first and most important stage of PM is to understand the data structure of the dataset intended for building the model and then prepare it for predictive analysis. To do this, data needs to be cleaned in order to correctly deal with the missing values and to remove redundant data. This cleansing process helps with making correct predictions. Once redundant data is removed, data needs to be transformed, inspected and modelled into useful information by evaluating all the variables available by understanding the construction, the population and the relationships of the variables to form a hypothesis of the data. A model is then built by mapping the data into various machine-learning algorithms that can help in solving a series of problems such as regression, clustering and classification problems. The first thing to do is to split the data into three views or datasets, which is random data for modelling, training data used for

training the model and then a testing dataset for testing the built model. The predictive models are built using the training dataset, then validated and tested using the testing data. This process involves filtering of the response variables to help improve the performance of the model and predictive efficiency.

To efficiently and correctly design and deploy a PAM, a simple predictive analytics logic illustrated in Figure 6-2 is followed. The aim of the PAM in this case is to determine whether an attack will occur, using the collected matrices of the data provided.

Predictive Analytics Logic

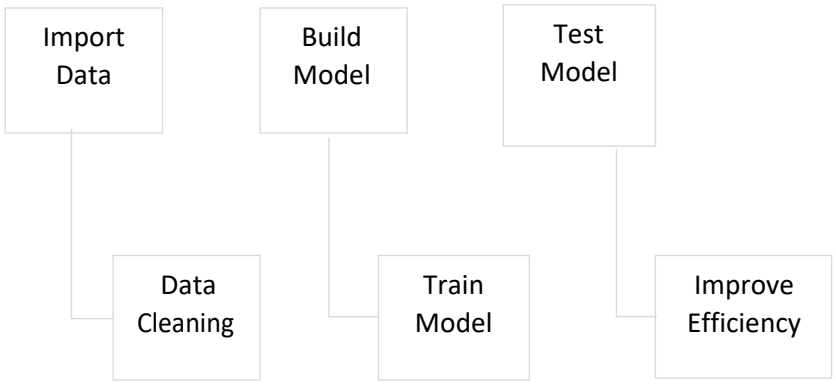


Figure 5-2 Predictive Analytics Logic

Predictive analytics can be achieved using various languages or tools and techniques; for this part of the project, a combination of WEKA and an R-console (an R version embedded in WEKA) are the choice of analytical tools that are used. R was chosen due to its simplicity and availability of in-built functions for data manipulation, statistical analysis, visualisation and the number of packages readily available for use, together with the ability to use R in conjunction with WEKA especially during modelling. These have been the main motivating options for the project.

For predictive modelling, the predictive analytics logic in Figure 6-2 is used to achieve the predictive value of the predictive model built. The first step is to import data into WEKA to

prepare it for analysis, this process leads to the model being built, trained, tested and improvements applied where necessary until it reaches its predictive value.

The imported dataset is explored analysed and prepared for predictive modelling using the explanatory data analytics process explained in the section that follows.

5.2 Explanatory Data Analysis

To ensure or understand the data, Explanatory Data Analytics (EDA) is carried out. EDA is an approach used for analysing datasets to summarise their characteristics using visualisation methods (Cox, 2017). EDA is an important process in data mining that provides an insight of the data collected in preparation for predictive analytics (Cox, 2017, Watts, 2016). To create an effective and efficient predictive analytics model for the management of virtualisation security, the right data needs to be collected. EDA in this case, helps to visualise the collected attributes as well as how they associate with one other in the dataset. There are several methods of visualising simple summaries of datasets, for example, a five number summary consisting of the smallest and largest data value, the median and the first and third quantiles can be visualised as a drawing where each number corresponds to a constituent like the altitude of a box (Kennedy and Allen, 2017). These simple methods are very useful for summarising low dimensional datasets. If the dimensionality increases, the ability to visualise interdimensional relations decreases. The main objective of the EDA phase in this project is to help

- Suggest the hypothesis about the causes of the observed phenomena
- Assess the given assumption on which statistical inference will be based
- Support the selection of appropriate predictive analytics tools and techniques
- Provide a basis for more data collection if needed through more simulation of attacks.

The emphasis in the EDA process will be on methods that illustrate structures in the given pre-specified dataset. Most EDA techniques are represented by box plots, scatter plots and histograms.

5.2.1 Data Pre-processing and Visualisation

The simplest method to visualise a dataset is to plot an outcome of each attribute in a two dimensional graph, in which the dimensions are enumerated on the x-axis with the corresponding value on the y-axis. Another method used to visualise data is to use a scatter plot where two dimensions of the data are selected as the location of an icon and the rest as the properties of the icon.

To discover the distribution of attributes and the association with each other, visual presentations of the dataset are illustrated in Figures 6-5 and 6-7 respectively.

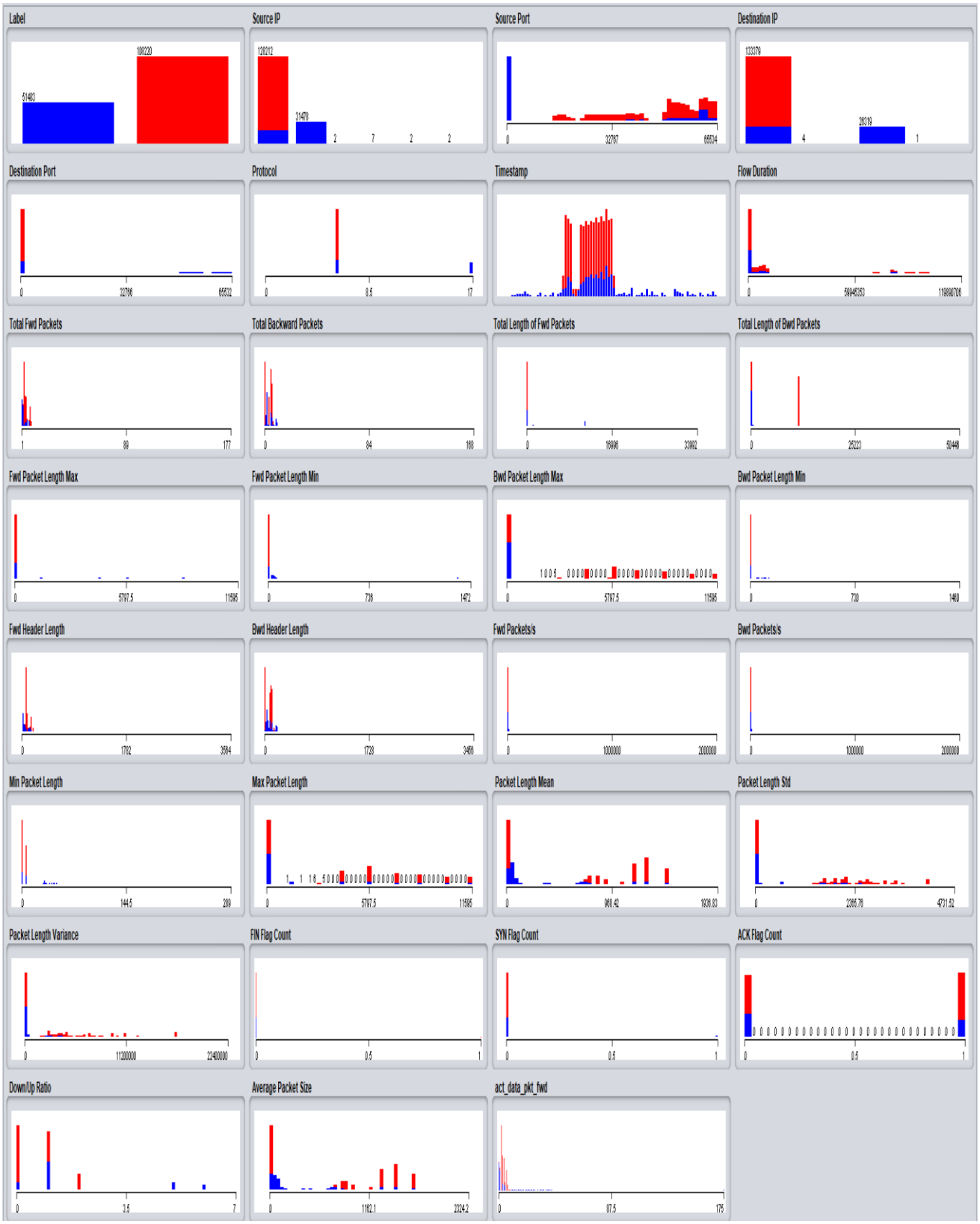


Figure 5-3 Attribute Distribution

Some attributes in Figure 6-5 such as source port, time stamp and flow duration have a Gaussian-like distribution, which could yield good results for Logistic Regression and Naïve Bayes. An example of which, is shown in Figure 6-6 below

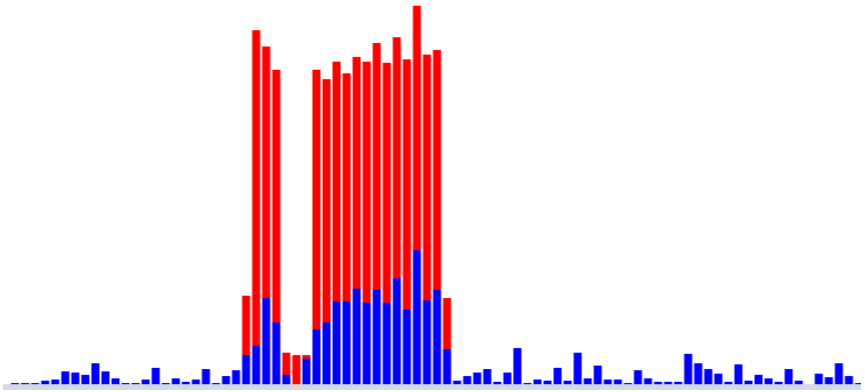


Figure 5-4 Gaussian representation of the Time Stamp attribute

Figure 6-7 below illustrates the average number of packets from the attacker to the target system. The graph shows the packets highlighted on the x-axis and the attacking IP on the y-axis.

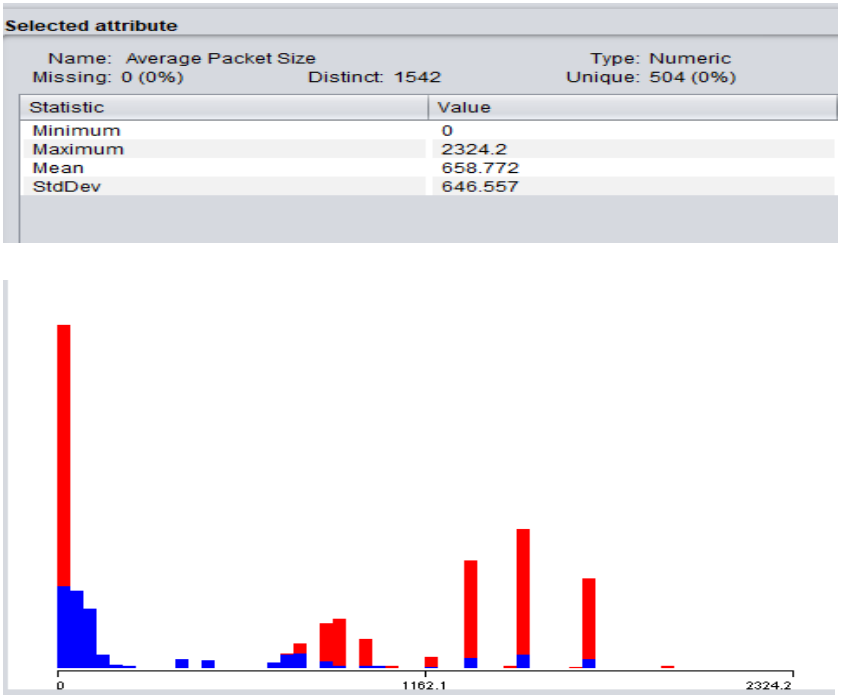


Figure 5-5 Average Packet Size per attack

Figure 6-8 shows the box plot of the total number of forwarded packets

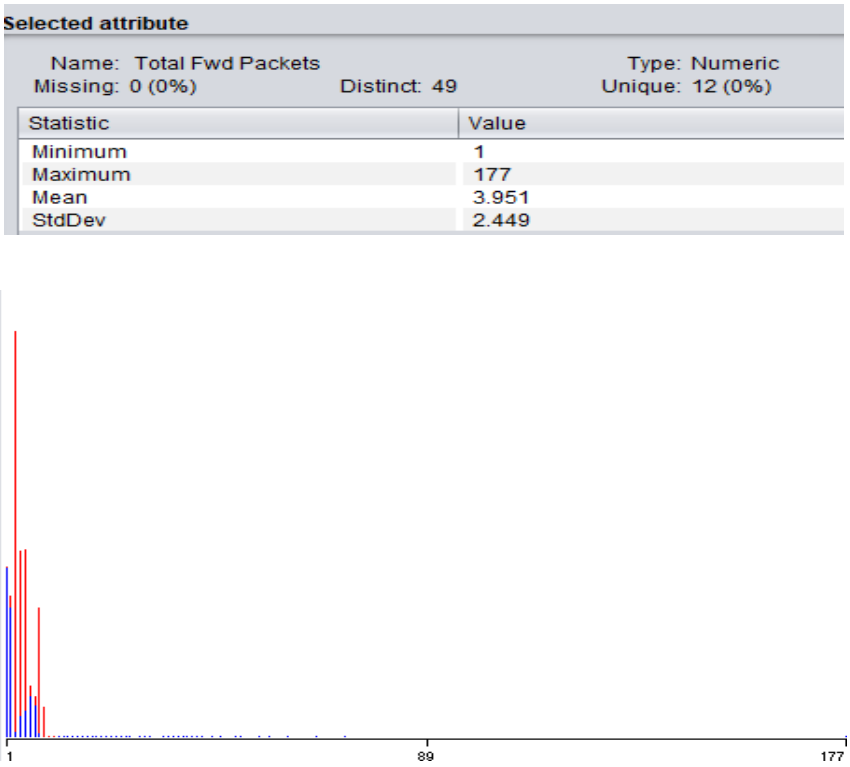


Figure 5-6 Total number of Packets to Source IP

There is also a lot of overlapping between the classes across the attribute values that shows a class imbalance. To deal with this problem multiple views of the data are prepared to help with the evaluation of the algorithms. The data views are created from the original dataset and are saved as individual files for use at a later stage. The association of the attribute results in Figure 6-5 are presented in the scatter plot in the Figure 6-19 below. The scatter plot shows howeach attribute is associated with the other attributes in the dataset and because of the size of the graph only a snapshot of the results is presented.

Scatter Plot of the attributes' association.

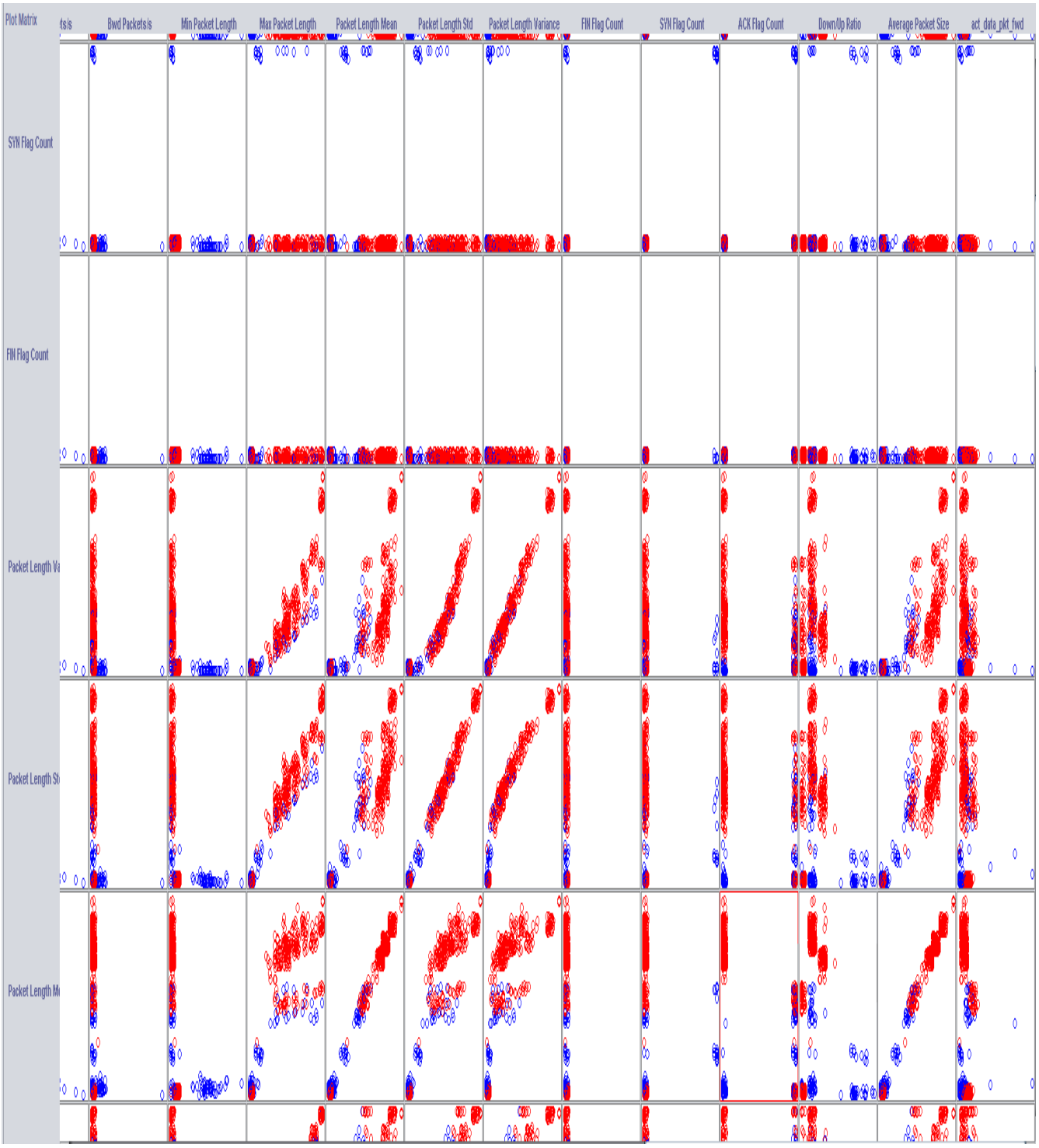


Figure 5-7 Scatter Plot for Attribute Association

A visual analysis review of the results in Figure 6-9 above yields good results as it shows that some of the attributes in the dataset are linearly related, which provides us with good knowledge of what attributes to use with what algorithms when building the predictive model in question. An example of these linear relationships is presented in the results of Figures 6-

10, which shows the maximum forward packet length on the x-axis and the backward packet lengths on the y-axis. Figure 6-11 shows the relation between maximum packets length and Packet Length Standard and figure 6-13 shows the relationship between packet length mean and average packet size.

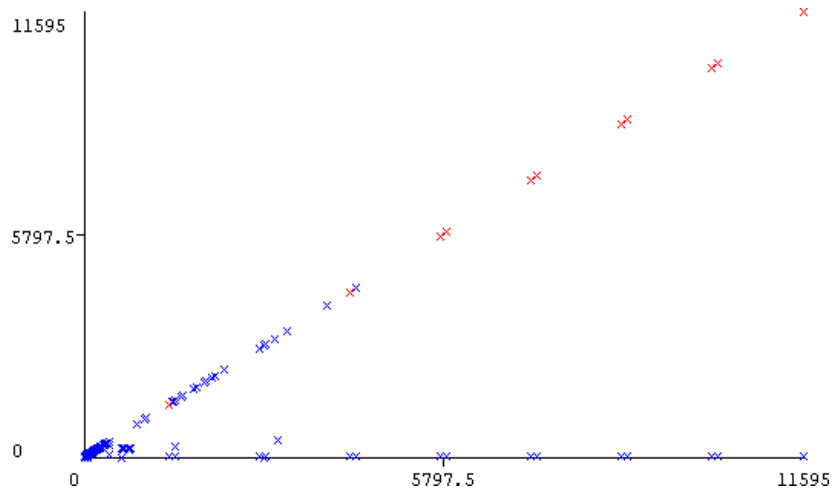


Figure 5-8 Maximum forward packets and backwards Packets

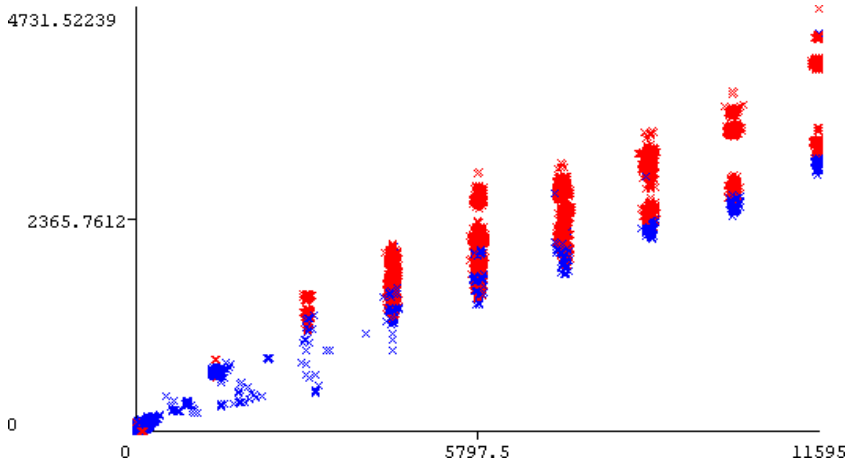


Figure 5-9 Maximum Packets Length and Packet Length Standard

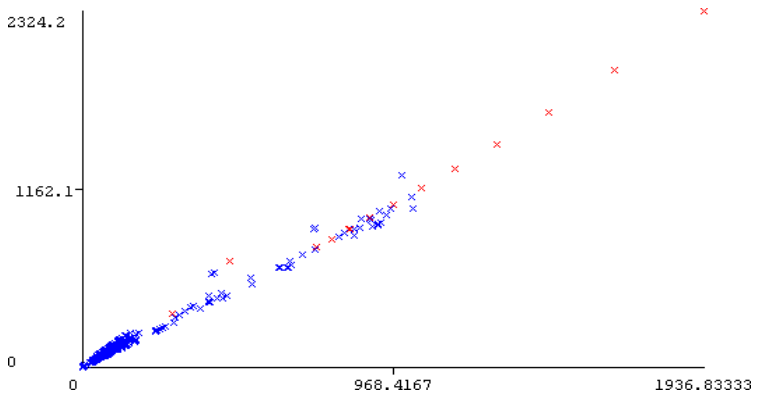


Figure 5-10 Packet length Mean and Average Packet Size

To help understand the association results visualised and represented in Figure 6-9 above, the attributes are analysed and processed using various classification methods and algorithms. At this stage, only simple classification algorithms are used.

5.2.2 Classification

In order to use the produced dataset and the selected features effectively and efficiently, data needs to be organised into different categories using the classification process. To classify the data, we use the ZeroR and the J48 algorithms based on the full training data classifier. This was done to select the best fitting model for the project. The ZeroR algorithm is a classifier that predicts the mean when applied to numeric classes and the mode if the class attributes are nominal as in our case. This provides an option of classifying any type of attribute without converting it to the suitable type for that particular attribute. The first classification scenario is based on full training dataset. The algorithm predicts the class value as being DoS with 68% accurately classified instances and 32% incorrectly classified instances. The results of the classification are shown in Listing 6-3 below.

Classification model on full training Dataset using ZeroR algorithm

```
=== Classifier model (full training set) ===

ZeroR predicts class value: DDoS

Time taken to build model: 0.02 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      108220               67.7633 %
Incorrectly Classified Instances     51483               32.2367 %
Kappa statistic                          0
Mean absolute error                      0.4369
Root mean squared error                  0.4674
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances           159703

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.322     Benign
                 1.000    1.000    0.678      1.000    0.808      ?        0.500     0.678     DDoS
Weighted Avg.    0.678    0.678    ?          0.678    ?          ?        0.500     0.563

=== Confusion Matrix ===

      a      b   <-- classified as
      0  51483 |      a = Benign
      0 108220 |      b = DDoS
```

Listing 5-1 Classification using ZeroR algorithm

The results in Listing 6-3 are illustrated using a threshold Curve in Figure 6-13, based on the DoS attacks with the x-axis showing the false positive rate and the y-axis representing the true positive rate.

Plot area under ROC = 0.5

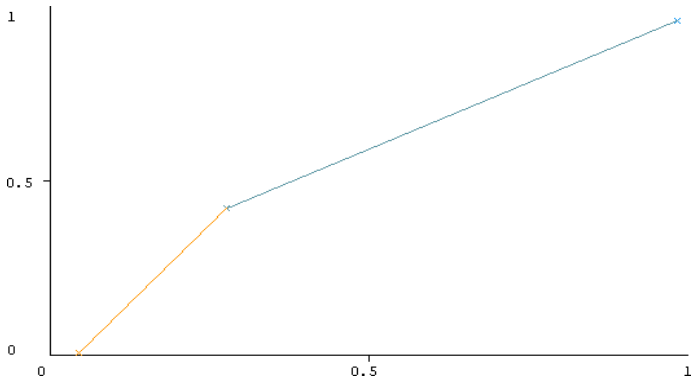


Figure 5-11 Classification using Cross validation under ROC for DDoS

To counter check the results in Listing 6-3 a cross validation technique is in the second scenario using a OneR algorithm that uses the minimum error attribute for prediction and also decentralises numeric attributes. The classification on this occasion uses cross validation methods with 10 folds, which splits the data into 10 parts based on the full training dataset. The first 9 folds are used to train the algorithm and the 10th fold is used to assess the algorithm. Using the 10 fold cross validation gives the algorithm an opportunity to make a prediction for each instance of the classified attributes of the dataset with different training folds. The following results present the summary of the predictions.

Classification model on full training Dataset using OneR algorithm

```
=== Classifier model (full training set) ===

 Total Length of Bwd Packets:
        < 2.0    -> DDoS
        < 1335.5         -> Benign
        < 1407.5         -> DDoS
        < 11513.5        -> Benign
        < 11947.5        -> DDoS
        >= 11947.5       -> Benign
(158939/159703 instances correct)


Time taken to build model: 0.83 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.23 seconds

=== Summary ===

Correctly Classified Instances      158939               99.5216 %
Incorrectly Classified Instances       764                0.4784 %
Kappa statistic                          0.989
Mean absolute error                      0.0048
Root mean squared error                  0.0692
Relative absolute error                  1.095  %
Root relative squared error             14.7985 %
Total Number of Instances           159703

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.987    0.001    0.998      0.987    0.993      0.989    0.993     0.989     Benign
                 0.999    0.013    0.994      0.999    0.996      0.989    0.993     0.994     DDoS
Weighted Avg.    0.995    0.009    0.995      0.995    0.995      0.989    0.993     0.992

=== Confusion Matrix ===

      a      b   <-- classified as
  50808    675 |      a = Benign
     89 108131 |      b = DDoS
```

Listing 5-2 Classification using OneR algorithm

The results of the OneR algorithm shown in Listing 6-4 above shows the classification accuracy of 99%. The results also show the confusion matrix with actual classes compared to the predicted classes. These results are illustrated in Figure 6-14.

Threshold Curve of the results in listing 6-4

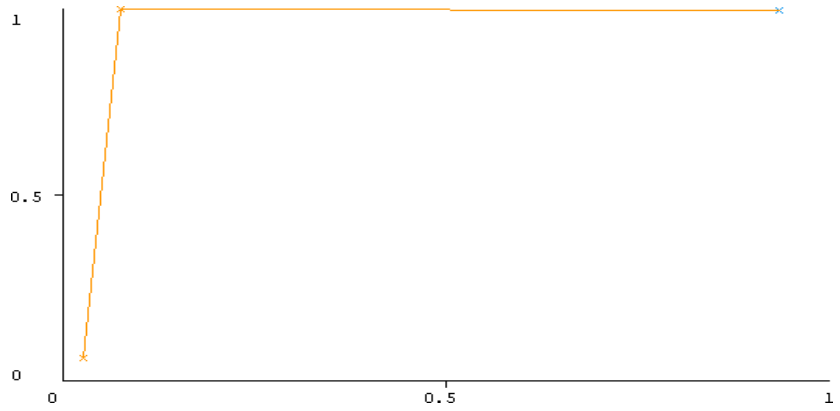


Figure 5-12 Threshold Curve of OneR algorithm using Cross Validation

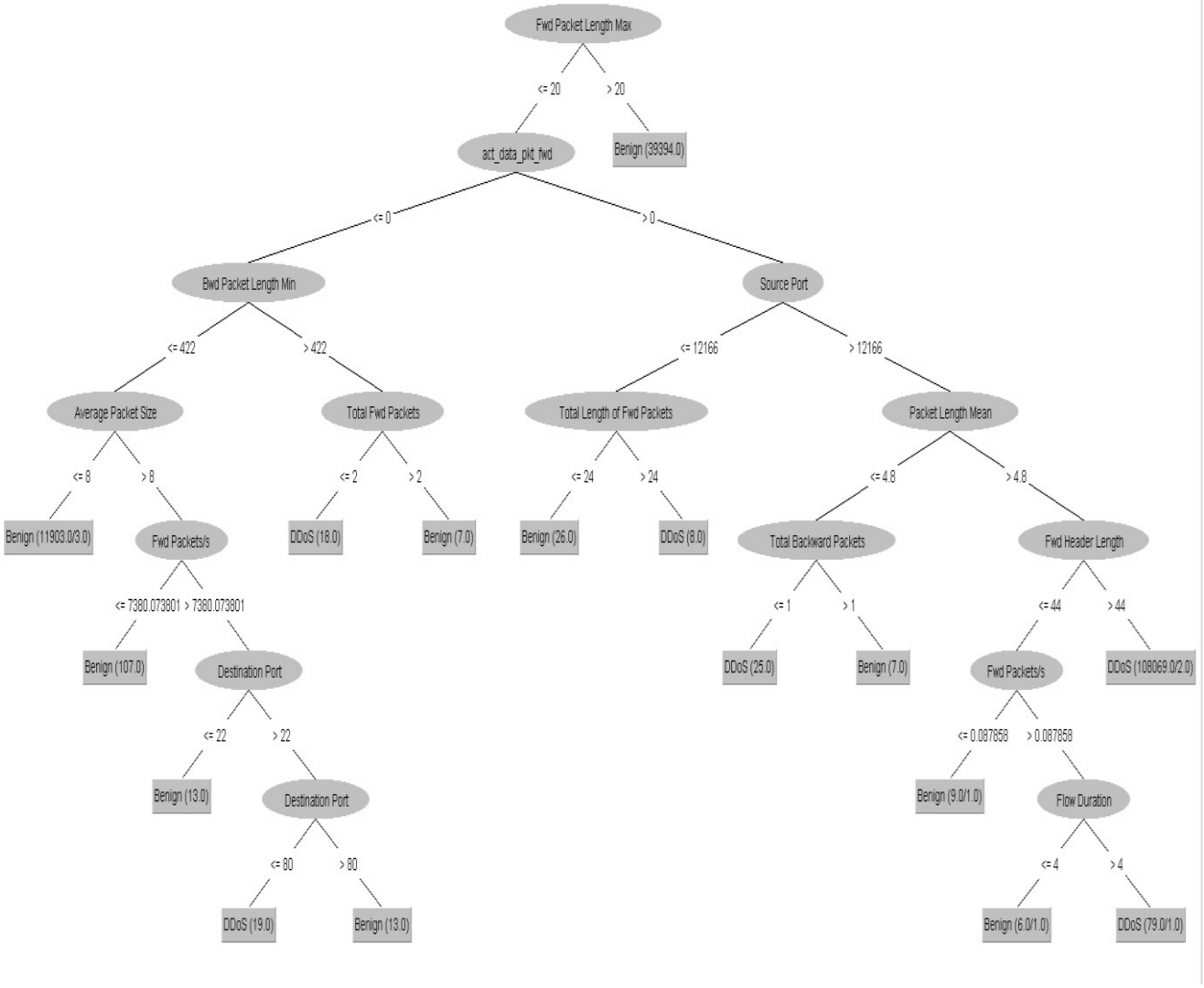
To classify using a decision tree on the training data, the J48 algorithm is used and the results are shown in Listings 6-5 for the pruned tree and 6-6 for the classification tree. The results in Listing 6-5 shows a tree size of 31 with 16 leaves. The model shows that there are 159695 correctly classified instances of the 159703 instances with 8 incorrectly classified instances showing a 99% classification accuracy with 0.05 classification error. This shows that using both classification algorithms to build the predictive analytics classification model yields similar accuracy results. The classification tree for the pruned tree are illustrated in Listing 6-6.

Classification using decision tree.

```
=== Classifier model (full training set) ===

J48 pruned tree
------------------

 Fwd Packet Length Max <= 20
|   act_data_pkt_fwd <= 0
|   |   Bwd Packet Length Min <= 422
|   |   |   Average Packet Size <= 8: Benign (11903.0/3.0)
|   |   |   Average Packet Size > 8
|   |   |   |   Fwd Packets/s <= 7380.073801: Benign (107.0)
|   |   |   |   Fwd Packets/s > 7380.073801
|   |   |   |   |   Destination Port <= 22: Benign (13.0)
|   |   |   |   |   Destination Port > 22
|   |   |   |   |   |   Destination Port <= 80: DDoS (19.0)
|   |   |   |   |   |   Destination Port > 80: Benign (13.0)
|   |   Bwd Packet Length Min > 422
|   |   |   Total Fwd Packets <= 2: DDoS (18.0)
|   |   |   Total Fwd Packets > 2: Benign (7.0)
|   act_data_pkt_fwd > 0
|   |   Source Port <= 12166
|   |   |   Total Length of Fwd Packets <= 24: Benign (26.0)
|   |   |   Total Length of Fwd Packets > 24: DDoS (8.0)
|   |   Source Port > 12166
|   |   |   Packet Length Mean <= 4.8
|   |   |   |   Total Backward Packets <= 1: DDoS (25.0)
|   |   |   |   Total Backward Packets > 1: Benign (7.0)
|   |   |   Packet Length Mean > 4.8
|   |   |   |   Fwd Header Length <= 44
|   |   |   |   |   Fwd Packets/s <= 0.087858: Benign (9.0/1.0)
|   |   |   |   |   Fwd Packets/s > 0.087858
|   |   |   |   |   |   Flow Duration <= 4: Benign (6.0/1.0)
|   |   |   |   |   |   Flow Duration > 4: DDoS (79.0/1.0)
|   |   |   |   Fwd Header Length > 44: DDoS (108069.0/2.0)
 Fwd Packet Length Max > 20: Benign (39394.0)

Number of Leaves  :     16

Size of the tree :      31


=== Summary ===

Correctly Classified Instances      159695               99.995  %
Incorrectly Classified Instances         8                0.005  %
Kappa statistic                          0.9999
Mean absolute error                      0.0001
Root mean squared error                  0.0069
Relative absolute error                  0.0221 %
Root relative squared error              1.4865 %
Total Number of Instances           159703

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     Benign
               1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     DDoS
Weighted Avg.  1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000

=== Confusion Matrix ===

     a      b   <-- classified as
 51480      3 |      a = Benign
     5 108215 |      b = DDoS
```

Listing 5-3 Pruned Tree using J48 algorithm

Classification Tree



Listing 5-4 Classification Tree

5.2.3 Clustering

To determine the patterns from the data, groups of the data are made. Clustering is an unsupervised machine-learning algorithm that can be used to improve the accuracy of supervised machine learning algorithms as well as clustering the data points into similar groups. Clustering in data mining, is a step further from classification as every attribute in the dataset is used to analyse the data, where as in classification, only a subset of the attributes is used in building the model. In this case, every attribute is normalised and each value is divided by the difference between the higher and lower values in the dataset of the same attribute. To

find the best model for predictive analytics two clustering algorithms (K-mean and Expectation Maximisation) are used.

K-means is an iterative clustering algorithm that aims to find local maxima in each iteration. The algorithm iterates until there is no more movement on the data-points or centroids. The K-means algorithm works in five steps that

- Specifies the desired number of clusters
- Randomly assigns each data point to a cluster
- Computes cluster centroids
- Re-assigns each point to the closest cluster centroid
- Re-computes cluster centroids

Clustering was applied to the training data set. The results shown in listing 6-7 show two clusters of either a 0 or a 1 which are benign and DDoS respectively. The total clustered instances based on the DDoS attacks are 19,654 with 62% of the clustering being DDoS and 38% are the post scans. There were eight iterations made within the sum of 28,069 squared errors. The missing values have been replaced by either the mode or the mean. The final cluster centroids are presented in listing 6-7 below.

```
Number of iterations: 5
Within cluster sum of squared errors: 370321.44523596833

Initial starting points (random):

Cluster 0: DDoS,192.168.56.103,42522,192.168.56.101,80,6,'29/10/2019 04:15',16966,3,5,26,11607,20,0,10135,0,72,112,176.824237,294.707061,0,10135,1292.555556,3350.634907,11200000,0,0,0,1,1454.125,2
Cluster 1: Benign,192.168.56.103,44926,192.168.56.101,53,17,'29/10/2019 04:05',179,2,2,66,164,33,33,82,82,80,80,11173.18436,11173.18436,33,82,52.6,26.838405,720.3,0,0,0,1,65.75,1

Missing values globally replaced with mean/mode

Final cluster centroids:
                                        Cluster#
Attribute                  Full Data            0                1
                           (159703.0)    (69717.0)        (89986.0)
=============================================================================
 Label                            DDoS         DDoS           Benign
 Source IP             192.168.56.103   192.168.56.103   192.168.56.103
 Source Port              39265.4616       46246.4587       33856.9071
 Destination IP        192.168.56.101   192.168.56.101   192.168.56.101
 Destination Port          8415.5494        1189.6815        14013.8183
 Protocol                     7.6217                6           8.8781
 Timestamp  29/10/2019 04:13 29/10/2019 03:57 29/10/2019 04:13
 Flow Duration       16034400.3201    22715672.7161    10858058.8043
 Total Fwd Packets           3.951           4.4535           3.5616
 Total Backward Packets      3.4716          5.0535           2.2461
Total Length of Fwd Packets  977.1592        268.5038        1526.1927
 Total Length of Bwd Packets 5011.7221       11369.866         85.7256
 Fwd Packet Length Max       603.3728        226.5824         895.2926
 Fwd Packet Length Min        30.1333          0.0002           53.479
Bwd Packet Length Max        3131.2047         7121.44          39.7546
 Bwd Packet Length Min       16.8198          0.1778           29.7132
 Fwd Header Length           87.8335         101.3344          77.3737
 Bwd Header Length           77.4365         112.8072          50.0329
Fwd Packets/s                1514.8578         21.078         2672.1695
 Bwd Packets/s               792.5451         30.2061         1383.1702
 Min Packet Length             8.0909          0.0002           14.3592
 Max Packet Length           3715.3397       7328.0228         916.3996
 Packet Length Mean           589.2252        1162.6202         144.9852
 Packet Length Std           1258.5515        2512.4605         287.0806
 Packet Length Variance     3325101.4685     6866367.2282      581492.1851
FIN Flag Count                  0.0002                0           0.0004
 SYN Flag Count                 0.0227                0           0.0402
 ACK Flag Count                  0.512           0.2805           0.6913
 Down/Up Ratio                  1.0828          0.9607           1.1775
 Average Packet Size           658.7719        1298.2344         163.3458



Time taken to build model (full training data) : 0.83 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       69717 ( 44%)
1       89986 ( 56%)
```

Listing 5-5 Clustering model using K means

The results in listing 6-7 show the K-means clustering of data with two groups of packets; benign and DDoS highlighted based on all selected attributes, where 0 represents the benign

packets and 1 represents the DDoS packets. The clustering results in listing 6-7 are illustrated in the graph in Figure 6-15 below.

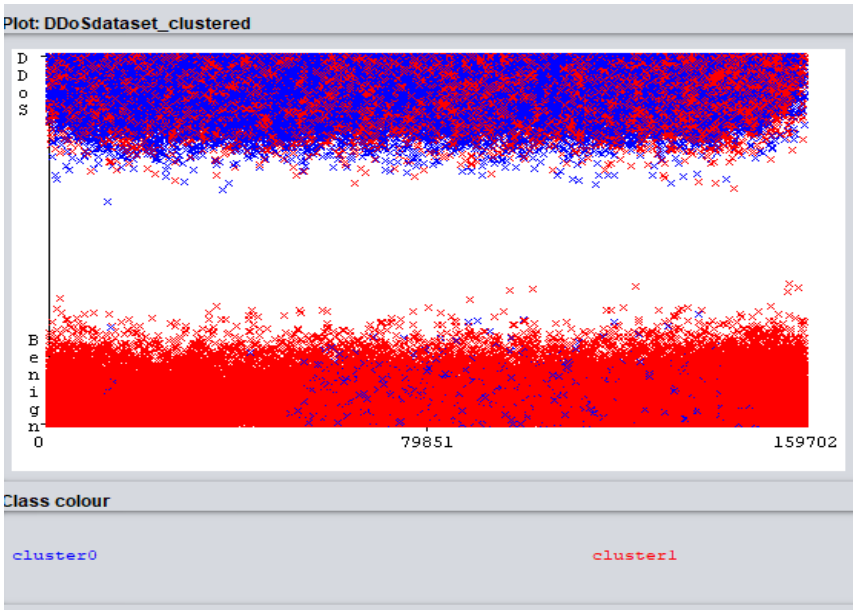


Figure 5-13 Visualisation of the of the K means Clustering

Expectation maximisation (EM) algorithm on the other hand is an iterative method of finding maximum likelihood maximum a-posteriori (MAP) estimates of parameters in statistical modelling. The model in EM depends on unobserved variables. The EM algorithm iterates alternately between performing an expectation (E) which creates a function for the expectation of the log likelihood that is evaluated using current estimates for the given parameters, and a maximisation (M) which computes parameters that maximise the log likelihood found in E. A sample of the results of the EM clustering are shown in listing 6-8 with a visual representation shown in Figure 6-16. The results presented are based on one iteration performance using cross validation methods with 5 folds. There are five clusters produced and each attribute in the dataset is represented in the 5 clusters. The clustered Instances are broken down as follows

```
Clustered Instances

0       69717 ( 44%)
1       89986 ( 56%)
```

```
EM
==

Number of clusters selected by cross validation: 8
Number of iterations performed: 2
```

Attribute	Cluster 0 (0.15)	1 (0.25)	2 (0.08)	3 (0)	4 (0.06)	5 (0.25)	6 (0.12)	7 (0.09)
Label								
Benign	24470.0123	1	12378.8833	622.3493	1	16.046	1	14000.7091
DDoS	63	39472.7878	1	146.951	9496.2122	39689.049	19355	4
[total]	24533.0123	39473.7878	12379.8833	769.3003	9497.2122	39705.095	19356	14004.7091
Source IP								
192.168.56.103	19388.5899	39472.7878	5	739.0125	9496.2122	39704.095	19355	59.3026
192.168.56.102	5144.4224	1	12374.8833	30.2878	1	1	1	13932.4065
74.125.29.157	1	1	1	1	1	1	1	3
104.24.21.85	1	1	1	1	1	1	1	8
172.217.6.234	1	1	1	1	1	1	1	3
172.217.9.238	1	1	1	1	1	1	1	3
[total]	24537.0123	39477.7878	12383.8833	773.3003	9501.2122	39709.095	19360	14008.7091
Source Port								
mean	50350.1828	43163.7074	83.1059	42074.5302	43115.8864	44656.9434	57445.0277	314.6149
std. dev.	16409.7887	15067.7133	172.7927	12253.3034	14888.7488	15102.4366	4864.2668	3448.5489
Destination IP								
192.168.56.101	24500.8956	39472.7878	5	767.3003	9496.2122	39702.095	19355	87.7091
192.168.56.102	2	1	1	1	1	3	1	2
192.168.56.103	31.1167	1	12374.8833	1	1	1	1	13916
146.20.129.103	1	1	1	2	1	1	1	1
[total]	24535.0123	39475.7878	12381.8833	771.3003	9499.2122	39707.095	19358	14006.7091
Destination Port								
mean	186.4582	80	56637.7423	102.6844	80	85.1053	80	44950.9936
std. dev.	2414.4647	0	6436.8231	664.9293	0	509.709	19447.6809	14743.0935
Protocol								
mean	16.5575	6	6	6	6	6	6	6
std. dev.	2.1661	3.9002	3.9002	3.9002	3.9002	3.9002	3.9002	3.9002
Timestamp								
29/10/2019 03:32	33	1	1	1	1	1	1	2
29/10/2019 03:33	25	1	1	1	1	1	1	1
29/10/2019 03:34	25	1	1	1	1	1	1	1

Listing 5-6 Clustering results using EM algorithm

Clustered instances are

```
Time taken to build model (full training data) : 2927.91 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       24506 ( 15%)
1       42536 ( 27%)
2       12373 (  8%)
3         828 (  1%)
4        6431 (  4%)
5       39684 ( 25%)
6       19354 ( 12%)
7       13991 (  9%)

Log likelihood: -104.5344
```

The results in listing 6-8 are visualised in the scatterplot shown in Figure 6-16 below.

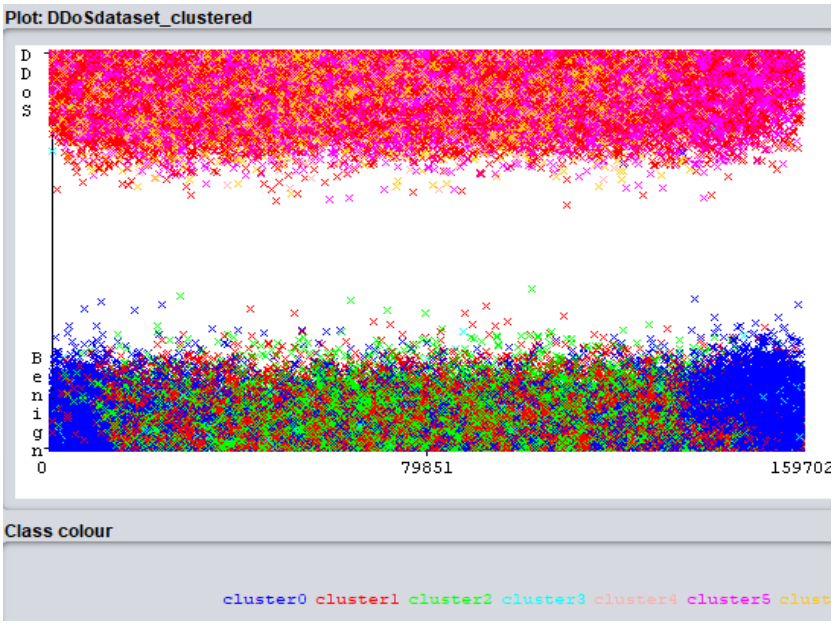


Figure 5-14 Clustering using the EM algorithm

Clustering of instances marks the final stage of the EDA process. Once the results of the EDA process have been achieved, the next stage is to build the predictive analytics model. Predictive analytics modelling will use the results obtained in the EDA process. Section 6.3 describes the process of building, validating and testing the predictive model using various algorithms. As most available analytical problems proposed by many security professionals as explained in chapter 3 end with EDA, this project moves a step further from exploration or exploratory analytics to predictive analytics as described in the section that follows.

5.3 Building Predictive Model

To build a machine-learning model, all the steps of the EDA are taken into consideration from features selected in section 6.2.2 of the data preparation process all the way to the classification and clustering of the attributes. The results from the training data used in the initial classification during data preparation are used to build the security management predictive model and to train the model, and the testing data will be used to test the built predictive model.

Since the dataset used has 31 attributes with 159,703 instances as shown in Listing 6-2. The dataset is split into two using a 90% to 10% split ratio with 90% used as training data and 10% is the testing data for each model created. The security management model is built using a linear regression algorithm using supervised learning techniques as described in section 6.1which provides a brief description of what linear regression is and how it is applied to predictive analytics modelling and the various equations used in predictive modelling.

5.3.1 Predictive Analytics Modelling using Linear Regression

Linear regression is used to predict the value of an outcome variable Y based on one or more input predictor variables X. A linear regression model assumes a linear relationship between the input variables and the outcome variables. There is a difference between a regression problem and classification problem in that, regression problems try to predict numeric values whereas classification problems predict nominal values as demonstrated in the EDA process in section 6.2. For predictive linear modelling, the class attribute represented as label in the dataset is used. There is a problem however, with the class attribute, as nominal values cannot be used for linear regression modelling. An example of how the labels are currently presented in the dataset as nominal values is shown Figure 6-17 below.

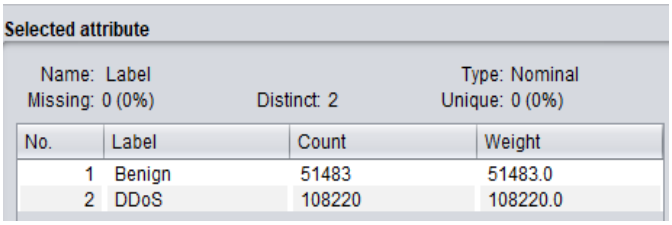


Figure 5-15 Sample of attributes used for Linear Regression

The nominal values therefore need to be converted into binary or numeric attribute values; a form linear regression algorithm can easily read or work with. To convert nominal values to binary values, unsupervised learning is used, the results of which, are illustrated in Figure 6-18 below. The results now show the class label as 0 for benign and 1 for DDoS with no missing

values. Figure 6-18 shows the minimum value of representation as a 0 and the maximum represented value of 1 with the mean of 0.6 and the standard deviation of 0.4 whereas Figure 6-18 shows the total counts of the instances together with their weight values.

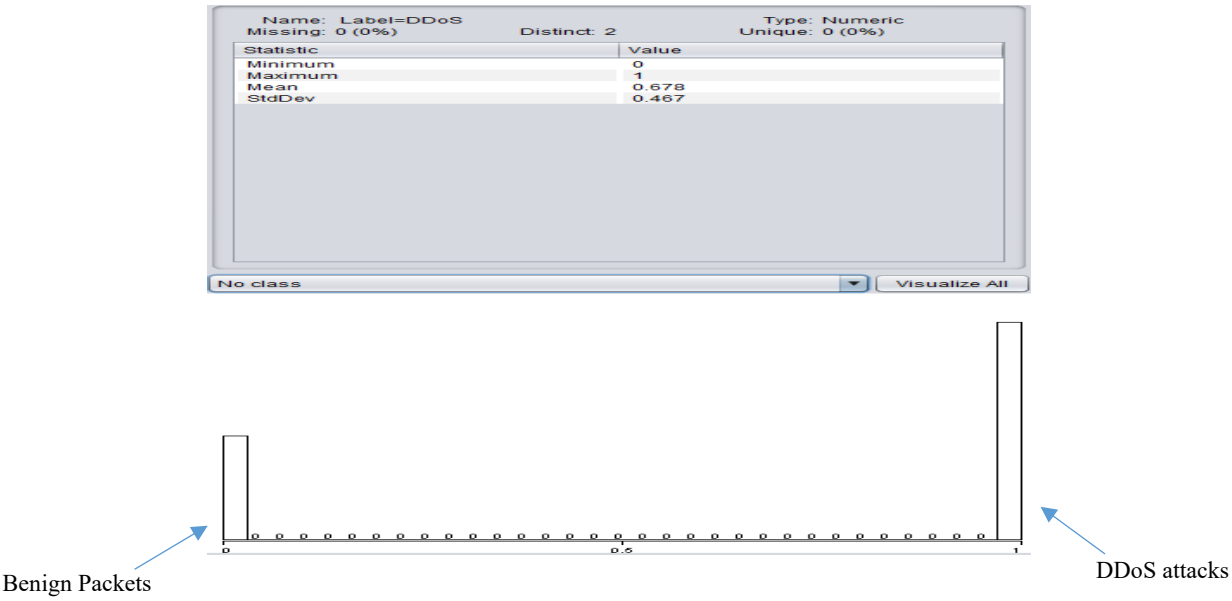


Figure 5-16 Results of Nominal to Binary values

Linear regression modelling can either be done using the training data to train the model or cross validation methods to build the model. Once the nominal values have been converted to numeric, the training dataset is used in this instance for predictive linear regression modelling. In the second instance, cross validation methods with 10 folds are used for comparison purposes. The unsupervised learning technique was used in both instances to build the models.

Since the dataset used has both nominal and numeric attributes with two-class values, it is only logical to draw a basic straight-line using the equation

$$\boldsymbol{w_0 a_0^{(1)} + w_1 a_1^{(1)} + w_2 a_2^{(1)} + \ldots w_k a_k^{(1)}}$$

, which calculates the weight of each dimension from the training data. Once the weights of the attributes are calculated a prediction of the values from the first training instance $\boldsymbol{a_1}$ is made using the first value of the test instance that has just been calculated. The $\boldsymbol{w}$ represents

the training values that are being calculated $\boldsymbol{a}$ represents the attribute values of the test data and the number (1) in brackets represents the first attribute of the test instance.

The whole equation can be represented by the sigma equation after the equal sign as illustrated below.

Calculates the first attribute values of the test instances

Calculates the predictive values of the training instances

$$w_0 a_0^{(1)} + w_1 a_1^{(1)} + w_2 a_2^{(1)} + \ldots w_k a_k^{(1)} = \sum_{j=0}^{k} w_j a_j^{(1)}$$

By this, $\sum_{j=0}^{k} w_j a_j^{(1)}$ we are trying to define the zeroth attribute value to be a 1, which gives out a predicted value $\boldsymbol{x}$ of the first training instance and then choose the weights to minimise the squared error on the training data which is achieved by the equation below.

The actual value of the training instance

$$\sum_{i=1}^{n} \left(x^i - \sum_{j=0}^{k} w_j a_j^{(1)} \right)^2$$

The Predictive value for the training instance

We take the difference between the actual value and the predictive value, then square them up then add them together, and that is what we are trying to minimise. The weights are obtained by minimizing the sum of square errors and given the large sum of instances, this is the best option to minimise errors. Standard matrix problems work well if there are more instances than attributes as in this case. To perform linear regression the considered nominal attributes are converted to numeric class labels of 0s and 1s.

A sample of the linear regression modelling results using the training dataset is shown in listing 6-9 below.

Linear Regression Model

```
Linear Regression Model

 Label=DDoS =

     0.7606 *  Source IP=104.24.21.85,172.217.9.238,172.217.6.234,192.168.56.102,192.168.56.103 +
     0.0115 *  Source IP=192.168.56.103 +
    -0      *  Source Port +
    -0.5381 *  Destination IP=146.20.129.103,192.168.56.103,192.168.56.101 +
     0.8661 *  Destination IP=192.168.56.101 +
    -0      *  Destination Port +
    -0.0682 *  Protocol +
    -0.0082 *  Timestamp=29/10/2019 04:13,29/10/2019 03:56,29/10/2019 04:11,29/10/2019 04:00,29/10/2019 04:07,29/10/2019 04:15,29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,
    -0.0034 *  Timestamp=29/10/2019 03:56,29/10/2019 04:11,29/10/2019 04:00,29/10/2019 04:07,29/10/2019 04:15,29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,
     0.0052 *  Timestamp=29/10/2019 04:11,29/10/2019 04:00,29/10/2019 04:07,29/10/2019 04:15,29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,
    -0.0072 *  Timestamp=29/10/2019 04:00,29/10/2019 04:07,29/10/2019 04:15,29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,
     0.004  *  Timestamp=29/10/2019 04:07,29/10/2019 04:15,29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,
     0.0062 *  Timestamp=29/10/2019 04:15,29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,
    -0.0047 *  Timestamp=29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,
     0.0093 *  Timestamp=29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
    -0.0147 *  Timestamp=29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
     0.015  *  Timestamp=29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
     0.0034 *  Timestamp=29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
    -0.0043 *  Timestamp=29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
     0.0044 *  Timestamp=29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
    -0.0008 *  Timestamp=29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
    -0.0052 *  Timestamp=29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
    -0.0072 *  Timestamp=29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
     0.0132 *  Timestamp=29/10/2019 03:57,29/10/2019 04:01 +
     0.008  *  Timestamp=29/10/2019 04:01 +
    -0      *  Flow Duration +
    -0.0763 *  Total Fwd Packets +
    -0.0241 *  Total Backward Packets +
    -0      *  Total Length of Fwd Packets +
     0      *  Total Length of Bwd Packets +
     0      *  Fwd Packet Length Max +
    -0      *  Fwd Packet Length Min +
    -0      *  Bwd Packet Length Max +
    -0.0002 *  Bwd Packet Length Min +
     0.0001 *  Fwd Header Length +
     0.0003 *  Bwd Header Length +
    -0      *  Fwd Packets/s +
    -0      *  Bwd Packets/s +
     0.0011 *  Min Packet Length +
    -0      *  Packet Length Std +
    -0      *  Packet Length Variance +
    -0.0264 *  FIN Flag Count +
    -0.0238 *  SYN Flag Count +
     0.1693 *  ACK Flag Count +
     0.0112 *  Down/Up Ratio +
     0.0002 *  Average Packet Size +
     0.0896 *  act_data_pkt_fwd +
     0.0817

Time taken to build model: 42.38 seconds
```

Listing 5-7 Linear Regression Model

The formula calculates the weights represented by the values on the left side of the results, which are multiplied, by the attribute values, which keep adding and multiplying cumulatively from the first attribute value or instance all the way to the last. Due to the number of results

displayed and the wide screen margin, the results in both Listing 6-9 and 6-10 are screen shots used for illustration purposes only. The error margins are calculated by the equations listed besides each error as shown in listing 6-10. The results illustrate a DDoS linear regression model with a summary of 0.98 correlation coefficient, a mean absolute error of 0.03, a root squared error of 0.08 and a relative squared error of 17.2. The results in listing 6-9 show the linear regression formula for computing the class model as depicted in listing 6-10.

Regression Model Computational Equatio

```
Linear Regression Model

 Label=DDoS =

     0.7606 *  Source IP=104.24.21.85,172.217.9.238,172.217.6.234,192.168.56.102,192.168.56.103 +
     0.0115 *  Source IP=192.168.56.103 +
     -0     *  Source Port +
    -0.5381 *  Destination IP=146.20.129.103,192.168.56.103,192.168.56.101 +
     0.8661 *  Destination IP=192.168.56.101 +
     -0     *  Destination Port +
    -0.0682 *  Protocol +
    -0.0082 *  Timestamp=29/10/2019 04:13,29/10/2019 03:56,29/10/2019 04:11,29/10/2019 04:00,29/10/2019 04:07,29/10/2019 04:15,29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,
    -0.0034 *  Timestamp=29/10/2019 03:56,29/10/2019 04:11,29/10/2019 04:00,29/10/2019 04:07,29/10/2019 04:15,29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,
     0.0052 *  Timestamp=29/10/2019 04:11,29/10/2019 04:00,29/10/2019 04:07,29/10/2019 04:15,29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,
    -0.0072 *  Timestamp=29/10/2019 04:00,29/10/2019 04:07,29/10/2019 04:15,29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,
     0.004  *  Timestamp=29/10/2019 04:07,29/10/2019 04:15,29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,
     0.0062 *  Timestamp=29/10/2019 04:15,29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,
    -0.0047 *  Timestamp=29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,
     0.0093 *  Timestamp=29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
    -0.0147 *  Timestamp=29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
     0.015  *  Timestamp=29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
     0.0034 *  Timestamp=29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
    -0.0043 *  Timestamp=29/10/2019 04:08,29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
     0.0044 *  Timestamp=29/10/2019 04:12,29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
    -0.0008 *  Timestamp=29/10/2019 04:04,29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
    -0.0052 *  Timestamp=29/10/2019 03:59,29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
    -0.0072 *  Timestamp=29/10/2019 04:03,29/10/2019 03:57,29/10/2019 04:01 +
     0.0132 *  Timestamp=29/10/2019 03:57,29/10/2019 04:01 +
     0.008  *  Timestamp=29/10/2019 04:01 +
     -0     *  Flow Duration +
    -0.0763 *  Total Fwd Packets +
    -0.0241 *  Total Backward Packets +
     -0     *  Total Length of Fwd Packets +
      0     *  Total Length of Bwd Packets +
      0     *  Fwd Packet Length Max +
     -0     *  Fwd Packet Length Min +
     -0     *  Bwd Packet Length Max +
    -0.0002 *  Bwd Packet Length Min +
     0.0001 *  Fwd Header Length +
     0.0003 *  Bwd Header Length +
     -0     *  Fwd Packets/s +
     -0     *  Bwd Packets/s +
     0.0011 *  Min Packet Length +
     -0     *  Packet Length Std +
     -0     *  Packet Length Variance +
    -0.0264 *  FIN Flag Count +
    -0.0238 *  SYN Flag Count +
     0.1693 *  ACK Flag Count +
     0.0112 *  Down/Up Ratio +
     0.0002 *  Average Packet Size +
     0.0896 *  act_data_pkt_fwd +
     0.0817

Time taken to build model: 42.38 seconds
```

Regression Classification Equation

```
=== Evaluation on training set ===

Time taken to test model on training data: 0.73 seconds

=== Summary ===

Correlation coefficient                  0.9849
Mean absolute error                      0.0303
Root mean squared error                  0.0808
Relative absolute error                  6.9367 %
Root relative squared error             17.2859 %
Total Number of Instances           159703
```

Error Margins

Listing 5-8 Formula for computing the Class and the error Margins

To check how the attributes behave in the model a tree model is used. The results of the predictive linear regression model are validated by a model tree shown in listing 6-11 below.

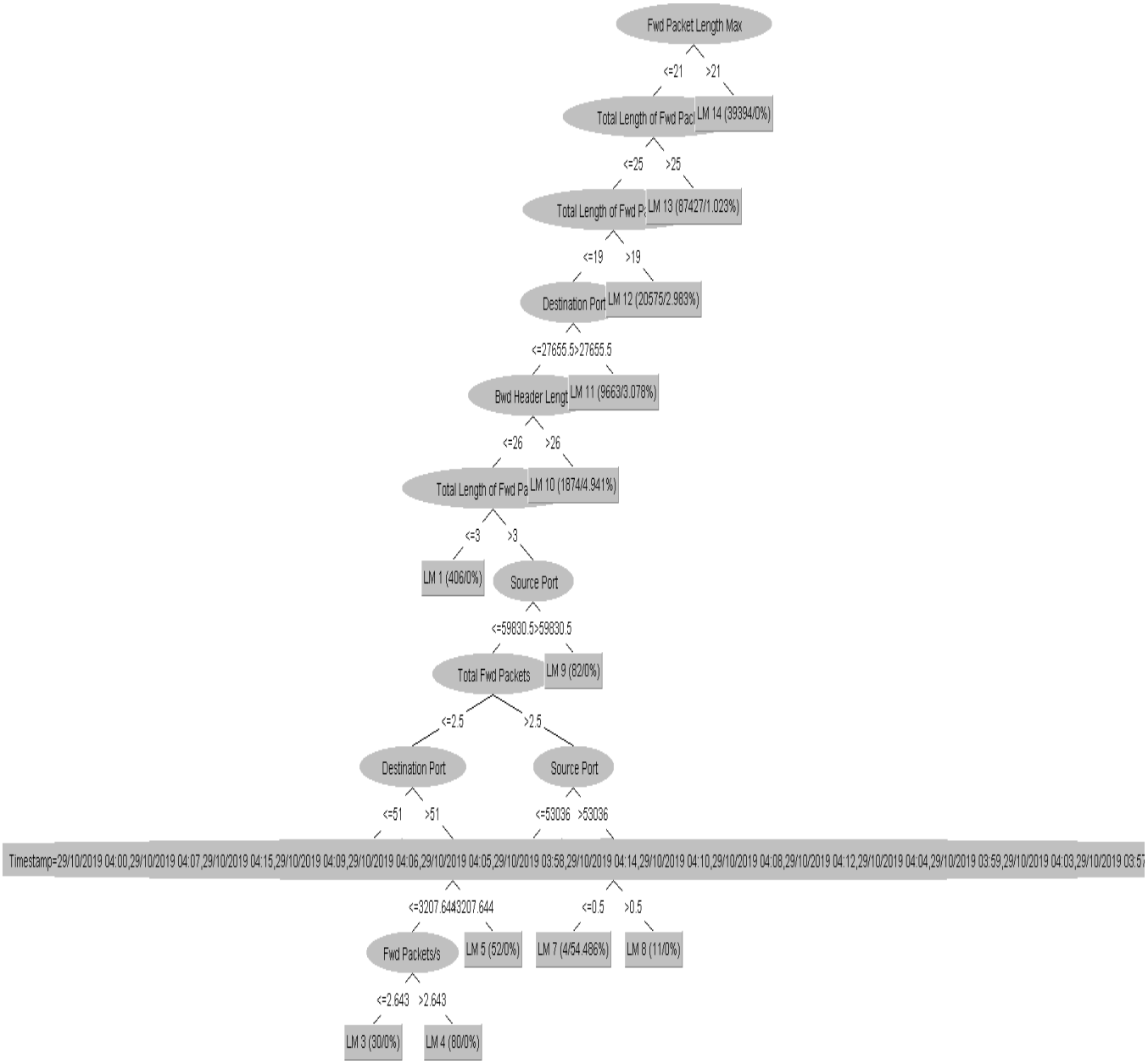
Pruned Linear Regression Model Tree

```
=== Classifier model (full training set) ===

M5 pruned model tree:
(using smoothed linear models)

 Fwd Packet Length Max <= 21 :
|  Total Length of Fwd Packets <= 25 :
|  |  Total Length of Fwd Packets <= 19 :
|  |  |  Destination Port <= 27655.5 :
|  |  |  |  Bwd Header Length <= 26 :
|  |  |  |  |  Total Length of Fwd Packets <= 3 : LM1 (406/0%)
|  |  |  |  |  Total Length of Fwd Packets >  3 :
|  |  |  |  |  |  Source Port <= 59830.5 :
|  |  |  |  |  |  |  Total Fwd Packets <= 2.5 :
|  |  |  |  |  |  |  |  Destination Port <= 51 : LM2 (49/0%)
|  |  |  |  |  |  |  |  Destination Port >  51 :
|  |  |  |  |  |  |  |  |  Bwd Packets/s <= 3207.644 :
|  |  |  |  |  |  |  |  |  |  Fwd Packets/s <= 2.643 : LM3 (30/0%)
|  |  |  |  |  |  |  |  |  |  Fwd Packets/s >  2.643 : LM4 (80/0%)
|  |  |  |  |  |  |  |  |  Bwd Packets/s >  3207.644 : LM5 (52/0%)
|  |  |  |  |  |  |  Total Fwd Packets >  2.5 :
|  |  |  |  |  |  |  |  Source Port <= 53036 : LM6 (56/0%)
|  |  |  |  |  |  |  |  Source Port >  53036 :
|  |  |  |  |  |  |  |  |  Timestamp=29/10/2019 04:00,29/10/2019 04:07,29/10/2019 04:15,29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019
|  |  |  |  |  |  |  |  |  Timestamp=29/10/2019 04:00,29/10/2019 04:07,29/10/2019 04:15,29/10/2019 04:09,29/10/2019 04:06,29/10/2019 04:05,29/10/2019 03:58,29/10/2019 04:14,29/10/2019 04:10,29/10/2019 04:08,29/10/2019
|  |  |  |  |  |  |  Source Port >  59830.5 : LM9 (82/0%)
|  |  |  |  Bwd Header Length >  26 : LM10 (1874/4.941%)
|  |  |  Destination Port >  27655.5 : LM11 (9663/3.078%)
|  |  Total Length of Fwd Packets >  19 : LM12 (20575/2.983%)
|  Total Length of Fwd Packets >  25 : LM13 (87427/1.023%)
 Fwd Packet Length Max >  21 : LM14 (39394/0%)


=== Evaluation on training set ===

Time taken to test model on training data: 0.25 seconds

=== Summary ===

Correlation coefficient                  0.9998
Mean absolute error                      0.0004
Root mean squared error                  0.0096
Relative absolute error                  0.0863 %
Root relative squared error              2.0527 %
Total Number of Instances           159703
```

Listing 5-9 Regression Model Tree

The model tree produced represents each leaf of the model as a linear regressions model. These are like linear patches, which approximate each linear function. The results of the regression tree model is illustrated by the regression model tree listed in Listing 6-12 below. The results of the tree show that it has 9 leaves each with a linear model for DoS class.

Regression Model Tree



Listing 5-10 Regression Model Tree Visualisation

The results in listing 6-13 below show the predicted values for linear predictive model built in listing 6-9 above. The results show that predictions are made for each instance of the

model with the actual values, which are either 0s or 1s correctly predicted. The error margins of each prediction are also presented in listing 6-13 below.

Predicted Values of the Regression Tree Model

inst#	actual	predicted	error
1	1	1	0
2	1	1	-0
3	1	1	-0
4	1	1	-0
5	1	1	-0
6	1	1	-0
7	0	-0	-0
8	1	1	-0
9	0	-0	-0
10	1	1	-0
11	1	1	-0
12	1	1	-0
13	0	0	0
14	0	-0	-0
15	1	1	-0
16	0	0.148	0.148
17	1	1	-0
18	0	0	0
19	1	1	-0
20	0	0	0
21	1	1	-0
22	1	1	-0
23	0	-0	-0
24	1	1	-0
25	0	0	0
26	1	1	-0
27	1	1	-0
28	1	1	-0
29	1	1	-0
30	1	1	-0
31	1	1	-0
32	0	0	0
33	1	1	-0
34	1	1	-0
35	1	1	-0
36	1	1	-0
37	1	1	-0
38	1	1	-0
39	1	1	-0
40	0	0	0
41	1	1	-0
42	1	1	-0
43	1	1	-0

Listing 5-11 Predictive model Values

These predictions are illustrated in the boxplot shown in the graph in Figure 6-19 below.

Figure 5-17 Predictive Model Visualisation

To create a predictive model two class values are considered since we are dealing with a two-class problem where we are trying to determine whether a packet is an attack or just a genuine scan. We call the two class values 0 and 1, where 0 represents a benign packet and 1 represents DDoS attack. Prediction is set to a threshold for 0 or 1 and a prediction is done for each class value where the output is set to 1 for training instances that belong to the class 1 and 0 for those that are not. This is illustrated in the results in Figure 6-20

Name: Label=DoS Type: Nominal
Missing: 0 (0%) Distinct: 2 Unique: 0 (0%)

No.	Label	Count	Weight
1	0	4450	4450.0
2	1	15204	15204.0

Figure 5-18 Total number of Predicted Values for both Class Attributes

The linear regression predictive model that has been built needs to be checked for accuracy through the validation process. During model validation, the trained model is evaluated using the test dataset that was set at 10% during the dataset split. Model validation is presented in the section that follows.

5.4 Validating the Predictive Model

The trained model is validated by calculating the accuracy of the model prediction on new data. To test the accuracy of the predictive model that has been created, the model was run against the testing dataset using supervised learning techniques. Since the prediction is being made on whether or not the packet forwarded is an attack, only the label class is used for classification testing, the rest of the attributes were deleted and a new attribute consisting of the regression classification computational algorithm illustrated in listing 6-10 for the predictive model was added. A linear regression algorithm was again applied to the two attributes without the dependant attributes of the dataset to see if the model can predict the class values correctly.

This will help determine if the predictive model created can accurately predict the DoS attacks and the port scans by just using the class labels.

To avoid overfitting of the model onto the new data because of the vast number of instances of the value attributes, a sampling range of 1,000 was used. The results obtained show the accuracy level of prediction of 19,654 instances as originally depicted in section 6.2 where 17,711 instances are correctly predicted giving a 90% accuracy and 1,943 incorrectly predicted instances at 10%. These results show improved predictive value which means that the predictive model built is a good model and can be deployed as a predictive analytics model for the management of security in a virtualised environment. Deployment of the model is discussed in section 6.5.

The results of the validation are illustrated in listing 6-1below

```
Attributes:   2
              Label=DoS
              classification
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

classification:
        < 1.199809239623554     -> 0
        < 7.193560271178628     -> 1
        >= 7.193560271178628    -> 0
(17629/19654 instances correct)


Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       17711               90.114  %
Incorrectly Classified Instances      1943                9.886  %
Kappa statistic                          0.6773
Mean absolute error                      0.0989
Root mean squared error                  0.3144
Relative absolute error                 28.2199 %
Root relative squared error             75.1282 %
Total Number of Instances            19654

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.604    0.012    0.937      0.604    0.735      0.702    0.796     0.656     0
                 0.988    0.396    0.895      0.988    0.939      0.702    0.796     0.894     1
Weighted Avg.    0.901    0.309    0.904      0.901    0.893      0.702    0.796     0.840

=== Confusion Matrix ===

     a     b   <-- classified as
  2689  1761 |     a = 0
   182 15022 |     b = 1
```

Listing 5-12 Fitting the Predictive Model to New Data

The accuracy of the predictive model is further tested using a probability model using Naïve Bayes and logistic algorithms, explained in the sections that follow.

5.4.1 Naïve Bayes

To test the accuracy of the predictive regression model we use the Naïve Bayes algorithm to check the probability of a packet being an attack or not. Naïve Bayes is a probabilistic classifier that calculates a set of probabilities by counting the frequency and combination of the given dataset (Patil and Sherekar, 2013). Naïve Bayes algorithm uses Bayes theorem, which assumes that all attributes are independent of the class variable. This assumption is rarely true hence, the classification as Naïve and yet the algorithm tends to perform well and learns rapidly in a supervised learning classification.

To test the accuracy of the predictive model Naïve Bayes algorithm is applied to the trained model and the class only rather than to the entire dataset. This is illustrated in the results Listings 6-15 and 6-16 respectively. The results show the probability of the first instance with the actual value of 2:1 correctly predicted as a 2:1 probability of a DoS attack happening. There are some errors for example instance 2 shows a 1.0 actual value but with a probability of 2:1. Instance 16 on the other hand, has also predicted the probability of the packet being a port scan as expected. There are a few errors in the results represented by the + in the error column of the second instance.

```
nstances:     19654
Attributes:   2
              Label=DoS
              classification
Test mode:    split 90.0% train, remainder test

=== Classifier model (full training set) ===

Naive Bayes Classifier

                      Class
Attribute               0        1
                     (0.23)   (0.77)
================================
classification
  mean                1.8322 3.0323
  std. dev.           3.6997 1.3966
  weight sum             4450  15204
  precision           0.0046 0.0046


Time taken to build model: 0.02 seconds

=== Predictions on test split ===

     inst#     actual  predicted error prediction
         1        2:1        2:1          0.643
         2        1:0        2:1     +    0.535
         3        2:1        2:1          0.904
         4        2:1        2:1          0.866
         5        2:1        2:1          0.904
         6        2:1        2:1          0.903
         7        2:1        2:1          0.878
         8        1:0        2:1     +    0.72
         9        2:1        2:1          0.867
        10        2:1        2:1          0.882
        11        2:1        2:1          0.886
        12        2:1        2:1          0.908
        13        2:1        2:1          0.903
        14        2:1        2:1          0.887
        15        2:1        2:1          0.904
        16        1:0        1:0          0.731
        17        2:1        2:1          0.879
        18        2:1        2:1          0.908
        19        1:0        2:1     +    0.908
        20        2:1        2:1          0.882
        21        2:1        2:1          0.883
        22        2:1        2:1          0.885
        23        2:1        2:1          0.868
        24        2:1        2:1          0.88
        25        2:1        2:1          0.908
```

Listing 5-13 Testing the predictive value using Naïve Bayes Result1

To evaluate the accuracy of the classification algorithm the performance of the classification is assesed. This is usualy determined by calculating the percentage of the tuples placed in the correct class as shwon in Listing 6-16 below, which shows the probabilty of the model correctly predicting packets at 78% with 1525 instances correctly classified and only 440 incorrectly classified based on the test split data used in the intial validation of predcitve model in section 6.4. The confusion matrix shown at the end of the resluts in listing 6-16 illustrates the accuracy of the solution to a classfication problem. Te confusion matrix provides information about the actual and predicted classifications normally evaluated by the the data matrix. The entries in the confusion matrix translates as a being the number of correct predictions of instance being negative and b is the number of incorrect predictions of instances being positive.

```
=== Evaluation on test split ===

Time taken to test model on test split: 1.11 seconds

=== Summary ===

Correctly Classified Instances        1525               77.6081 %
Incorrectly Classified Instances       440               22.3919 %
Kappa statistic                          0.1428
Mean absolute error                      0.2742
Root mean squared error                  0.3821
Relative absolute error                 78.8775 %
Root relative squared error             92.0964 %
Total Number of Instances             1965

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0.157    0.048    0.479      0.157    0.236      0.174    0.770     0.497     0
               0.952    0.843    0.799      0.952    0.869      0.174    0.770     0.908     1
Weighted Avg.  0.776    0.668    0.728      0.776    0.729      0.174    0.770     0.818

=== Confusion Matrix ===

    a    b   <-- classified as
   68  366 |    a = 0
   74 1457 |    b = 1
```

Listing 5-14 Testing the predictive value using Naïve Bayes Result1

To perform another type of validation on the built model, logistic regression was used. Like most if not all regression analysis algorithms, logistic regression is also a predictive analytics algorithm, which is used in this project to test and validate the accuracy of the built predictive regression model. Logistic regression performs well with linearly separable classes and that is the main reason we are using it to validate the linear regression predictive model that has been built. Logistic regression will help iron out any errors of the model as it uses maximum likelihood estimation of an outcome. The application of logistic regression to validate the linear regression model. This is explained in a little more detail in the section that follows.

5.4.2 Logistic Regression

To build a logistic regression model, we consider a logistic model with the given parameters and then see how the coefficients can be estimated from the data. We take the linear regression predictive model built earlier, which has two predictors, a DDoS expressed as x1 and a benign expressed as x2 and one response variable Y which is our expected outcome, and denote it as p=P(Y=1). Logistic regression modelling is used here to predict the probability of an attack

happening using the built predictive model. This is to establish the probability of the packet being a DDoS attack or is it just a normal random packet from the total number of forward packets sent, which is depicted by the formula below.

$$P(Packet = DoS|totalFwdPackets\ )$$

In other ways, we are modelling the probability that X belongs to the class (Y=1) which is expressed as

$$P(X) = P(P\ = 1|X)$$

We assume there is a linear relationship between the predictor variables and the log-odds of the event that Y=1. The linear relationship is expressed as

$$l\ = log_b\ \frac{p}{1-p} =\ \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

The odds are recovered by raising the log-odds to the power of the odds using the following equation

$$\frac{p}{1-p} = b^{\beta_0+\beta_1 x_1+\beta_2 x_2}$$

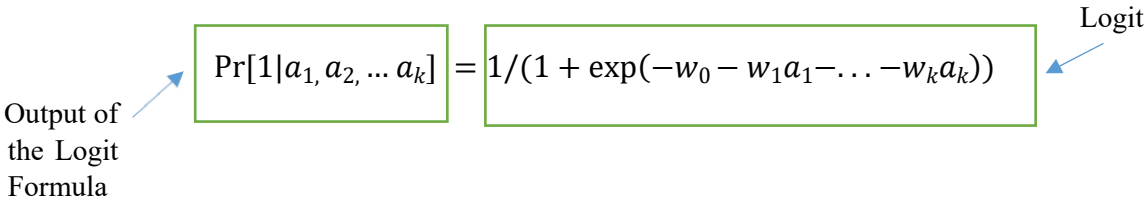
The probability that Y=1 is represented by the equation

$$p = \frac{b^{\beta_0+\beta_1 x_1+\beta_2 x_2}}{b^{\beta_0+\beta_1 x_1+\beta_2 x_2+1}} = \frac{1}{1+b^{\beta_0+\beta_1 x_1+\beta_2 x_2}}$$

The probability prediction will be transformed into binary values 0 and 1. Although logistics regression is a linear method, the predictions are made using the logistic function, which means that the predictions can no longer be considered as a linear combination of the inputs as in linear regression but can be expressed by the following equation

$$p(X) = e\hat{}(b0 + b1 * X)/(1 + e\hat{}(b0 + b1 * X))$$

To build a regression model a simple logit expression using the equation below was followed to transform the logit.



To build a linear model that estimates the class probability we use cross validation with a minimized log-likelihood as opposed to the squared error used in linear regression modelling. We also used ridge estimator to run the model. To achieve logistic regression an equation listed herein is used to calculate the probability.

$$\sum_{j=1}^{n}(1 - x^{(i)})\log(1 - \Pr\{1|a^1_1, a^2_2, \ldots, a^k_k\} + x^{(i)}\log(Pr1\{1|a^1_1, a^2_2, \ldots, a^k_k\})$$

The results obtained are listed in listing 6-17 below.

```
Logistic Regression with ridge parameter of 1.0E-8
Coefficients...
                   Class
Variable               0
=========================
classification    -0.5861
Intercept          0.1497


Odds Ratios...
                   Class
Variable               0
=========================
classification     0.5565

Time taken to build model: 0.09 seconds

=== Predictions on test data ===

    inst#     actual  predicted error prediction
        1        2:1        2:1       0.965
        2        2:1        2:1       0.711
        3        2:1        2:1       0.848
        4        2:1        2:1       0.738
        5        2:1        2:1       0.973
        6        2:1        2:1       0.758
        7        2:1        2:1       0.851
        8        2:1        2:1       0.879
        9        2:1        2:1       0.747
       10        2:1        2:1       0.957
       11        2:1        2:1       0.746
       12        2:1        2:1       0.828
       13        2:1        2:1       0.881
       14        2:1        2:1       0.721
       15        2:1        2:1       0.714
       ..        ...        ...       .....


     1945        1:0        1:0       0.511
     1946        1:0        2:1   +   0.928
     1947        1:0        2:1   +   0.956
     1948        1:0        2:1   +   0.82
     1949        1:0        2:1   +   0.829
     1950        1:0        2:1   +   0.918
     1951        1:0        2:1   +   0.708
     1952        1:0        1:0       0.568
     1953        1:0        1:0       0.531
     1954        1:0        2:1   +   0.595
     1955        1:0        2:1   +   0.946
     1956        1:0        2:1   +   0.835
     1957        1:0        2:1   +   0.919
     1958        1:0        2:1   +   0.976
     1959        1:0        2:1   +   0.627
     1960        1:0        2:1   +   0.946
     1961        1:0        2:1   +   0.85
     1962        1:0        2:1   +   1
     1963        1:0        2:1   +   0.851
     1964        1:0        1:0       0.591
     1965        1:0        2:1   +   0.6

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       16432               83.6064 %
Incorrectly Classified Instances      3222               16.3936 %
Kappa statistic                          0.3711
Mean absolute error                      0.2943
Root mean squared error                  0.3698
Relative absolute error                 84.0186 %
Root relative squared error             88.3683 %
Total Number of Instances            19654

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.276    0.000    0.999      0.276    0.433      0.477    0.764     0.738     0
                 1.000    0.724    0.825      1.000    0.904      0.477    0.764     0.867     1
Weighted Avg.    0.836    0.560    0.865      0.836    0.797      0.477    0.764     0.838

=== Confusion Matrix ===

     a     b   <-- classified as
  1229  3221 |     a = 0
     1 15203 |     b = 1
```

Listing 5-15 Logistic Modelling using Ridge

The results of the logistic model produced in listing 6-17 above show an 83% predictive accuracy of the built predictive model with 16% incorrectly classified instances.

The results show the probability of the class labels being either a 1 or a 0 as correctly predicted by the model.

With so many tools and techniques available for machine learning and deep learning predictive analytics is achieved by classification, statistical and probability problems that bridge the gap between big data analytics and data mining. It is important however to understand the critical steps involved in establishing a model capable of delivering real (ish) time predictive analytics models required in this project. To answer the question of being able to use predictive analytics as a security management tool in virtualised systems, that is predicting attacks in near to real time, the right data and the right algorithm needs to be used for predictive modelling. Previous sections of chapter 6 have answered the question by using machine-learning algorithms to illustrate how a predictive modelling, using various algorithms can be built and tested for predictive value. The use of time series data often arises when monitoring processes or tracking matrices as in our case. The difference between time series analytics and those discussed earlier in the chapter is that, time series analytics accounts for the fact that data points that are collected over time have internal structure variations that should be accounted for.

The following section discusses and illustrates the use of time series algorithms in building a model for real time predictive analytics.

5.4.3 Predictive Analytics Modelling using Time Series

Time series modelling uses statistical techniques to model and explain a time dependant series of data points. Time series is a sequence of series of numerical data points listed in time order taken at successive equally spaced points in time. For this part of model development we use time series forecasting, which is a process for using a model to generate predictions for future attacks based on the known past events. The reason for this is that time series data has a natural time ordering system that differs from other machine learning applications where each data point is an independent value of the expected outcome. The main objective of predictive

modelling is to estimate the value of an unknown variable. Time series predictive modelling is twofold:

- Obtain an understanding of the structure of the produced data
- Fit a model that forecasts, monitors and provides feedback and feedforward control

Time series in this project takes the form of machine learning and data mining approaches to model a time series predictive model by dependence through the additional inputs variables also referred to as lagged variables. To prepare the data for time series, all unnecessary attributes are removed and only the timestamp, total forward and backwards packets are used. Once the data is transformed, we apply multiple linear and non-linear regression algorithms such as support vector algorithms for regression and model trees. For model building using time series, we use WEKA to analyse and build the time series predictive model. The time series framework in WEKA takes machine learning and data mining approaches to model time series by transforming data into a form that a standard learning proposition algorithm can process to model trends and seasonality.

Since the dataset was generated at intervals over a period, the data used for the time series modelling was collected within a 24-hour period and it is represented in minutes and not in days or months. We apply regression algorithms to make continuous predictions. To carry out the time series modelling process we only use two variables; timestamp and total number of packets so the rest of the attributes are removed once the dataset is loaded into the modelling tool.

To prepare for time series predictive modelling we set the forecast parameter to ten times beyond the end of the training data. The confidence interval level at which the prediction is set is 95% but the period is automatically detected by the tool. We set the target values, which are the values from the data we wish to forecast to multiple targets in order to capture dependences

between the values. Two algorithms are used for time series forecasting. To use the parameters for linear regression and SMoreg algorithms, we set the time unites to 12-steps ahead. A time unit is the number of steps into the future we set the predictions to. We use the time stamp as the field in the data that holds the time we are forecasting and the periodicity is left at default because there is not enough date separation in the data being used. The results of the learned model based on the training data are shown in listing 6-18

```
 Total Fwd Packets:
SMOreg

weights (not support vectors):
 +       0.0018 * (normalized) ArtificialTimeIndex
 +       1.0428 * (normalized) Lag_ Total Fwd Packets-1
 +       0.001  * (normalized) Lag_ Total Fwd Packets-2
 +       0.001  * (normalized) Lag_ Total Fwd Packets-3
 +       0.0006 * (normalized) Lag_ Total Fwd Packets-4
 -       0.0005 * (normalized) Lag_ Total Fwd Packets-5
 +       0.0003 * (normalized) Lag_ Total Fwd Packets-6
 +       0.0005 * (normalized) Lag_ Total Fwd Packets-7
 +       0.0018 * (normalized) Lag_ Total Fwd Packets-8
 +       0.0002 * (normalized) Lag_ Total Fwd Packets-9
 +       0.0005 * (normalized) Lag_ Total Fwd Packets-10
 +       0      * (normalized) Lag_ Total Fwd Packets-11
 +       0.0001 * (normalized) Lag_ Total Fwd Packets-12
 -       0.0039 * (normalized) ArtificialTimeIndex^2
 +       0.0029 * (normalized) ArtificialTimeIndex^3
 +       0.0079 * (normalized) ArtificialTimeIndex*Lag_ Total Fwd Packets-1
 -       0.0013 * (normalized) ArtificialTimeIndex*Lag_ Total Fwd Packets-2
 -       0.0009 * (normalized) ArtificialTimeIndex*Lag_ Total Fwd Packets-3
 -       0.0005 * (normalized) ArtificialTimeIndex*Lag_ Total Fwd Packets-4
 +       0.0008 * (normalized) ArtificialTimeIndex*Lag_ Total Fwd Packets-5
 +       0.0003 * (normalized) ArtificialTimeIndex*Lag_ Total Fwd Packets-6
 +       0.0001 * (normalized) ArtificialTimeIndex*Lag_ Total Fwd Packets-7
 -       0.0009 * (normalized) ArtificialTimeIndex*Lag_ Total Fwd Packets-8
 +       0      * (normalized) ArtificialTimeIndex*Lag_ Total Fwd Packets-9
 -       0.0003 * (normalized) ArtificialTimeIndex*Lag_ Total Fwd Packets-10
 +       0.0002 * (normalized) ArtificialTimeIndex*Lag_ Total Fwd Packets-11
 -       0.0001 * (normalized) ArtificialTimeIndex*Lag_ Total Fwd Packets-12
 +       0
```

Listing 5-16 Time Series Forecasting

To illustrate the results in listing 6-18 we use the root-squared error, the root squared mean error and the results are shown in figure 6-24 below.

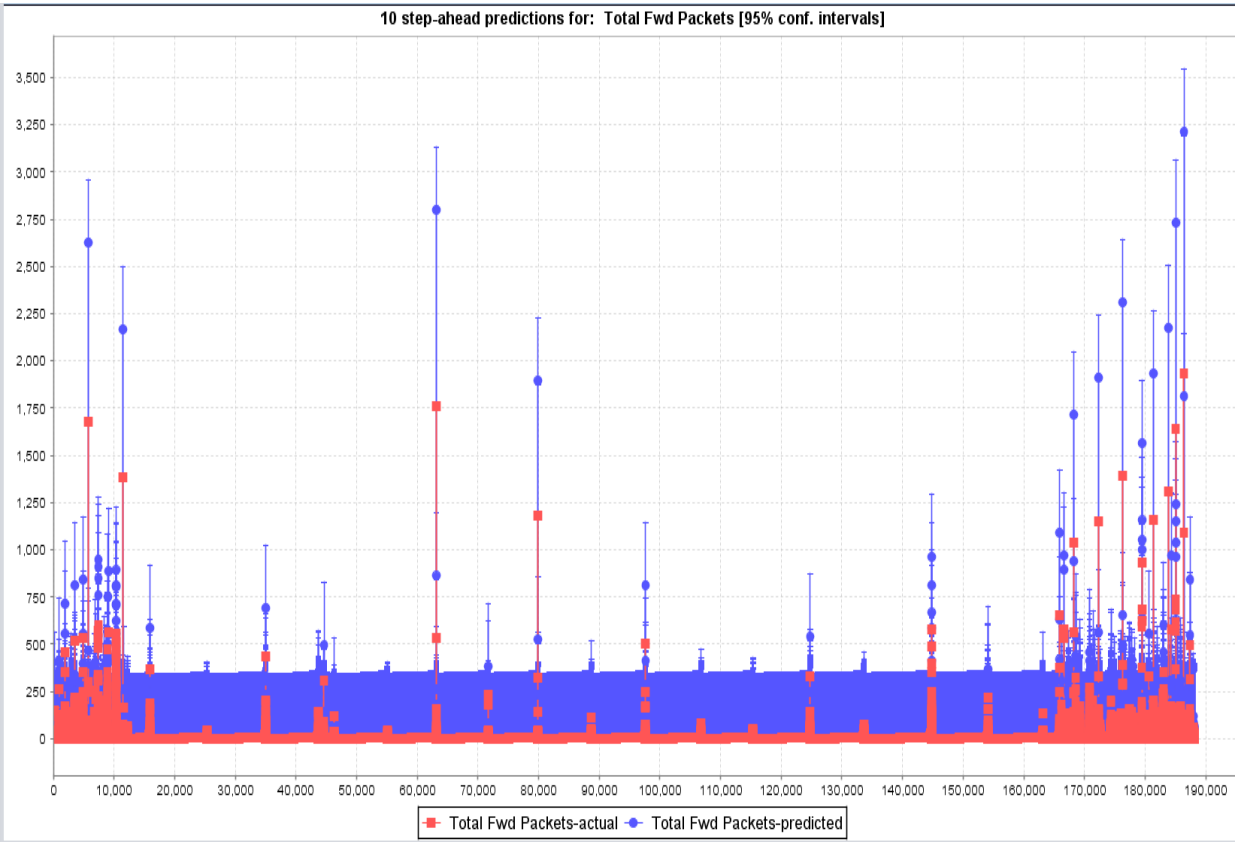


Figure 5-19 Time Series Forecasting using 10 step-ahead predictions

The output results of the actual predicted values for the target values at a single step of the 1-step-ahead prediction is listed in listing 6-19

```
=== Predictions for training data:  Total Fwd Packets (1-step ahead) ===

      inst#        actual     predicted                conf         error
         13             1        1.0165       -0.44:84.103        0.0165
         14             1        1.0166       -0.44:84.103        0.0166
         15             1        1.0166       -0.44:84.103        0.0166
         16             1        1.0166      -0.439:84.103        0.0166
         17             1        1.0166      -0.439:84.103        0.0166
         18             1        1.0166      -0.439:84.103        0.0166
         19             1        1.0166      -0.439:84.103        0.0166
         20             1        1.0167      -0.439:84.103        0.0167
         21             1        1.0167      -0.439:84.103        0.0167
         22             1        1.0167      -0.439:84.103        0.0167
         23             2        1.0167      -0.439:84.104       -0.9833
         24             2        2.0596       0.604:85.146        0.0596
         25             2        2.0606       0.605:85.147        0.0606
         26             2        2.0616       0.606:85.148        0.0616
         27             2        2.0622       0.606:85.149        0.0622
         28             2        2.0617       0.606:85.149        0.0617
         29             2        2.0621       0.606:85.149        0.0621
         30             2        2.0626       0.607:85.149        0.0626
         31             2        2.0644       0.608:85.151        0.0644
         32             2        2.0646       0.609:85.151        0.0646
         33             2        2.0652       0.609:85.152        0.0652
         34             2        2.0652       0.609:85.152        0.0652
         35             2        2.0654       0.609:85.152        0.0654
         36             2        2.0654       0.609:85.152        0.0654
         37             2        2.0654       0.609:85.152        0.0654
         38             2        2.0655       0.609:85.152        0.0655
         39             3        2.0655       0.609:85.152       -0.9345
         40             3        3.1083       1.652:86.195        0.1083
         41             5        3.1094       1.653:86.196       -1.8906
         42             6         5.196        3.74:88.283        -0.804
```

Listing 5-17 Time Series using 1-step-ahead Prediction

The output results of future predictions that are beyond the end of series of the training data, outputs both the training data and the predicted values of up to the maximum number of time units that are available. The results in listing 6-19 are illustrated in figure 6-25

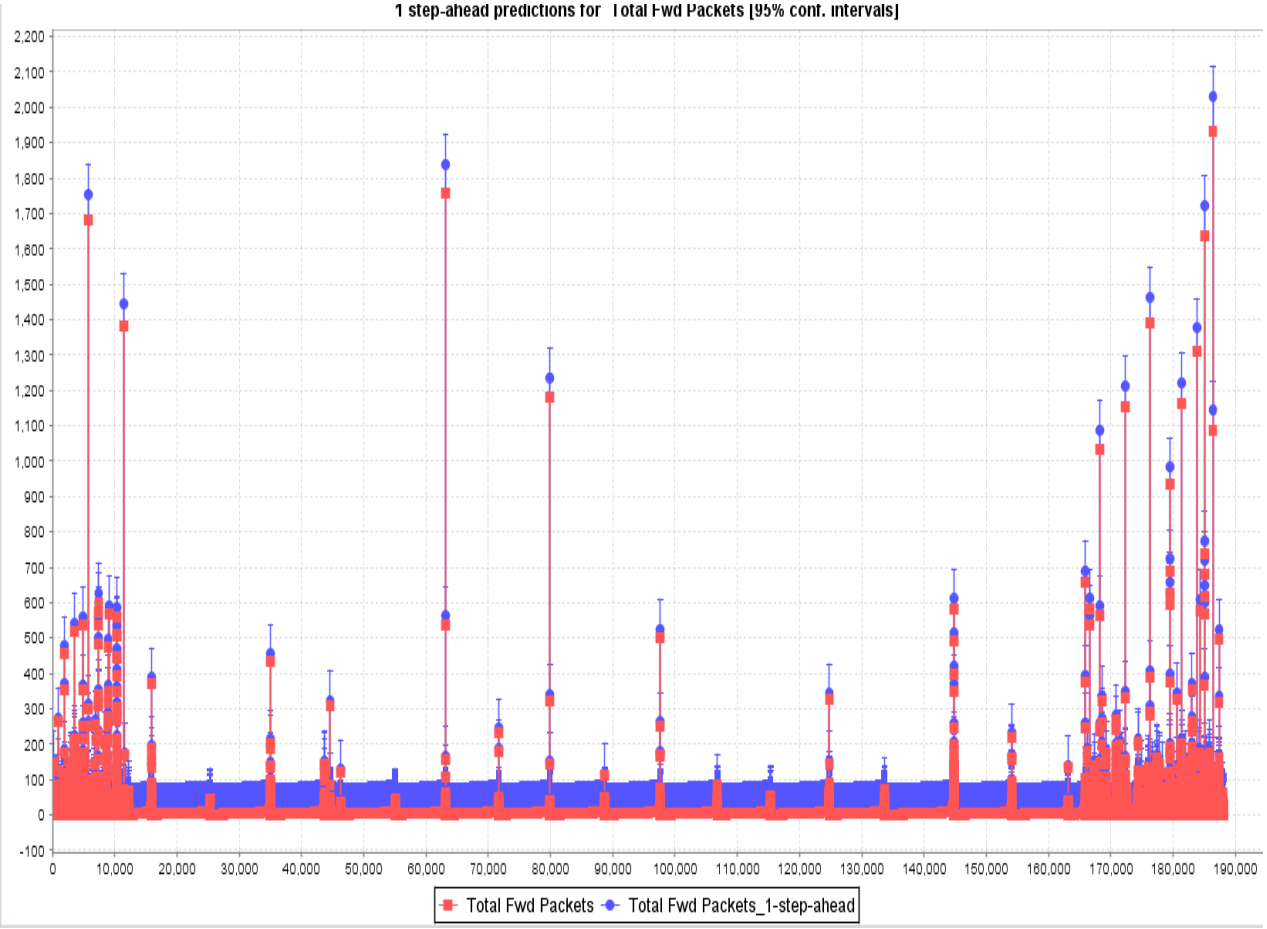


Figure 5-20 Time Series Forecasting using 1-step-ahead predictions

Using the 12 step ahead predictions produces lagged data values as illustrated in listing 6-20

```
Transformed training data:

               Tot Fwd Pkts=1
               ArtificialTimeIndex
               Lag_Tot Fwd Pkts=1-1
               Lag_Tot Fwd Pkts=1-2
               Lag_Tot Fwd Pkts=1-3
               Lag_Tot Fwd Pkts=1-4
               Lag_Tot Fwd Pkts=1-5
               Lag_Tot Fwd Pkts=1-6
               Lag_Tot Fwd Pkts=1-7
               Lag_Tot Fwd Pkts=1-8
               Lag_Tot Fwd Pkts=1-9
               Lag_Tot Fwd Pkts=1-10
               Lag_Tot Fwd Pkts=1-11
               Lag_Tot Fwd Pkts=1-12
               ArtificialTimeIndex^2
               ArtificialTimeIndex^3
               ArtificialTimeIndex*Lag_Tot Fwd Pkts=1-1
               ArtificialTimeIndex*Lag_Tot Fwd Pkts=1-2
               ArtificialTimeIndex*Lag_Tot Fwd Pkts=1-3
               ArtificialTimeIndex*Lag_Tot Fwd Pkts=1-4
               ArtificialTimeIndex*Lag_Tot Fwd Pkts=1-5
               ArtificialTimeIndex*Lag_Tot Fwd Pkts=1-6
               ArtificialTimeIndex*Lag_Tot Fwd Pkts=1-7
               ArtificialTimeIndex*Lag_Tot Fwd Pkts=1-8
               ArtificialTimeIndex*Lag_Tot Fwd Pkts=1-9
               ArtificialTimeIndex*Lag_Tot Fwd Pkts=1-10
               ArtificialTimeIndex*Lag_Tot Fwd Pkts=1-11
               ArtificialTimeIndex*Lag_Tot Fwd Pkts=1-12
```

Listing 5-18 Transformed Lagged data

A linear regression model of the time series prediction is illustrated in listing 6-21

```
Linear Regression Model

Tot Fwd Pkts=1 =

      0      * ArtificialTimeIndex +
      1.025  * Lag_Tot Fwd Pkts=1-1 +
     -0.0217 * Lag_Tot Fwd Pkts=1-12 +
     -0      * ArtificialTimeIndex^2 +
      0      * ArtificialTimeIndex^3 +
     -0      * ArtificialTimeIndex*Lag_Tot Fwd Pkts=1-1 +
      0      * ArtificialTimeIndex*Lag_Tot Fwd Pkts=1-12 +
     -0.0001
```

Listing 5-19 Time Series linear Predictive model

The time series linear predictive model is represented in figure 6.26

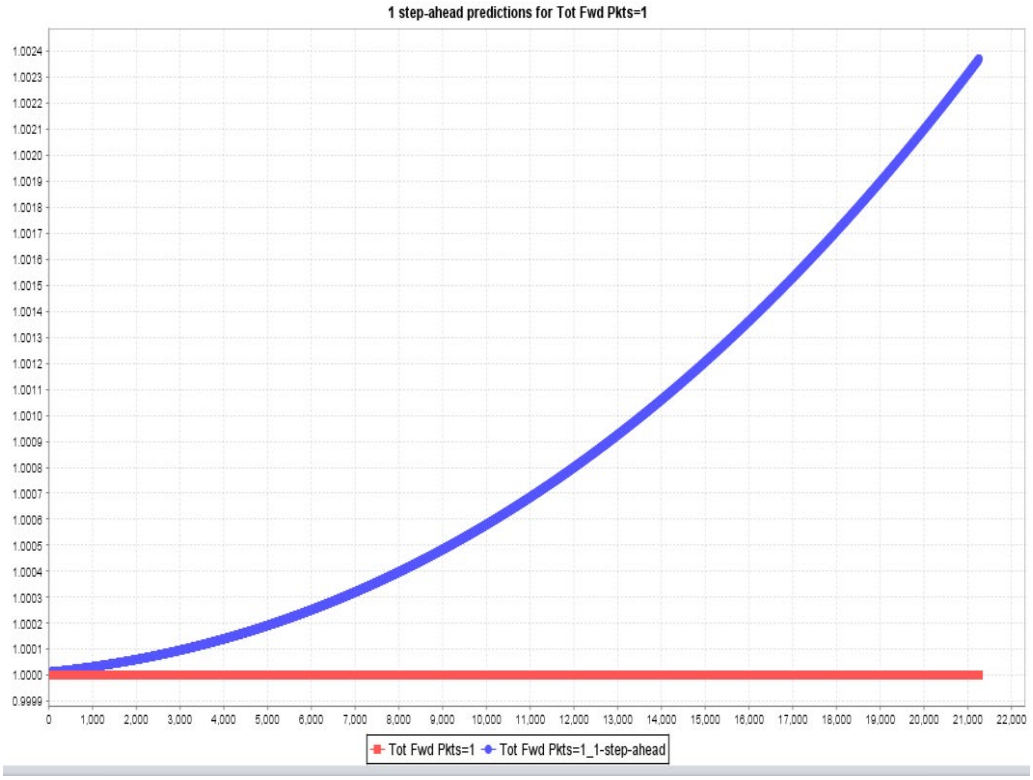


Figure 5-21 Time Series Linear predictive model using 1-step-ahead prediction

To put the built models to the test they need to be paired with real time systems in operation using a deployment process. Deployment is a method by which machine learning models are integrated into real time environment systems in order to make practical decisions based on the data. Model deployment is the last stage of the project where we take the built models and integrate them into the virtual environment to see if they can predict attacks. The following final section of this chapter discusses the many ways of deploying a model into a working environment.

5.5 Deployment of the Built Predictive Analytics Mode

Predictive model deployment provides the option to deploy the analytical results and the built model into everyday decision-making processes by automating the decision-making process. Model deployment in data science refers to application of a model for prediction into day to day decision making using new data (Mannan et al., 2019).

5.5.1 Deployment Methods

There are many different methods of deploying an artificial intelligence (AI) model into operation such as the use of

- data science tools or the cloud
- programming languages such as Java, C or visual basic
- databases and SQL scripts and
- Predictive Model Mark-up Language (PMML)

5.5.2 Implemented Deployment Method

We use PMML for the deployment of the built models. The reason for the choice of this deployment method is that PMML eliminates the need for custom model deployments and allows for clear separation of model development and model deployment tasks. PMML supports data science modelling methods such as regression, decision trees, clustering and time series used in this project. It allows for the explicit specification of valid, invalid and missing values which allows for the appropriate handling of missing and invalid values (Mannan et al., 2019). PMML has embedded functions for arithmetic expression, handling of data, time and strings used for implementing logic and Boolean problems. It allows for predictive modelling to be fully expressed and model outputs to be scaled.

PMML consists of

- The header, which contains general information about the PMML documents,
- Data dictionary that contains definitions for all possible attributes used in the model,
- Data transformation that allows data to be transformed into usable data for model building,
- Model information that contains the definition of the model and lastly
- The list of attributes used in building the predictive model.

5.5.3 Monitoring of the Deployed Predictive Model

Deployment and Model deployment and monitoring work hand in hand, and follow a step by process as illustrated in Figure 6-24.

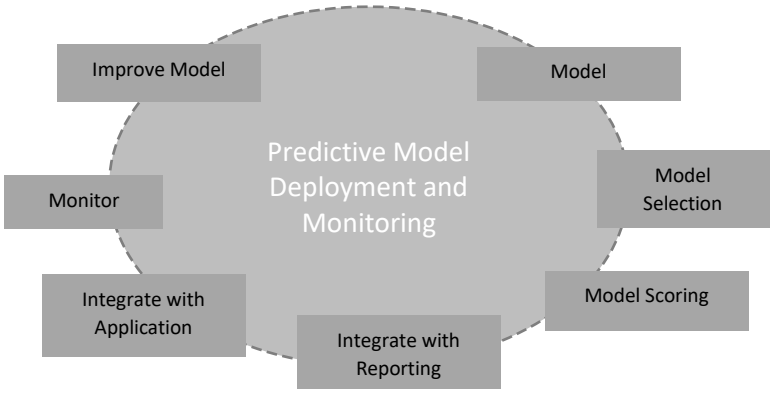


Figure 5-22 Predictive Model and Monitoring

Model deployment for the project has five phases:

- Deployment planning – which involves the initial deployment processes using Rs PMML package and python.
- Monitoring and Maintenance planning - involves setting the correct matrices to be monitored for predictive value as well as configuring the right response to the monitored matrices.
- Deployment – which involves the final evaluation and testing of the model by ensuring that all the model values meet the set predictive values
- Monitoring and Reporting – where the model is evaluated and tested with new data

Validation of predictive models is done to ensure that the predictors used do not have issues and for the validation of distribution, analytical algorithms and pre deployment scoring. Once the model is validated, it is then scored by applying it to new data that does not have dependent variables

5.6 Summary

In this chapter, we have illustrated the use of explanatory analytics to gain an in-depth understanding of the dataset generated in chapter 5. The chapter provides the use of various classification algorithms in the EDA such as feature selection, clustering of attributes of the dataset and the use of simple classification analytics processing using various algorithms to prepare the data for predictive analytics. The dataset is manipulated using various algorithms to ensure that the right attributes are identified, explained and presented using both scatter plots and box plots. The results obtained in this chapter demonstrate that the right attacks simulated in chapter 5 are the right classes depicted in EDA process. Three models (classification model, Clustering Model and forecasting models) are created and will be used in the next section; building the predictive modelling of the same chapter. Section 6.3 implements the findings in the EDA process by further cleaning and analysing the data models used to build the classification models. The data mining process in this part of the section was used to build and validate the model. Section 6.3 also demonstrates the use of various classification algorithms such as regression, decision trees and time series algorithms to achieve desired results; which is the use of predictive analytics as a security management tool in virtualised systems. The chapter has demonstrated the process of building predictive models using linear regression algorithms, which is then validated and tested using logistic and Naïve Bayes algorithms. The results of the built models together with the validation models are also presented. Predictive analytics modelling using time series is also illustrated in section 6.4.1 of the same chapter and the results in forms of graphs are presented. The model deployment process is also explained in section 6.5.

6. Conclusion

The need to contribute to the knowledge of existing solutions for effective virtualisation security in cloud computing, and the need to implement predictive analytics in empirical information technology as a security management tool for the virtualised environment, has been the main motivation for this project. With so many tools and techniques available for machine learning and deep learning it is often not easy to understand what defines predictive analytics, although most classification, statistical and probability problems bridge the gap. It is important however to understand the critical steps involved in establishing a model capable of delivering real (ish) time predictive analytics models required to achieve the predictive value for the project. To answer the question of using predictive analytics as a security management tool in virtualised systems that predict attacks in near to real time, the right data and the right algorithm need to be used for predictive modelling. The thesis has highlighted the right attributes required for predictive modelling and has demonstrated the use of machine learning using various algorithms to achieve predictive analytics modelling for the prediction of DoS attacks in virtualised systems.

Predictive analytics has been around for many years and research through the evaluation of many solutions, has proved that although empirical predictive analytics is an information technology process, its application to empirical information technology such as cloud computing technology, is very limited. Suggestions of how predictive analytics can be adopted into information systems such as cloud have been made and attempts provided by many researchers, but there is still no extensive use of predictive analytics in empirical information systems. As cloud computing is in need of robust, effective security solutions, virtualisation with its security benefits is being used to help eliminate some security concerns faced by cloud computing. Virtualisation however, does not only bring about security benefits but comes with its own security issues brought about by its strong properties. With the ever-evolving cyber

security attacks, it is imperative to find solutions for cloud computing that not only find attacks (such as IDS or VMI) after they have happened but also combat (PAM) them before they actually happen. This project illustrates how these virtualisation security issues can be managed by implementing predictive analytics as a security management tool as a novel contribution and in turn demonstrates the use of predictive analytics in by designing and implementing a predictive security model using different algorithms. The work covered in this project demonstrates how empirical predictive analytics can be adopted as a security management tool in virtualised environments by building and deploying various predictive models that predict attacks using real time big data analytics and a combination of algorithms.

6.1 Thesis Summary

Chapter 2 presents an overview of virtualisation and the types of architectures available. The chapter defines virtualisation as the simulation of software or hardware upon which other software runs. The chapter defines the simulated environment as a virtual machine or guest operating system, which is managed by the virtual machine monitor. Two architectural types: type I known as bare metal, which is directly installed onto the hardware and type II also referred to as hosted virtualisation that is installed right on top of the host operating system are identified and discussed. The types of virtualisation (full, storage, application and desktop virtualisation) and the security risks (transparent, insertion, introspection, intervention nonlinear VM operation and software decoupling) virtualisation poses to cloud computing together with the security implications from (control channels, data and software flows as well as non-VMM software) are also discussed in this chapter. The end of the chapter discusses the security vulnerabilities, threats and attacks to both the hypervisor and the virtual machine.

Chapter 3 provides a detailed survey of existing security solutions such as

- The Home Alone: CO-Residence in cloud via side channel analysis, which aims at verifying remotely that the tenants' virtual machines are physically isolated and have exclusive use of the physical machine.
- The Network Intrusion Detection Countermeasure Election (NICE) framework that is aimed at identifying malicious VMs on the network and lastly
- The Distributed Graph Lab cloud model an extension to MapReduce that uses machine learning and data mining that aims at modelling big data dependences

The literature reviewed suggests that although the above solutions claim the use of predictive analytics to combat security in their frameworks, there is still need for empirical predictive analytics as a security management tool as the reviewed solutions still use data mining techniques and handle security threats after the fact. The chapter also defines predictive analytics as the process for refining data using knowledge to extract hidden value from newly discovered patterns in comparison to data mining, which is the process for pattern recognition in large datasets to identify relationships. Examples of the differences between predictive analytics and data mining together with the tools and techniques for predictive modelling are presented.

Chapter 4 discusses the design structure and methodology followed in the project. The chapter presents an overview of the architecture or experimental environment, which has been set up with five components namely the API, reporting, analytical, monitoring, and data collection tools as well as the database. The predictive analytics process has six stages. The first stage is to understand the predictive goal intended for the project, then capture and understand the data, which is followed by the data preparation process, model design, model deployment and then lastly the model deployment process.

Chapter 5 covers the attack simulation process used for the collection and generation of the dataset used in predictive analytics. The chapter highlights the experimental environment, which is a decomposition of the architectural overview presented in chapter 4. The attack simulation process presented in the chapter has three parts: the attack part where the attack process is carried out, the detection component, which helps the attacks to be identified, and the response component where the attacks are being managed. Kali Linux is used to launch attacks and for intrusion detection using the built-in functions, Metasploitable with other virtual machines are used as attack victims or targets. The attacks being simulated are DoS and port scan attacks. The chapter demonstrates the data generating and collection process through the various attack simulating processes using different attack scenarios and attack vectors and presents results of each simulation. Sample dataset generated and imported into a spreadsheet is also shown in this chapter

Chapter 6 is the final experimental chapter of the thesis. It presents the predictive modelling process, which starts with the exploration of the collected data using various classification algorithms such as feature selection, clustering, and simple linear and non-linear algorithms. The predictive modelling process uses the data in the explanatory data process illustrated at the beginning of the chapter and builds various models using different algorithms such as linear regression models, which are validated and tested using naïve Bayes and logistic algorithms. Predictive analytics modelling using time series is also illustrated. To answer the question of the project, predictive models are built, validated and deployed as a security management tool in virtualised environments.

6.2 Contributions

The use of predictive analytics to combat various security issues of virtualisation has been attempted by many researchers as highlighted chapter 3. Although these researchers start with predictive analytics in mind they outcome still uses data mining techniques such as introspection and intrusion detection to find attacks after the fact. To move a step further from datamining, this research demonstrates how predictive analytics as a novel contribution, is used as a security management tool for virtualised information systems by building, validating and testing predictive models that accurately predict DoS attacks using new data and in real(ish) time using time series forecasting and prediction. The thesis has demonstrated novel contributions to the body of knowledge by:

- Designing and successfully implementing a novel predictive analytics modelling framework that was followed to successfully design, build, implement and build the predictive model in place.
- Successfully building implementing and deploying a predictive analytics model used for risk analysis and management of security threats in virtualised information systems using different algorithms such as linear regression algorithms used for building the PAM, logistic regression and Naïve Bayes algorithms for validating and testing the predictive accuracy of the built model and then using the built predictive analytics model to predict DoS attacks and then demonstrated the accuracy level of the built PAM in predicting new data in virtualised systems.
- Providing visible use of predictive analytics in multi-tenant systems by implementing a predictive analytics model as a security measure in virtualised environments and has therefore extended on Shmuel et al's idea of how empirical predictive analytics can be designed and implemented into information systems such as cloud computing in this case therefore contributing to existing theories on using predictive analytics to combat

security in virtualised systems by extending on the existing theories that suggest a different use of algorithms other than those used in datamining techniques by using predictive analytics algorithms.

- Demonstrating that attacks can be analysed and mitigated in real time by using time series forecasting and time series predictive analytics as opposed to data mining techniques that use Intrusion detection systems to find attacks after they have already happened.

6.3 Challenges and Limitations

Building and implementing a predictive analytics model as a security management tool for virtualisation systems came with a lot of challenges and limitations as explained below:

I. Due to the nature of the project which required a physical cloud environment to be built and implemented with multi-tenant system installed for the simulation of attacks, there was limited resources available, and this lead to designing and implementing a simulation environment using virtual box. Using a simulation environment inside a box came with challenges such as configuration problems, network limitations as there was a likelihood of attacks moving out of the simulation environment. There was a limitation of what could be simulated for this reason, hence DoS attacks.

II. The simulation of the attacks required for data collection, could not run for a longer period of time due to the limited system resources hence the limited time period in the dataset. The system was not robust enough to capture continuous data without being terminated.

III. Achieving the required results was also a challenge as time for running simulations was constrained due to unforeseen circumstances brought about by an ongoing illness.

IV. The built models could not be deployed into the cloud due to limited resources used as it required a real time cloud environment to deploy. The built model could not be deployed into the virtual environment as time was constrained as this required time to program the

deployment using the PMML, attach the model to the cloud, test and continuous monitoring of the performance of the model in real life.

6.4 Further Work

To successfully achieve predictive analytics as a security management tool for virtualised system, model deployment, testing and monitoring in real time needs to be accomplished. Future work includes:

I. Further testing of the predictive value of the built models using different algorithms is necessary. This will help validate the built models for easy deployment of the same into a real time cloud system.

II. Deployment of the built models by incorporating the model into a working virtual environment of the cloud using PMML. This involves an iterative deployment process of fine tuning, testing and validation of the model for predictive accuracy using real time data.

III. Using different predictive analytical tools for predictive modelling, testing, validation, visualisation and mode deployment. Since an analytical tool was used in this project for analytics and visualisation using an API such as Python and R to build a predictive model and visualise the results could make deployment of the model easier to manage as programming and fine-tuning of the model could be made within the APIs used.

IV. Using different methods for data collection methods to improve the quality of the dataset being used.

V. Using different monitoring techniques that will not require an agent being installed on the monitored virtual machines.

VI. Extending the use of predictive analytics as a security management for all security threats of virtualisation in cloud computing rather than DoS attacks as implemented in this thesis. This could be achieved by simulating a number of attacks aimed at

virtualisation from different attack vectors like the hypervisor or virtual machines or even the host and collecting the data for predictive modelling.

References

ALMUTAIRY, N. M., AL-SHQEERAT, K. H. & AL HAMAD, H. A. 2019. A Taxonomy of Virtualization Security Issues in Cloud Computing Environments. *Indian Journal of Science and Technology,* 12**,** 3.

AMINI, A. & JAMIL, N. A comprehensive review of existing risk assessment models in cloud computing. Journal of Physics: Conference Series, 2018. IOP Publishing Ltd., 012004.

ARMSTRONG, B., ENGLAND, P., FIELD, S. A., GARMS, J., KRAMER, M. & RAY, K. D. 2008. Computer security management, such as in a virtual machine or hardened operating system. Google Patents.

AZMANDIAN, F., MOFFIE, M., ALSHAWABKEH, M., DY, J., ASLAM, J. & KAELI, D. 2011. Virtual machine monitor-based lightweight intrusion detection. *ACM SIGOPS Operating Systems Review,* 45**,** 38-53.

BRATUS, S., LOCASTO, M. E., RAMASWAMY, A. & SMITH, S. W. Traps, events, emulation, and enforcement: managing the yin and yang of virtualization-based security. Proceedings of the 1st ACM workshop on Virtual machine security, 2008. ACM, 49-58.

BROHI, S. N., BAMIAH, M. A., BROHI, M. A. & KAMRAN, R. 2012. Identifying and analyzing security threats to virtualized cloud computing infrastructures. *Cloud Computing Technologies, Applications and Management 2012 International Conference on Cloud Computing Technologies, Applications and Management (ICCCTAM).*

BROOKS, T., CAICEDO, C. & PARK, J. Security challenges and countermeasures for trusted virtualized computing environments. World Congress on Internet Security (WorldCIS-2012), 2012. IEEE, 117-122.

CARROLL, M., KOTZÉ, P. & VAN DER MERWE, A. 2011. Secure virtualization: benefits, risks and constraints.

CHUNG, C.-J., KHATKAR, P., XING, T., LEE, J. & HUANG, D. 2013. NICE: Network intrusion detection and countermeasure selection in virtual network systems. *IEEE transactions on dependable and secure computing,* 10**,** 198-211.

COX, V. 2017. Exploratory data analysis. *Translating Statistics to Make Decisions.* Springer.

CRISCIONE, C. 2010. Virtually Pwned Pentesting Virtualization. *Proceedings of Blackhat USA*.

DILDAR, M. S., KHAN, N., ABDULLAH, J. B. & KHAN, A. S. Effective way to defend the hypervisor attacks in cloud computing. 2017 2nd International Conference on Anti-Cyber Crimes (ICACC), 2017. IEEE, 154-159.

DOWTY, M. & SUGERMAN, J. 2009. GPU virtualization on VMware's hosted I/O architecture. *ACM SIGOPS Operating Systems Review,* 43**,** 73-82.

ECKERSON, W. W. 2007. Predictive analytics. *Extending the Value of Your Data Warehousing Investment. TDWI Best Practices Report,* 1**,** 1-36.

ELKAN, C. 2013. *Predictive analytics and data mining*, University of California.

FUENTES, A. 2018. *Hands-On Predictive Analytics with Python: Master the complete predictive analytics process, from problem definition to model deployment*, Packt Publishing Ltd.

GANDOMI, A. & HAIDER, M. 2015. Beyond the hype: Big data concepts, methods, and analytics. *International journal of information management,* 35**,** 137-144.

GARFINKEL, T., ADAMS, K., WARFIELD, A. & FRANKLIN, J. Compatibility Is Not Transparency: VMM Detection Myths and Realities. HotOS, 2007.

GARFINKEL, T. & ROSENBLUM, M. When Virtual Is Harder than Real: Security Challenges in Virtual Machine Based Computing Environments. HotOS, 2005.

GE, Q., YAROM, Y., COCK, D. & HEISER, G. 2018. A survey of microarchitectural timing attacks and countermeasures on contemporary hardware. *Journal of Cryptographic Engineering,* 8**,** 1-27.

GEBHARDT, C., DALTON, C. I. & BROWN, R. 2008. Preventing hypervisor-based rootkits with trusted execution technology. *Network Security,* 2008**,** 7-12.

GEBHARDT, C., DALTON, C. I. & TOMLINSON, A. 2010. Separating hypervisor trusted computing base supported by hardware. *Proceedings of the fifth ACM workshop on scalable trusted computing.* ACM.

GEBHARDT, C. & TOMLINSON, A. 2008. Security consideration for virtualization. *University of London, Tech. Rep. RHUL-MA-2008-16*.

GUIZANI, N. & GHAFOOR, A. 2020. A Network Function Virtualization System for Detecting Malware in Large IoT Based Networks. *IEEE Journal on Selected Areas in Communications,* 38**,** 1218-1228.

GULATI, H. Predictive analytics using data mining technique. 2015 2nd International Conference on Computing for Sustainable Global Development (INDIACom), 2015. IEEE, 713-716.

HONG, C.-H., SPENCE, I. & NIKOLOPOULOS, D. S. 2017. GPU virtualization and scheduling methods: A comprehensive survey. *ACM Computing Surveys (CSUR),* 50**,** 35.

IM, J., KIM, J., KIM, J., JIN, S. & MAENG, S. On-demand virtualization for live migration in bare metal cloud. Proceedings of the 2017 Symposium on Cloud Computing, 2017. ACM, 378-389.

KAPASA, R. M., FORSYTH, H., LAWS, A. & LEMPEREUR, B. Predictive Analytics as a Security Management Tool in Virtualised Environment. 2015 International Conference on Developments of E-Systems Engineering (DeSE), 2015. IEEE, 102-106.

KARGER, P. A. & SAFFORD, D. R. 2008. I/O for virtual machine monitors: Security and performance issues. *IEEE Security & Privacy,* 6**,** 16-23.

KELLEHER, J. D., MAC NAMEE, B. & D'ARCY, A. 2015. *Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies*, MIT Press.

KENNEDY, H. & ALLEN, W. 2017. Data visualisation as an emerging tool for online research. *The SAGE Handbook of Online Research Methods***,** 307-26.

KHALIL, I., KHREISHAH, A. & AZEEM, M. 2014. Cloud computing security: A survey. *Computers,* 3**,** 1-35.

KHAN, I., ANWAR, Z., BORDBAR, B., RITTER, E. & REHMAN, H.-U. 2016. A Protocol for Preventing Insider Attacks in Untrusted Infrastructure-as-a-Service Clouds. *IEEE Transactions on Cloud Computing,* 6**,** 942-954.

KUMARA, A. & JAIDHAR, C. Hypervisor and virtual machine dependent Intrusion Detection and Prevention System for virtualized cloud environment. 2015 1st International Conference on Telematics and Future Generation Networks (TAFGEN), 2015. IEEE, 28-33.

LAROSE, D. T. 2015. *Data mining and predictive analytics*, John Wiley & Sons.

LEE, S.-W. & YU, F. Securing KVM-based cloud systems via virtualization introspection. 2014 47th Hawaii International Conference on System Sciences, 2014a. IEEE, 5028-5037.

LEE, S.-W. & YU, F. Securing KVM-based cloud systems via virtualization introspection. 2014 47th Hawaii International Conference on System Sciences, 2014b. IEEE, 5028-5037.

LI, J., LI, B., WO, T., HU, C., HUAI, J., LIU, L. & LAM, K. 2012. CyberGuarder: A virtualization security assurance architecture for green cloud computing. *Future Generation Computer Systems,* 28**,** 379-390.

LIU, S., WANG, X., LIU, M. & ZHU, J. 2017. Towards better analysis of machine learning models: A visual analytics perspective. *Visual Informatics,* 1**,** 48-56.

LOGANAYAGI, B. & SUJATHA, S. 2012. Enhanced cloud security by combining virtualization and policy monitoring techniques. *Procedia Engineering,* 30**,** 654-661.

LOMBARDI, F. & DI PIETRO, R. 2011. Secure virtualization for cloud computing. *Journal of network and computer applications,* 34**,** 1113-1122.

LOW, Y., BICKSON, D., GONZALEZ, J., GUESTRIN, C., KYROLA, A. & HELLERSTEIN, J. M. 2012. Distributed GraphLab: a framework for machine learning and data mining in the cloud. *Proceedings of the VLDB Endowment,* 5**,** 716-727.

LUO, X., YANG, L., MA, L., CHU, S. & DAI, H. Virtualization security risks and solutions of Cloud Computing via divide-conquer strategy. 2011 Third International Conference on Multimedia Information Networking and Security, 2011. IEEE, 637-641.

MANNAN, M. A., MEHMOOD, S. & SHAFIQ, M. 2019. Data Management and Visualization Using Big Data Analytics. *Data Science and Digital Business.* Springer.

MAVROGIORGOU, A., KIOURTIS, A. & KYRIAZIS, D. A comparative study of classification techniques for managing iot devices of common specifications. International Conference on the Economics of Grids, Clouds, Systems, and Services, 2017. Springer, 67-77.

MCDANIEL, L. & NANCE, K. Identifying weaknesses in VM/hypervisor interfaces. 2013 46th Hawaii International Conference on System Sciences, 2013. IEEE, 5089-5095.

MEHROTRA, R., DUBEY, A., ABDELWAHED, S. & ROWLAND, K. W. 2011. Rfdmon: A real-time and fault-tolerant distributed system monitoring approach. *ISIS,* 11**,** 107.

MISHRA, M. & SAHOO, A. On theory of vm placement: Anomalies in existing methodologies and their mitigation using a novel vector based approach. IEEE CLOUD, 2011. Citeseer, 275-282.

MISHRA, P., PILLI, E. S., VARADHARAJAN, V. & TUPAKULA, U. Out-VM monitoring for malicious network packet detection in cloud. 2017 ISEA asia security and privacy (ISEASP), 2017a. IEEE, 1-10.

MISHRA, P., PILLI, E. S., VARADHARAJAN, V. & TUPAKULA, U. Out-VM monitoring for malicious network packet detection in cloud. 2017 ISEA asia security and privacy (ISEASP), 2017b. IEEE, 1-10.

MÜLLER, O., JUNGLAS, I., BROCKE, J. V. & DEBORTOLI, S. 2016. Utilizing big data analytics for information systems research: challenges, promises and guidelines. *European Journal of Information Systems,* 25**,** 289-302.

NEAL, C. 2013. *Forensic recovery of evidence from deleted Oracle VirtualBox virtual machines.* Utica College.

ORMANDY, T. 2007. An empirical study into the security exposure to hosts of hostile virtualized environments. Citeseer.

PADHY, R. P., PATRA, M. R. & SATAPATHY, S. C. 2011. Cloud computing: security issues and research challenges. *International Journal of Computer Science and Information Technology & Security (IJCSITS),* 1**,** 136-146.

PATIL, T. R. & SHEREKAR, S. 2013. Performance analysis of Naive Bayes and J48 classification algorithm for data classification. *International journal of computer science and applications,* 6**,** 256-261.

PAYNE, B. D., CARBONE, M., SHARIF, M. & LEE, W. Lares: An architecture for secure active monitoring using virtualization. 2008 IEEE Symposium on Security and Privacy (sp 2008), 2008. IEEE, 233-247.

PEARCE, M., ZEADALLY, S. & HUNT, R. 2013. Virtualization: Issues, security threats, and solutions. *ACM Computing Surveys (CSUR),* 45**,** 17.

PLATT, M. 2002. Microsoft architecture overview. *Retrieved March,* 11**,** 2010.

POPEK, G. J. & GOLDBERG, R. P. 1974. Formal requirements for virtualizable third generation architectures. *Communications of the ACM,* 17**,** 412-421.

RAJ, H. & SCHWAN, K. High performance and scalable I/O virtualization via self-virtualized devices. Proceedings of the 16th international symposium on High performance distributed computing, 2007. ACM, 179-188.

REN, Y., LIU, L., ZHANG, Q., WU, Q., GUAN, J., KONG, J., DAI, H. & SHAO, L. 2016. Shared-memory optimizations for inter-virtual-machine communication. *ACM Computing Surveys (CSUR),* 48**,** 49.

RUEDA, S., SREENIVASAN, Y. & JAEGER, T. Flexible security configuration for virtual machines. Proceedings of the 2nd ACM workshop on Computer security architectures, 2008. 35-44.

SABAHI, F. 2012. Secure virtualization for cloud environment using hypervisor-based technology. *International Journal of Machine Learning and Computing,* 2**,** 39.

SAHOO, J., MOHAPATRA, S. & LATH, R. Virtualization: A survey on concepts, taxonomy and associated security issues. 2010 Second International Conference on Computer and Network Technology, 2010. IEEE, 222-226.

SANFILIPPO, A. Workshop on current issues in predictive approaches to intelligence and security analytics. 2010 IEEE International Conference on Intelligence and Security Informatics, 2010. IEEE, 177-178.

SAÑUDO, I., CAVICCHIOLI, R., CAPODIECI, N., VALENTE, P. & BERTOGNA, M. 2018. A survey on shared disk I/O management in virtualized environments under real time constraints. *ACM SIGBED Review,* 15**,** 57-63.

SAPSFORD, R. & JUPP, V. 1996a. *Data collection and analysis*, Sage.

SAPSFORD, R. & JUPP, V. 1996b. Validating evidence. *Data collection and analysis***,** 1-24.

SAROHA, R. 2014. NICE: Network Intrusion Detection and Countermeasure selection in Virtual Network Systems. *International Journal of Science, Engineering and Computer Technology,* 4**,** 158.

SHMUELI, G. 2010. To explain or to predict? *Statistical science,* 25**,** 289-310.

SHMUELI, G. & KOPPIUS, O. R. 2011. Predictive analytics in information systems research. *MIS quarterly***,** 553-572.

SIMMA, A., EPPLER, J. & LANG, B. Hands-on Learning of Computer Security: A Cost-effective Laboratory Infrastructure Based on Virtualization Software. European Conference on Cyber Warfare and Security, 2015. Academic Conferences International Limited, 276.

SINGH, R., SHENOY, P., NATU, M., SADAPHAL, V. & VIN, H. 2013. Analytical modeling for what-if analysis in complex cloud computing applications. *ACM SIGMETRICS Performance Evaluation Review,* 40**,** 53-62.

SOUPPAYA, M. P., SCARFONE, K. & HOFFMAN, P. 2011. Guide to Security for Full Virtualization Technologies.

SUBASHINI, S. & KAVITHA, V. 2011. A survey on security issues in service delivery models of cloud computing. *Journal of network and computer applications,* 34**,** 1-11.

SUGERMAN, J., VENKITACHALAM, G. & LIM, B.-H. Virtualizing I/O Devices on VMware Workstation's Hosted Virtual Machine Monitor. USENIX Annual Technical Conference, General Track, 2001. 1-14.

SURESH, I. & KANNAN, M. 2014. A study on system virtualization techniques. *International Journal of Advanced Research in Computer Science and Technology,* 2**,** 134-139.

SZEFER, J., KELLER, E., LEE, R. B. & REXFORD, J. Eliminating the hypervisor attack surface for a more secure cloud. Proceedings of the 18th ACM conference on Computer and communications security, 2011. 401-412.

TAJ, M. S., ULLAH, S. I., SALAM, A. & KHAN, W. U. 2020. Enhancing Anomaly Based Intrusion Detection Techniques for Virtualization in Cloud Computing Using Machine Learning. *International Journal of Computer Science and Information Security (IJCSIS),* 18.

VAN CLEEFF, A., PIETERS, W. & WIERINGA, R. J. Security implications of virtualization: A literature study. 2009 International conference on computational science and engineering, 2009. IEEE, 353-358.

VASUDEVAN, A., CHAKI, S., JIA, L., MCCUNE, J., NEWSOME, J. & DATTA, A. Design, implementation and verification of an extensible and modular hypervisor framework. 2013 IEEE Symposium on Security and Privacy, 2013. IEEE, 430-444.

WATTS, D. D. 2016. *Classifying Discoveries: Implementing a Generalized Multiple Testing Protocol for Exploratory Data Analysis.* Oklahoma State University.

WOJTCZUK, R. & RUTKOWSKA, J. 2009. Attacking intel trusted execution technology. *Black Hat DC,* 2009**,** 1-6.

WOJTCZUK, R. & RUTKOWSKA, J. 2011. Following the White Rabbit: Software attacks against Intel VT-d technology. *ITL: http://www. invisiblethingslab. com/resources/2011/Software% 20Attacks% 20on% 20Intel% 20VT-d. pdf.*

WON, S., CHO, I., SUDUSINGHE, K., XU, J., ZHANG, Y., VAN DER SCHAAR, M. & BHATTACHARYYA, S. S. 2013. A design methodology for distributed adaptive stream mining systems. *Procedia Computer Science,* 18**,** 2482-2491.

YANG, W. & FUNG, C. A survey on security in network functions virtualization. 2016 IEEE NetSoft Conference and Workshops (NetSoft), 2016. IEEE, 15-19.

ZACHAR, F. IRC control bot for Zabbix monitoring system.

ZHANG, X., WUWONG, N., LI, H. & ZHANG, X. Information security risk management framework for the cloud computing environments. 2010 10th IEEE international conference on computer and information technology, 2010. IEEE, 1328-1334.

ZHANG, Y., JUELS, A., OPREA, A. & REITER, M. K. Homealone: Co-residency detection in the cloud via side-channel analysis. 2011 IEEE symposium on security and privacy, 2011. IEEE, 313-328.

ZHANG, Y., JUELS, A., REITER, M. K. & RISTENPART, T. Cross-VM side channels and their use to extract private keys. Proceedings of the 2012 ACM conference on Computer and communications security, 2012. ACM, 305-316.

ZHAO, L. & MANNAN, M. Hypnoguard: Protecting secrets across sleep-wake cycles. Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, 2016. ACM, 945-957.

ZOU, D., ZHANG, W., QIANG, W., XIANG, G., YANG, L. T., JIN, H. & HU, K. 2013. Design and implementation of a trusted monitoring framework for cloud platforms. *Future Generation Computer Systems,* 29**,** 2092-2102.