STATUS OF THESIS

Title of thesis | BIOLOGICAL INSPIRED INTRUSION PREVENTION AND SELF-HEALING SYSTEM FOR CRITICAL SERVICES NETWORK

I, MUNA ELSADIG MOHAMED AHMED ELSHEIK

hereby allow my thesis to be placed at the Information Resource Center (IRC) of University Teknologi PETRONAS (UTP) with the following conditions:

1. The thesis becomes the property of UTP
2. The IRC of UTP may make copies of the thesis for academic purposes only.
3. This thesis is classified as

☐ Confidential

☑ Non-confidential

If this thesis is confidential, please state the reason:

_____

_____

_____

The content of the thesis will remain confidential for years.

Remarks on disclosure:

_____

_____

_____

Endorsed by

_____

Signature of Author

Muna Elsadig Mohamed

Aberoof, Wad-Elbana,

Block 190/101

Sudan, Omdurman

Date: _____

_____

Signature of Supervisor

Assoc. Prof. Dr. Azween Bin Abdullah

Department of Computer and Information Science

Universiti Teknologi PETRONAS

Date: _____

UNIVERSITI TEKNOLOGI PETRONAS

BIOLOGICAL INSPIRED INTRUSION PREVENTION AND SELF-HEALING SYSTEM FOR CRITICAL SERVICES NETWORK

by

MUNA ELSADIG MOHAMED AHMED ELSHEIK

The undersigned certify that they have read, and recommend to the Postgraduate Studies Programme for acceptance this thesis for the fulfillment of the requirements for the degree stated.

Signature: _____

Main Supervisor: Assoc.Prof. Dr. Azween Bin Abdullah

Signature: _____

Co-Supervisor: Dr.Brahim Belhaouari Samir

Signature _____

Head of Department: Dr. Mohd Fadzil Bin Hassan

Date: _____

BIOLOGICAL INSPIRED INTRUSION PREVENTION AND SELF-HEALING SYSTEM FOR CRITICAL SERVICES NETWORK

by

MUNA ELSADIG MOHAMED AHMED ELSHEIK

A Thesis

Submitted to the Postgraduate Studies Programme

as a Requirement for the Degree of

DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER AND INFORMATION SCIENCE

UNIVERSITI TEKNOLOGI PETRONAS

BANDAR SERI ISKANDAR

PERAK

MARCH 2011

DECLARATION OF THESIS

Title of thesis | BIOLOGICAL INSPIRED INTRUSION PREVENTION AND SELF-HEALING SYSTEM FOR CRITICAL SERVICES NETWORK

I, MUNA ELSADIG MOHAMED AHMED ELSHEIK

hereby declare that the thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at UTP or other institutions.

Witnessed by

Signature of Author

Muna Elsadig Mohamed

Aberoof, Wad-Elbana,

Block 190/101

Sudan, Omdurman

Date: _____

Signature of Supervisor

Assoc. Prof. Dr. Azween Bin Abdullah

Department of Computer and Information Science

Universiti Teknologi PETRONAS

Date: _____

To my late father, may the mercy of Allah be upon him,
my mother, my sisters and my brothers with love

ACKNOWLEDGEMENTS

First and foremost, I would like to thank ALLAH the Almighty, for without His consent, it would be impossible to achieve what has been done in this work, for giving me the strength and persistence to keep going without losing hope in scientific research even during the most difficult moments. May Allah accept this work, count it as a good deed and make it useful.

I am deeply indebted to many people whose help and support have made this thesis possible. I would like to express my profound thanks and gratitude to my thesis advisor, Assoc. Professor Azween bin Abdullah, and to my co-supervisor, Dr. Brahim Belhaouari Samir, for their insightful guidance, unlimited help and for this wonderful opportunity of a challenging research.

I am grateful to my colleagues in the Computer Science and Information Department, especially to postgraduate students: Shah Syed Nasir, Mythim Kamal, Intan Najua, Mohamed Abdee and members of the Hybrid Intelligence and Self-healing System research group for the long discussions that had provided me with enlightening views of the research problems that I was struggling with.

I am also privileged to have the opportunity to collaborate with some of the most brilliant friends: Lina Al khateib, Aisha Osman, Nuha Husein, Salah, KhalidHaroun, Dr. Asim, Amel, Najla, Nuha Elshafee, Fatma Mabrookand Khalid Abdelbast.

I would also like to thank all my fellow country mates and international students at the University Teknologi Petronas for their endless support and spirit of brotherhood. I am mostly indebted to my teacher Ms Maryam Al Khatieb for caring for me.

In particular, I would like to express my sincere appreciation to my mother for her support and blessings, you are always my inspiration. Many thanks also to my sisters and brothers for their support and love. Last but not least, my appreciation is extended to Professor Abdel Aziz Abdullah and Gaida Abdurhman for their help as reviewers of this thesis and for their valuable comments.

ABSTRACT

With the explosive development of the critical services network systems and Internet, the need for networks security systems have become even critical with the enlargement of information technology in everyday life. Intrusion Prevention System (IPS) provides an in-line mechanism focus on identifying and blocking malicious network activity in real time. This thesis presents new intrusion prevention and self-healing system (SH) for critical services network security. The design features of the proposed system are inspired by the human immune system, integrated with pattern recognition nonlinear classification algorithm and machine learning. Firstly, the current intrusions preventions systems, biological innate and adaptive immune systems, autonomic computing and self-healing mechanisms are studied and analyzed. The importance of intrusion prevention system recommends that artificial immune systems (AIS) should incorporate abstraction models from innate, adaptive immune system, pattern recognition, machine learning and self-healing mechanisms to present autonomous IPS system with fast and high accurate detection and prevention performance and survivability for critical services network system. Secondly, specification language, system design, mathematical and computational models for IPS and SH system are established, which are based upon nonlinear classification, prevention predictability trust, analysis, self-adaptation and self-healing algorithms. Finally, the validation of the system carried out by simulation tests, measuring, benchmarking and comparative studies. New benchmarking metrics for detection capabilities, prevention predictability trust and self-healing reliability are introduced as contributions for the IPS and SH system measuring and validation.

Using the software system, design theories, AIS features, new nonlinear classification algorithm, and self-healing system show how the use of presented systems can ensure safety for critical services networks and heal the damage caused by intrusion. This autonomous system improves the performance of the current intrusion prevention system and carries on system continuity by using self-healing mechanism.

ABSTRAK

Perkembangan pesat sistem perkhidmatan rangkaian dan Internet pada hari ini memerlukan sistem keselamatan rangkaian yang kritikal dengan kecanggihan teknologi maklumat. Sistem Pencegah Intrusi (IPS) menyediakan penyelarasan mekanisme bertujuan mengecam dan menyekat aktiviti rangkaian berbahaya secara nyata. Tesis ini menyajikan pencegahan intrusinew dan sistem pemulihan sendiri (SH) untuk keselamatan rangkaian perkhidmatan yang kritikal. Cadangan rekabentuk sistem diilhamkan oleh sistem kekebalan tubuh manusia, diintegrasikan dengan algoritma pengenalan pola klasifikasi tidak linier dan mesin pembelajaran. Pertama, sistem pencegahan intrusi saat ini, biologi bawaan dan sistem penyesuaian imun, pengkomputeran autonomi dan mekanisme pemulihan sendiri dipelajari dan dianalisis. Pentingnya IPStelah mengesyorkan sistem kekebalan tubuh buatan (AIS) memerlukan model abstraksi dari bawaan, sistem penyesuaian imun, pengenalan pola, mesin pembelajaran dan mekanisma pemulihan sendiri untuk mempresentasikan sistem autonomi IPS dengan pengesanan yang cepat dan ketepatan yang tinggi dan prestasi pencegahan dan ketahanan untuk sistem rangkaian perkhidmatan yang kritikal. Kedua, spesifikasi bahasa, sistem rekabentuk, model matematik dan pengkomputeran untuk IPS dan sistem SH ditetapkan bersandarkan pada klasifikasi tidak linier, kepercayaan pencegahan meramal, analisis, adaptasi diri dan algoritma pemulihan sendiri. Akhirnya, pengesahan sistem yang dilakukan oleh ujian simulasi, pengukuran, perbandingan dan kajian banding. New perbandingan metrik kemampuan pengesanan, kepercayaan pencegahan meramal dan kehandalan penyembuhan diri diperkenalkan sebagai sumbangan untuk pengukuran sistem dan pengesahan IPS dan SH.

Penggunaan sistem perisian, teori rekaan, ciri AIS, algoritma klasifikasi new tidak linier, dan sistem pemulihan sendiri menunjukkan kegunaan sistem yang disajikan memastikan keselamatan rangkaian perkhidmatan yang kritikal dan menyembuhkan kerosakan berpunca daripada intrusi. Sistem autonomi ini meningkatkan prestasi sistem pencegahan intrusi saat ini dan menjalankan kelangsungan sistem dengan menggunakan mekanisma penyembuhan diri.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

Symbol	Definition
$\in$	element of
$\exists$	There exists
::	Defined as
$\forall$	For all
$\xrightarrow{Train}$	Training state of dataset
$\xrightarrow{sense}$	Sense state of input data
$\xrightarrow{prevent}$	Prevention state when abnormal behavior is sensed
$\xrightarrow{block}$	Block event the abnormal behavior destination
$\xrightarrow{continue}$	Continue receiving data from destination in normal behavior
$\xrightarrow{scan}$	Scanning data set
$\xrightarrow{receive}$	Receiving message state
$\triangleleft\downarrow$	Event of receiving in data
$\xrightarrow{analyze}$	Analyzing state of the abnormal behavior
$\xrightarrow{send}$	Sending message
$\nexists$	Non existing event
$\uparrow$	Event of sending data to
$\vdots$	Waiting event
$\xrightarrow{wait}$	Wait for receiving message
$\otimes$	Misuse behavior function definition
$\wedge$	and
$\oslash$	Adaptation event
$\neg$	Not

Symbol	Definition
$\xrightarrow{fixadapt}$	Fix adaptation of healing state
$\xrightarrow{adapt}$	Adaptation in healing state
$\supset$	Recognition of abnormal behavior event
$\vee$	or
$\oplus$	Damage definition event
$\xrightarrow{extract}$	Extract features of the damaged state
$\xrightarrow{FIX}$	Fix the damage function event
$\xrightarrow{HealSearc\ h}$	Search for suitable healing function state
$\leftrightarrow$	Match the damage and healing of the damaged function
$\xrightarrow{HealFix}$	Fix the heal function to specific damage state
$\bowtie$	Fix the healing event
$\xrightarrow{Test}$	Test the healed component state
$\neg$	Undamaged
$\boxplus$	Regeneration of new component event
$\xrightarrow{Regenerate}$	Regeneration of new component
$\mapsto$	Function definition
$\times$	Set of all ordered pairs between two sets
$\subseteq$	Subset has less elements or equal to the set
U	Union of two sets
$\geq$	Greater than
α	Iteration number
β	False negative percent
δ	False positive percent
$\mathbb{N}$	Natural number

ABBREVIATION

Abbreviation	Description
A	Alert of detection
ADA	Adaptive agent
AIS	Artificial immune system
ANA	Analysis agent
Anomalyhealmsg	Anomaly abnormal behavior heal message
AnomalyMsg	Anomaly abnormal behavior message
APC	Antigen Presenting Cell
C1	Check the steadiness of the system components
C2	Check the availability of the system
C_{ID}	Capability of Detection
C-k-NN	Cluster k-nearest neighbor
CTMC	Continuous Time Markov Chains
DA	Detection Algorithm
DC	Dendritic cell
DCA	Dendritic cell Algorithm
DetectionMsg	Detection of abnormal behavior message
DMG	Damage
DoS	Denial of services
DT	Danger Theory
FBI	Federal Bureau of Investigation
FN	False negative

Abbreviation	Description
FP	False positive
GMM	Gaussian Mixture
HC	Healing Candidate
HIS	Human immune system
HK	Heal Knowledge database
HSP	Heat shock proteins
I	Intrusion
IDS	Intrusion detection system
IMAAD	Immune multiagent active defense
IPS	Intrusion prevention system
k-NN	k-nearest neighbor
LAN	local-area network
MD	Misuse database
MHC	Histo-compatibility complex
MisusehealMsg	Misuse abnormal behavior heal message
NBA	Network Behavior Analysis
NN	Nearest- Neighbor
NPV	Negative predictive value
NS	Negative selection
NTP	Network Time Protocol
P	Probability of
PAMP	Pathogen-associated molecular proteins
PPC	Prevention Predictability trust Coefficient

Abbreviation	Description
PPV	Positive predictive value
Probe	Surveillance and other probing
R2L	Unauthorized access from a remote machine
RBF	radial basis function
RC	Repair Access
RecognitionMsg	Recognition of the abnormal behavior message
Rs	Repair time
SEA	Sense agent
SH	Self-healing system
SHA	Self-healing agent
SNMP	Simple Network Management Protocol
SOA	Critical Service-oriented architectures
TLR	Toll-like receptor
TN	True negative
TP	True positive
U2R	Unauthorized access to local super user (root) privileges
WBC	White blood cells
WBC	White blood cells

CHAPTER 1
INTRODUCTION

1.1 Thesis Overview

With the explosive growth of the Network Systems and Internet, and the rise of information technology in everyday life, the need for networks security has become critical. Meanwhile, the complexity of attacks is on the rise regardless of the beefed-up security measures. Critical Service Oriented Architectures (SOA) and, more specific, Web Critical Services designs are currently being widely used for the development of open, large-scale interoperable systems. In those systems, performance, security and trustability are challenging research issues. Intrusion Prevention Systems provide an in-line mechanism that focuses on identifying and blocking abnormal network activities in real time.

In network systems in particular those used for critical services, the main security danger comes from insider abuse and from external intrusions. Two broad approaches exist to tackle this problem: anomaly detection and misuse detection. An anomaly detection system is qualified only on examples of normal links, and thus has the potential to detect novel attacks. However, many anomaly detection systems simply report the anomalous activity, rather than analyzing it further in order to report higher-level information that is of more use to a security representative. On the other hand, misuse detection systems recognize known attack patterns. However, such systems cannot be decentralized, and are unable to detect novel attacks, or stop the spread of the damage caused by the different malicious activities i.e. prevention response.

Immune System presents valuable metaphor for computer security systems and it is an appealing mechanism because firstly, the Human Immune System (HIS) defends the body with high level of protection features from pathogens, in a Self–Organized,

Robust, Distributed and Diverse manner. Secondly, current security systems are not able to handle the dynamic and increasingly complex nature of the computer systems and their security needs. Based on this deficiency, Artificial Immune Systems (AISs) have been successfully applied to a number of network security problem domains that include Intrusion Detection Systems (IDS), Intrusion Prevention Systems (IPS) and Anti-malware Systems. However, most of the developed intrusion detection system technologies and prevention technologies work fairly well in static systems but have certain deficiencies in dynamic systems, such as the lack of self-adaptation, lack of robustness and they are mostly centralized in design. Hence, it is necessary to construct an effective defense system, which has features of autonomous, self-adaptability, self-detecting, self-monitoring, and self-healing. Currently the trend is towards large and much more dynamically configured systems.

This research presents a new security system for critical services network based on the combination of biological intrusion prevention (IP) and Self-Healing (SH) concepts. This system integrates an artificial immune intrusion prevention system for network security inspired by the immunology theory known as the danger theory (DT), adaptive immune system and pattern recognition classifications. The proposed system is inspired by the Human Immune System (HIS), which is applied to the autonomous defense system. The inspired IPS mechanisms look at the danger model and its application to activate malicious behavior defense in order to create a fully decentralized model. The developed Intrusion Prevention System (IPS) analyzes the behavior of network system processes and critical data in network traffic to detect harmful events. The detection of damages caused by different types of abnormal events or attack features is used to trigger the Self-Healing (SH) mechanism. This system is to be autonomous and enhances the fault repair and the system recovery.

The main features of the biological immune system adopted by this new IPS are the features of two interacting subsystems: the innate and adaptive immune systems. While the reality of an innate immune system has long been obvious, it has little impact on the design of AISs [1]-[4]. AISs, to date, have largely been inspired by adaptive immune system. Scientists described the adaptive immune system as a system capable of specific recognition and remembrance of pathogen, while the innate immune system is characterized mainly as the first line of defense and rapid-

response system against pathogens. Based on the understanding of how both of these systems function, this research mapped a number of features to IPS design; at the same time the Intrusion Prevention System is enhanced by Self-Healing mechanism. These features provide a general structural framework in designing a new biological IPS generation, in general. These features are: Autonomy, Self-Repair, Distributed, Self-Organized, Lightweight, Multilayered, Adaptability, Diverse, and Disposable [5]. The definition of these design features is presented in Appendix A.1. The combination of these design features form a robust IPS specification and design with optimal performance.

In 1994, Negative Selection (NS) algorithm was the first AIS algorithm for intrusion detection introduced by Forrest et al. [6] that inspired from adaptive immune system. It is a loose abstract model of biological negative selection that concentrates on the generation of change detectors. Many researchers in [7]-[9] evaluated and improved the NS in addressing the problem of network security system, specifically concerning intrusion, and detection and prevention of the network system. Despite this successful abstraction of negative selection, further analysis showed that there were other problems such as scalability, coverage problem and high false positive error in detection. The main reason for this drawback is specific pattern matching rules using linear algorithm.

Recent new AIS algorithms for intrusion detection and prevention system were based upon the abstraction of Matzinger's danger theory Matzinger [10] introduced in 2002 [11]. Many researchers later followed this work [12]-[16]. The danger theory concept for AIS is a fast growing alternative, in addition to negative detection. Even though these algorithms have been quite successful at reducing the false positive error, the error remains high and need to be reduced further. Firstly, one of the disadvantages of these algorithms is their usage of linear detection rules for classifications [17], because as an assemblage of linear classifiers, there are severe restrictions on the datasets that the algorithm will be able to evaluate. This is made inferior by the fact that the gradients of the boundaries are constant, applying still further restrictions against the regions in signal space that the algorithm can differentiate between normal and abnormal behavior. Secondly, these algorithms have deficiencies of low predictability of prevention responses and inline response

activities. An intrusion detection and prevention system for critical services networks needs fast and accurate response against intelligent and abnormal behaviors. An autonomous intelligent IPS system can be achieved by integrating AIS, pattern recognition and machine learning.

Based on this integration, an effective biological IPS is developed to combine with self-healing system using autonomous multilayered multiagent system. To achieve optimal design features suitable for the environment of critical network systems, the system performance is adjusted through autonomous computing tool which is multiagent paradigm. The achievement of these design features is the main goal in constructing the system modules.

1.2. Motivation

Realizing the potential of intrusion and prevention systems as in-line security system, many techniques and models have been developed by many researchers. However, the network systems continue to be challenged by different types of abnormal activities that are spreading rapidly and causing damages in the system components. These factors lead to the necessity of developing new system for autonomous intrusion prevention and self-healing system that applies central authority that generates the defense mechanisms and deploys these to the systems in the field. While this strategy works fairly well in static systems, but it faces problems in dynamic and diverse network systems.

The motivation for this work is to investigate:

- How the functionalities of the human immune system can be used as an inspiration to develop a network security system. Human immune system has the capability to detect and stop anomaly and misuse of abnormal behavior with different integrated mechanisms.

- How the integration between AIS, pattern recognition and machine learning can introduce a new concept of autonomous and high accurate IPS and HIS system.

- How biologically inspired techniques coupled with intelligent agents paradigm technology can be used efficiently to design future generations of intrusion prevention and Self-Healing systems to achieve highly secured computer network systems.

1.3 Research Problem

In spite of the vast choices of network security systems, many existing network systems are still susceptible to intelligent, dynamic, and successful abnormal activities. To face this problem of vulnerabilities in critical network systems, a sensible tendency is towards more intelligent, autonomous, adaptable, dynamically configurable systems, and fault healing network security systems [18],[19]. Meanwhile, the future is likely to belong to ubiquitous systems where the number of devices and their diversity exceed the capacity to centrally administer them. Furthermore, ubiquitous systems will also include many devices that are not connected continuously or they shall dynamically change their status as demonstrated in [18]. Obviously, there is an urgent need for a network security system to address these issues. This research looks at a model of computer immune system and its application to intrusion prevention system combined with self-healing system, which shall create an autonomous system using autonomic computer techniques and tool [19].

1.4 Research Questions

In this thesis, we attempt to provide answers to these questions:

1. Are the mechanisms of Human Immune System a good metaphor for intrusion prevention system as a network security system?

2. Is it possible to detect, prevent, and heal the computer intrusions using the artificial immune system based on danger theory and adaptive immune system with minimum false alerts?

3. Is intelligent multiagent paradigm a suitable tool to build autonomous intrusion prevention and self-healing system?

1.5 Research Objectives

The main objective of this thesis is to develop a security system for critical network services by building an intrusion prevention system inspired by human immune system features, and one that is capable of implementing a healing system to recover the damages caused by abnormal behavior using self-healing model. The work has been divided into specific areas that aim to accomplish the following goals:

1. To develop an autonomous mechanism for intrusion prevention system that is effective for anomaly detection and prevention, based on artificial immune system.

2. To design a network security system that combines the intrusion prevention system with self-healing mechanism.

3. To simulate the model for efficiency and robustness, and compare and contrast it with existing security models.

1.6 Research Scope

This research deliberates on enhancing the performance of intrusion prevention system especially in reducing the false alerts: false positive and false negative errors. False positive detects normal behavior as abnormal, while false negative detects abnormal behavior as normal. These errors in detection cause huge damages and losses, particularly in network system for critical services such as Banking network systems, E-commerce and Internet Service providers, Health Care systems, Military network system, Security systems and Governmental Organizations. The scope of this research is to develop an integrated solution to improve the performance and security of network systems for critical services based on the use of an autonomic framework exploited to auto-configure and to auto-tune the system, guaranteeing high performance and fulfillment of any given security levels [20],[21].

1.7 Research Contributions

1. A system that responds effectively to new abnormal activities without human intervention which would significantly improve the security and optimize the performance of network systems used by critical services.

2. A robust multilayered security system that reduces false alerts and errors in detecting and preventing abnormal activities.

3. New non-classification algorithms for hybrid intrusion prevention system with capabilities to detect and prevent anomaly, and misuse of abnormal activities. The classifications algorithms use the features of k-N-N means cluster and Gaussian mixtures. These algorithms have been proven as:

- Having high detection accuracy and prevention predictability.
- Fast in training and testing.
- Requiring small training data.

4. Network systems with enhanced survivalability, which results from the combination of intrusion prevention system and self-healing mechanism.

5. The introduction of a specification language for IPS and SH system.

6. A new Conceptual Framework for AIS development.

7. The introduction of new metrics for prevention predictability trust, detection capability and self-healing.

1.8 Research Methodology and Activities

The main research objectives are realizable by taking it into the following research methodology and activities:

1. Analysis of the biological IPS and self-healing system according to the following activities:

- Analyze the components, properties and features of the human immune system architectures, and identify the mechanisms that we wish to duplicate in order to apply the hallmark features of the immune system. Find and define the different types of defenselessness and abnormal behaviors.

- Study the immune intrusion detection system, intrusion prevention system, self-healing system and autonomic computer system using multi-agent system.

- Study the immune intrusion detection system, intrusion prevention system, self-healing system and intelligent agent applications in autonomous network security.

- Analyze the Intrusion Prevention System (IPS) for network security. Find similarities and differences between IPS and human immune system.

- Construct an analytical model for developing AIS system using the conceptual frame work for AIS and layered framework for engineering AIS

- Construct abstract models of the IPS and SH using Multi agent system paradigm.

2. Create specification language for the biologically inspired IPS and SH systems:

- Using the set theory and Z-notation to specify the roles, functions and responsibilities of each agent.

3. Create and verify the theoretical design:

- Build a theoretical flow of the agent design and verify it using Petri nets.

- Verify the logical and computational models.

4. Create new mathematical and computational models:

    - Build the mathematical and computational model.
    - Construct the algorithms for the detection, prevention, and healing systems as inspired by human immune system mechanism.

5. Simulate the proposed system using Matlab software:

    - Simulate the system using different scenarios and diverse standard dataset.
    - Evaluate the model reliability using traditional and new metrics. Compare and contrast it with the contemporary models.

The flow of research activities are presented in Figure 1.1.

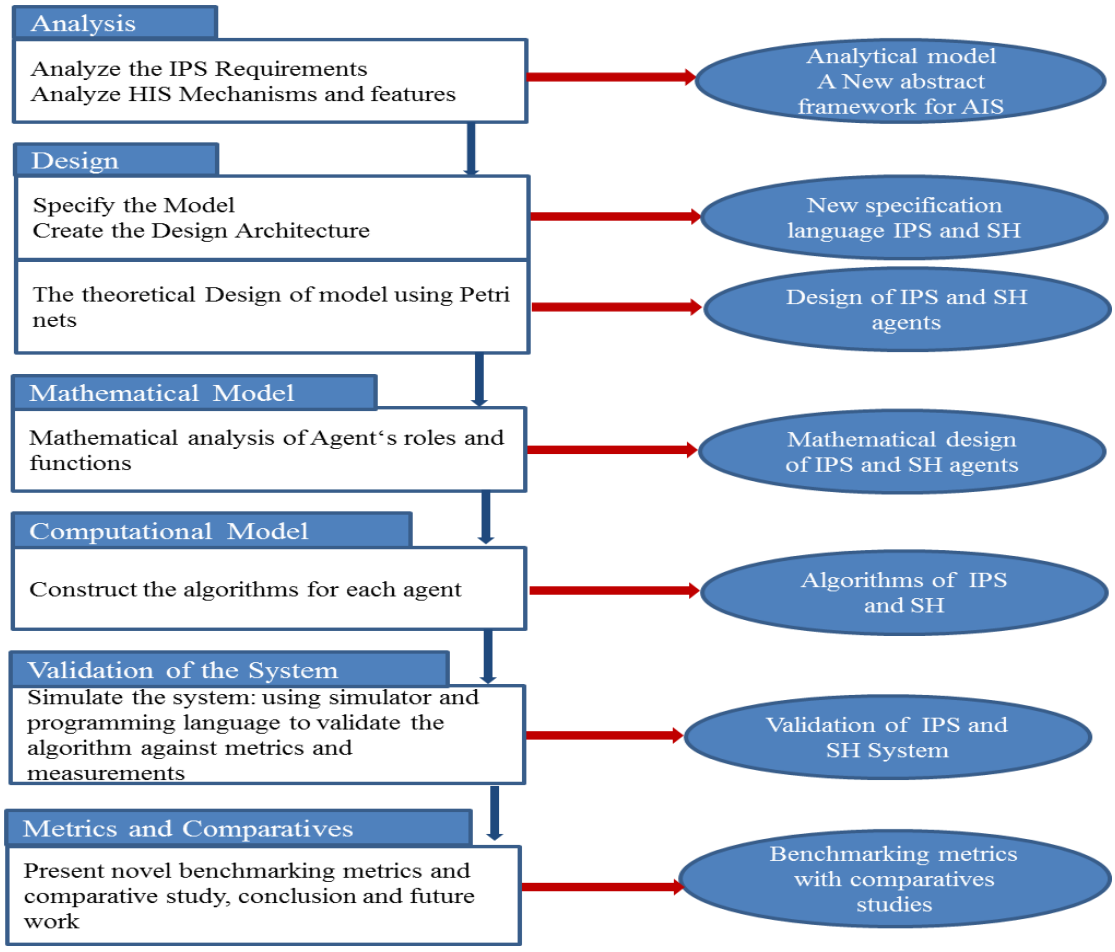


Figure 1.1: The Flow of the Research Activities

1.9 Structure of Thesis

This thesis is structured in chapters. The first chapter introduces the whole research in addition to a brief background on all the concepts involved in this work, motivation of the new approach, the problem statement, research question, objectives, and scope of research, contributions, research methodology, and relevant research activities. Chapter two provides an overview of critical services network, intrusion prevention system, human immune system, autonomic computing, self-healing system and covered many Related works. An analysis of the proposed system is introduced in chapter three. The IPS design specification, logic computational of the system is discussed in chapter four. Chapter five describes the mathematical model and system algorithm. Chapter six covers discussions on the evaluation and measurement of system, and the experiments and dataset used to verify the model. In chapter seven, new metrics for prevention predictability and self-healing system are introduced, comparison studies are carried out, and the results are discussed. Finally, conclusion and recommendations for future work are included in chapter eight.

1.10 Chapter Summary

This chapter presents the research problems, objectives, motivation, scope, methodologies and activities. The main problem in network security system for critical services is how to challenge the intelligent abnormal behavior activities. The aim of this research to develop an abstract system inspired from human immune system, integrated with algorithms for autonomous intrusion prevention system to reduce false errors in detection and prevention response. A self-healing mechanism is integrated in the system to heal any damages caused by successful intrusion.

CHAPTER 2

BACKGROUND STUDY

2.1 Chapter Overview

As a network security system developer, building new technologies for intrusion detection prevention system to ensure the security of critical services, it is essential to have an obvious understanding of why we need network security systems; what are network for critical services; security needs of critical services; IPS requirements, features, methods and techniques. Since we are developing a new IPS inspired from human immune system combined with a healing mechanism, it is also necessary to have a clear view of the human immune system from a biological perspective. Moreover, we need to discover the nature and mechanisms of the immune system. The area of immunology mechanisms is very interesting and rich for research, inspiration and modeling computer security systems. The entirety and wealth of these mechanisms, put together, will be extremely resourceful for network security systems developers.

The aims of this chapter are, first to introduce and review the current technologies of intrusion prevention system (IPS), roles, capabilities and high performance requirements for the system. Secondly, the structure, roles, and functions of human immune system are described. Thirdly, some artificial immune systems that have been developed are explored and discussed, followed by definitions and mechanisms of self-healing system. Finally, some related works are explored and discussed.

2.2 The Needs of Network Security Systems

Although network design, development and usage are critical to computing, they have become one of the main infrastructures in any organizations, and are being used

intensively in our daily activities. Recently, these systems have been subjected to diverse criminal activities. In 2009, the Federal Bureau of Investigations (FBI) reported that for the first time ever, income from cybercrime had exceeded drug trafficking as the most profitable illegal global business, evaluated at taking in more than $1 billion annually in profits. Separate hackers and groups insecurely tie themselves together into an organized criminal hierarchy where common goals are achieved through a reward system. In the first half of 2009, IBM reported that, abnormal behavior links on websites increased by 508%. Much of the malware spreading is accomplished by organized cybercrime networks. Meanwhile abnormal behaviors by insiders were listed as the top threat for 2009. With the downturn in the economy, it was no surprise that many desperate and discontented employees attempted to exploit the companies and organizations they currently or previously worked for. Here are just a few of the 2009 stories mentioned in [22]:

- An employee in Bank of New York Mellon was on identity theft charges. He was charged with grand larceny, identity theft, and money laundering after stealing and using New York Mellon employee information. He opened phony bank and brokerage accounts where he deposited stolen money.

- Another employee accessed a system a year after he was no longer an employee at United Way. He deleted files and deactivated the voicemail system.

- An employee in T-Mobilean Company stole customer data and sold them to a data broker who one at a time sold the data to T-Mobile competitors. It was comprised of millions of records that contained treasured information such as account expiration date, so competitors could target those customers at the time they may look for a new provider.

- After sequences of arguments with executives and shareholders, the previously YouSendIt co-founder and CEO left the company and later implemented a denial-of-service attack against YouSendIt systems.

One can notice from these very few examples the urgent requirements and needs to develop efficient and reliable security systems for network especially those used by critical services.

2.3 Network for Critical Services

Critical services networks are complex physical and interconnected-based systems that form the lifeline of modern global society, and their trustworthy and secure operation is of paramount importance to national security and economic strength. Many researchers have identified telecommunications, electric power systems, natural gas and oil, banking and finance, transportation, water supply systems, government services, and emergency services, as the critical services. The networks of these services providers transfer critical information that are highly interdependent among themselves, and hence, a disruption in one network or a part of a network will have cascading effects on other parts of the network. The disruption could be due to synthetic abnormal behavior events, such as physical destructions or intrusions into network systems. Identifying, understanding, and analyzing such interdependencies among network systems pose significant challenges. These challenges are greatly magnified by the expanse of the global network and complexity of individual network, and the nature of coupling among them. Figure 2.1 shows an abstract view of the critical services networks. In the figure, one can notice that much critical information can be transferred between the end users with the use of diverse type of communication devices and technologies connected with the network systems. Internet and global network systems are highly dynamic and varieties of communication networks are interconnected to provide different critical services. These services need security, monitoring, and control.

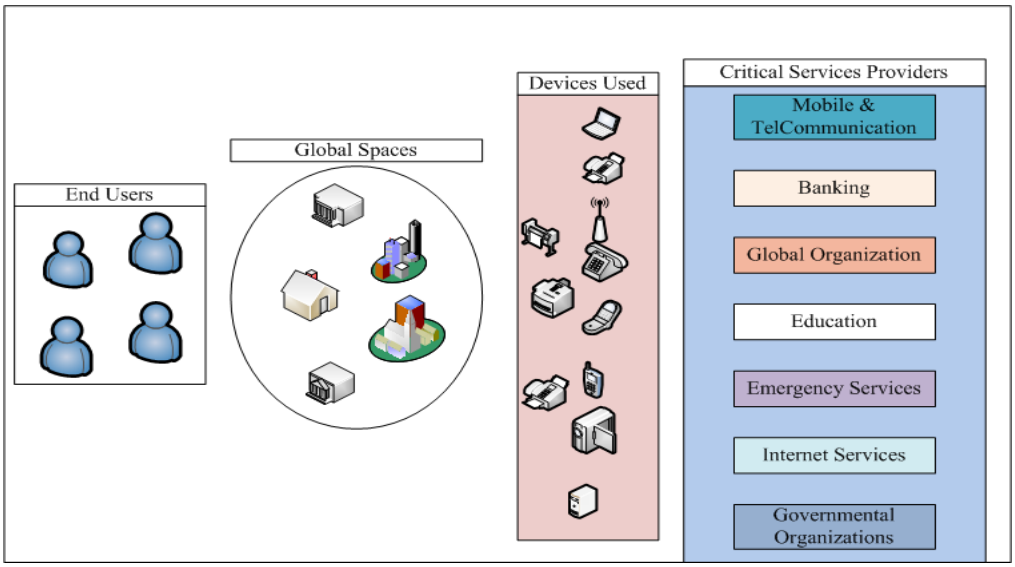


Figure 2.1: Abstract View of Critical Services Network System

These communication networks are closely associated with the supervisory control and data acquisition systems in the network. One of the primary concerns has been the issue of large-scale fault events, and their impact on the overall performance and stability of network systems [23]. Various incidents of abnormal activities in the recent past have indicated the extent to which the network system are vulnerable and the urgent need to protect them against intelligent intrusions and faults. One of these appealing security systems is intrusion prevention system. IPS can be resided as the first defending system in the network. The performance of such system must be optimized to ensure high classification between normal and abnormal activities. In critical services networks, IPS must function as the main and significant defense system to provide secure services. Figure 2.2 explains how an IPS can be resided to accomplish highly secured network system for critical services.

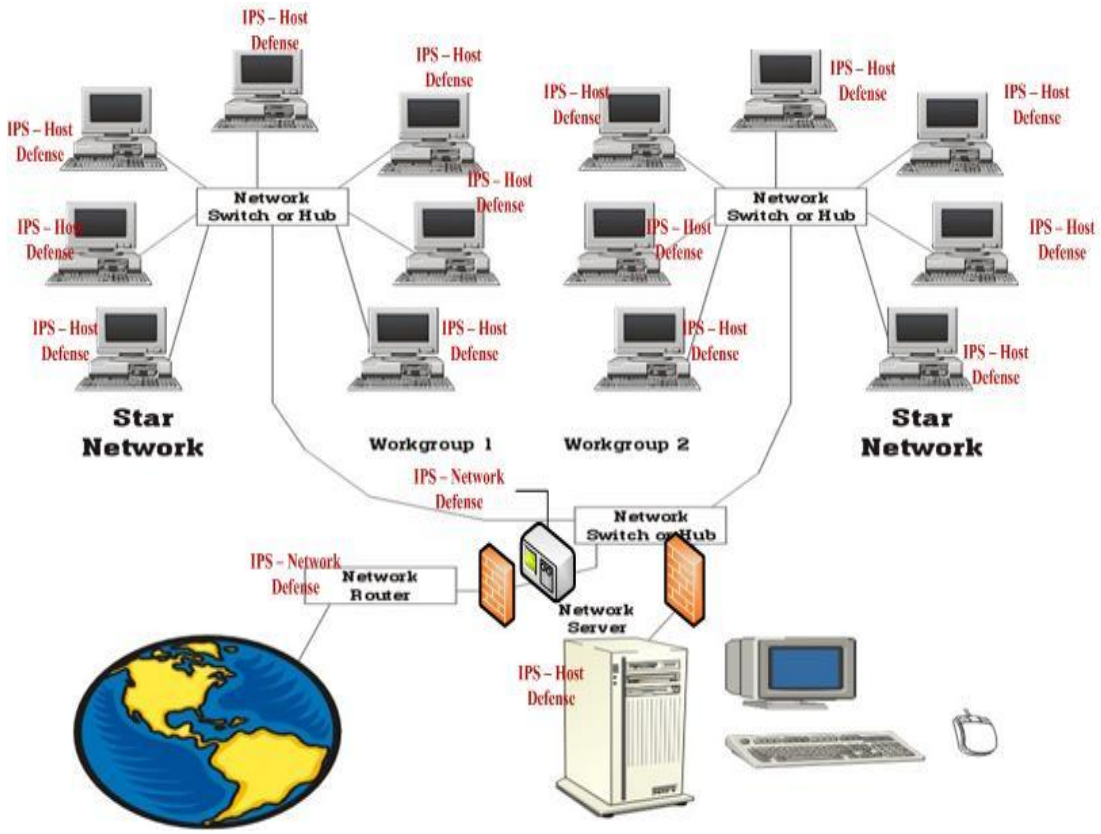


Figure 2.2: The Distribution of IPS in Network System

2.4. Intrusion Detection and Prevention Systems

Intrusion prevention systems (IPSs) were invented to resolve ambiguities in passive network monitoring by placing detection and prevention systems in-line. A considerable integration and improvement in intrusion detection, deep data packet inspection-based on stateful firewall and prevention technologies, would allow IP systems to make access control decisions based on applications necessity rather than internet protocol (IP) address or ports as traditional firewalls have been doing. Meanwhile, IPS systems were originally a factual conservatory of intrusion detection systems (IDS). IDS is a software system designed to identify the threats to computer networks and systems and monitors in-line, but no response is taken as in IPS where IPS is able to respond defensively to prevent the attackers from successfully creating harmful and abnormal activities. An IPS serves secondarily at the host level to prevent potentially malicious activity. It monitors network and system activities for malicious or abnormal behavior, and reacts in real-time to block or prevent those activities. For high performance and effectiveness, an IPS must provide a full array of safeguard against discovered and undiscovered attacks, offer high accuracy by enabling a low rate of false positive and false negative errors of detection and has high-speed response for prevention (When an IDS and IPS incorrectly identify benign activity as being malicious, a *false* positive would have occurred. When an IDS and IPS fail to identify malicious activity, a *false* negative would have occurred [24]). Moreover, professional IPS must introduce minimal degradation of network performance and have low latency rates [25]. IPS is a security system that preserves network integrity, availability and confidentiality from diverse types of intrusions. Table 2.1 details the types of intrusions, their target and defenselessness that can be detected and prevented.

Table 2.1: Network Defenselessness Controlled By IDS and IPS

Intention	Defenselessness
Programming flaws	Addressing error, buffer overflow, malicious type code, malicious code: (virus, worm, Trojan horse), and parameter modification code.
Confidentiality	Cookie, traffic flow analysis.
Precursors to attack	Port scan, reconnaissance, operating system application and finger printing.
Availability	DNS attack, protocol flow and connection flooding.
Integrity	Protocol flow, website defacement, protocol flow and DNS attack.

Many aspects have to be considered when developing IPS system such as: which IPS methodology must be followed, the technologies that can be used to develop IPS, and the capabilities required or a particular IPS being developed. These aspects of IPS are presented as in Figure 2.3. In the next section, detailed explanations of these criteria and aspects are given.

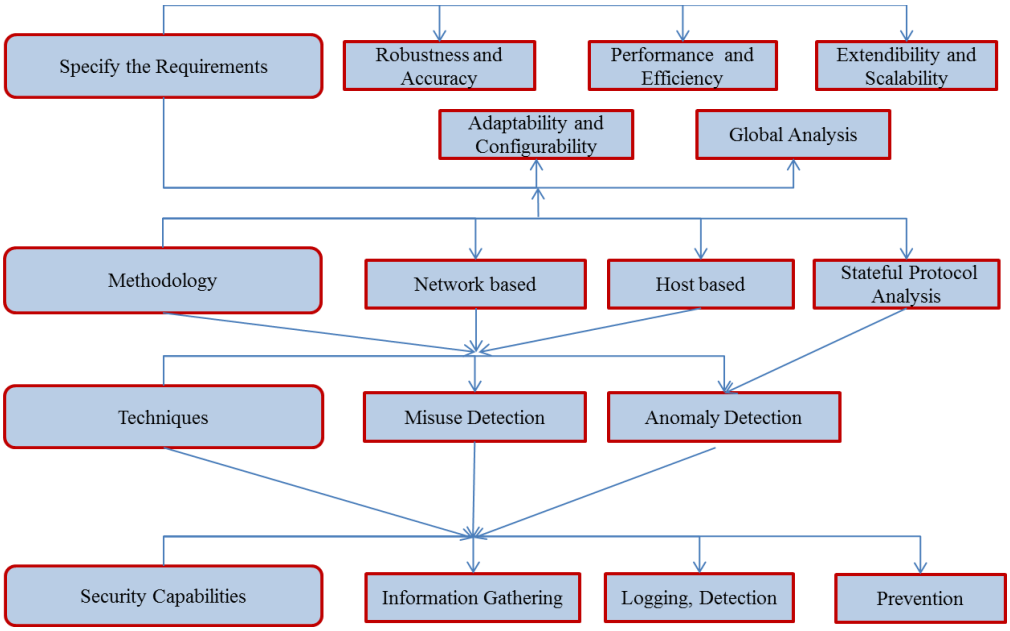


Figure 2.3: Aspects of IPS Systems

2.4.1 IPS Detection Methodologies

Throughout the years, network security system developers have introduced IP System using three common detection methodologies: misuse detection, novel or anomaly detection and stateful protocol analysis [26], [27]. In the next sections, we will discuss the various aspects of these methodologies.

2.4.1.1 Misuse Detection Methodology

The main feature of misuse detection methodology is to examine network data packet sequences for known abnormal activities. Misuse detection is the process of comparing signatures against observed events to identify possible occurrences, typically through some form of pattern-matching algorithms. Misuse detection is very effective at detecting known malicious activities with quite low false positive error rate, but largely ineffective at detecting previously unknown malicious activities [28],[29].

2.4.1.2 Anomaly Detection Methodology

Anomaly detection methodology bases its decisions on a profile of normal network or system behavior. Any event that does not conform to this profile is considered anomalous or novel [21], [26], [30], [31]. The major benefit of anomaly-based detection methods is that they can be very effective at detecting previously unknown malicious activities. An IPS using anomaly-based detection has profiles that represent the normal behavior of such things as users, hosts, network connections, or applications. The profiles are built up by monitoring the characteristics of typical activity over a period. The IPS then uses statistical methods to compare the characteristics of current activity to thresholds related to the profile, such as detecting when a web activity is consuming a significantly more bandwidth than expected and alerting an administrator of the anomaly.

2.4.1.3 Stateful Protocol Analysis Methodology

The method of comparing prearranged outlines of normally accepted definitions of considerate protocol action for each protocol condition against observed events to identify deviations is known as Stateful protocol analysis methodology. Stateful protocol analysis relies on common outlines that specify how a particular protocol must and must not work, which is dissimilar to anomaly-based detection that uses host or network-specific profiles. Stateful protocol analysis method has some major drawbacks such that it is very resource-intensive due to the complexity the analysis and overhead involved in performing state tracking for many simultaneous sessions. Another serious drawback is that it cannot detect attacks that do not violate the characteristics of generally acceptable protocol behavior [32], [33].

To capture the functionality of new IPS, the methodologies required for high secure network system must be specified. This need to know, which information has to be monitor and what the major objectives of the new develop IPS system.

2.4.2 IP System Technologies

A review of existing IPS detection methodologies has shown that in order to capture the functionality of new IPS, the methodologies required for high security network system must be specified. This requires the knowledge of which information that needs monitoring and what are the major objectives to develop new IPS. This section will discuss the technologies used by IPS.

With respect to residency, IPS technologies are usually divided into host-based, Network Behavior Analysis (NBA) and network-based technologies which are differentiated primarily by the types of events that they can recognize and the methodologies that they use to identify possible incidents. Host-based systems reside in each host that requires observation, and collect data of suspicious activity that might affect the operation of the host system. Network Behavior Analysis examines network traffic to identify threats that generate unusual traffic flows, such as DoS attacks, scanning, and certain forms of malware. In contrast, network-based IPSs

observe the traffic on the network holding the hosts to be defended. Any of the three techniques has its own advantages and disadvantages.

Host-based systems can detect local attacks, benefit appreciation attacks and attacks which are encrypted, and are capable to decide if an attempted attack has been definitely successful. However, such systems can be rather complicated to deploy and administer, especially when the number of hosts requiring protection is huge. In addition, host-based systems are unable to detect attacks against several targets of the network. While the Network Behavior Analysis, which is usually used for detecting special types of threats is unable to cover a wide range of malicious activities. On the other hand, network-based systems, even though they are capable of monitoring a large number of hosts with relatively little deployment outlay and are able to identify attacks to and from multiple hosts. However, are largely incapable to detect whether an attempted attack has actually been successful, and are unable to deal with local or encrypted attacks.

To overcome the shortcomings of those systems, the hybrid systems that include and integrate host and network-based elements can offer the best protective capabilities. Critical services network systems need such hybrid protection capabilities against attacks from multiple sources [20], [27]. The hybrid IPS can be resided to monitor both network traffic and each individual host, with different types of data monitored to guarantee a multilevel security. Generally, in order to develop an IP system that follow one of the methodologies and techniques, developers have to specify the major roles, the main security capabilities, the response techniques and the requirements of high performance IP system; the IPS development specification is as shown in Figure 2.4. The main roles, security capabilities, and IP system requirements are further elaborated in the following subsections.

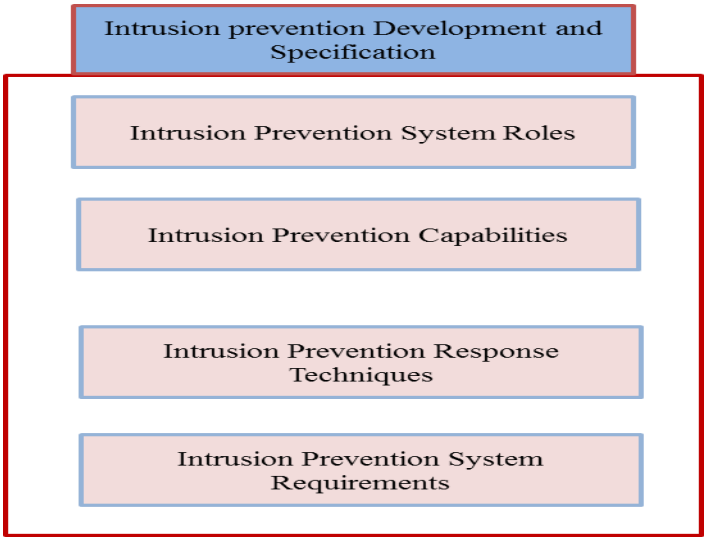


Figure 2.4: Intrusion Prevention System Development Specification

2.4.2.1 Major Roles of IPS Technologies

A variety of IPS technologies, in general, performs many major roles in addition to observing and examining events to identify undesirable network activity. These main roles are: recording information related to observed events, notifying security administrators of important observed events and producing reports.

The first role is registration of information, where the information is usually registered locally, and might be sent to separate systems such as security information management solutions, and enterprise management systems. The second role is notification, known as an alert. It occurs through any of the several available methods, including the following: e-mails, pages, messages on the IDS and IPS user interface, Simple Network Management Protocol (SNMP) traps, syslog messages, and user-defined programs and scripts. A warning message characteristically includes only basic information regarding an event, such that the administrators need to access the IPS for additional information of the event. Lastly, the third role is to provide reports that summarize the monitored events or provide details on particular events of interest. Some IDS and IPS are dynamic systems that are able to change their security profile when a new danger is detected. For example, an IPS might be able to collect more detailed information for a particular session after abnormal activity has been detected within that session. An IPS might also alter the settings when certain alerts

are triggered or what priority should be assigned to subsequent alerts after a particular threat is detected [27], [33]-[36].

Currently, system developers are looking at the implementation of autonomic computing concept to minimize interventions by system administrators. In the situation of network security system, the time and spread of abnormal behavior is critical and any delay in response to abnormal activity may cause huge damages. The preference is to have a quick response when abnormal behavior alert is triggered. To improve network reliability and ensure system's continuity, it is highly desirable that the next generation of IPS technologies is capable to deal with abnormal events autonomously.

2.4.2.2 Security Capabilities of IPS Technologies

Most IDS and IPS technologies can provide a wide variety of security capabilities that can be divided into four categories: information gathering, logging, detection, and prevention [35], [37]. IPS technologies offer information-gathering capability, such as collecting information on hosts or networks by observing the network traffic activities. Examples include identification of hosts and operating systems including the applications that they use, and general characteristics identification of the network.

Logging is an IPS capability which performs typically extensive logging of data related to detected events. The data can be used to confirm the validity of alerts, investigate incidents, and correlate events between the IPS and other logging sources. Data fields commonly used by IPSs include event date and time, event type, importance rating, and preventive action that have been performed. Specific types of IPSs log of additional data fields are such as network-based IPSs performing packet capture, and host-based IDS and IPS recording user IDs. Generally, logs should be stored both locally and centrally to support the integrity and availability of the data. In addition, IDS and IPS should have their clocks synchronized using the Network Time Protocol (NTP) or through frequent manual adjustments so that their log entries have accurate timestamps.

IPSs offer extensive and broad detection capabilities. Most systems use a combination of detection techniques, which generally support detection that is more accurate and more flexible in tuning and customization. The types of events detected and the typical accuracy of detection vary greatly, depending on the type of IDS and IPS technology. Finally, and the most important capability of an IPS is prevention capability. Existing IPSs offer a variety of prevention capabilities and the specific capabilities vary according to the type of IPS technology used.

Administrators of IPSs, generally, are allowed to identify the preventive measures recommended by the IPS for each type of alert. This includes enabling or disabling prevention, as well as specifying which type of preventive action should be used [37]. IPS sensors have learning or simulation mode that suppresses all preventive actions, and instead indicates when prevention should be performed. This allows system administrators to monitor and fine-tune configuration of the prevention before enabling the preventive actions, which reduces the risk of inadvertently blocking benign activity. By using inspired IPS and training Agent based system, the role of administrator is reduced which saves time and increases autonomous reaction rate.

IPSs require at least some tuning and customization to improve their detection accuracy, usability, and effectiveness, such as setting the appropriate preventive actions to be performed for a particular alert. Technologies vary widely in their tuning and customization capabilities. Typically, a more accurate IP system detection can be developed from the default configuration, adjusted by a powerful IP system product tuning and customization capabilities [38]. In a traditional IPS, code viewing and editing are the needed customization features, where some IDS and IPS technologies permit administrators to see some or all of the detection-related code. This is usually limited to signatures, but some technologies allow administrators to see additional codes, such as programs used to perform stateful protocol analysis. Viewing the code can help analysts to determine why a particular alert is generated; and is helpful for validating alerts and identifying false positives. The ability to edit all detection-related code and write new code (e.g. new signatures) is necessary to fully customize certain types of detection capabilities. The performance of these systems can be optimized and improved with the use of autonomous concept.

2.4.2.3 IPS Response Techniques

There are several response techniques in the IP systems, which can be divided into the following classes [25], [27], [37], [39]:

- IPSs that stop the attack itself. This may be achieved by terminating the network connection or user session that is being used for the attack.

- IPSs that block access to the intended target. The system will block all access to the targeted host, service, application, or other resources from the offending user account, IP address, or other attacker's attribute,

- IPSs response that change the security environment. The IPS could change the configuration of other security controls to disrupt an attack. Firewall, router and switch are the common examples of such IPSs where a network device is reconfigured to block access from the intruder or to the target, and a host-based firewall on a target is altered to block incoming attacks.

- IPSs that change the content of the attacker information. In this technique, an IPS is able to change the content of the attack where the IPS can either remove or replace the malicious portions of an attack to make it benign. A simple example is an IPS that acts as a proxy and normalizes incoming requests, which means that the proxy repackages the payloads of the requests and discards header information. This might cause certain attacks to be discarded as part of the normalization process.

2.4.2.4 IDS and IPS Requirement Features

To develop improved, effective and high performance IDS and IPS, many researchers have identified specific requirements, which are of particular interest because they could be satisfied by mechanisms inspired by features of the human immune system [3], [5], [40], [41]. The main seven requirements are as follows:

- Robustness: It should have multiple detection and prevention points with low operational failure rates and are resilient to attacks.

- Accuracy: It should be able to detect and classify the attacks accurately since these are the essential requirements of an IPS.

- Extendibility: It should be easy and simple to extend the scope of IPS monitoring for new hosts regardless of the operating systems.

- Scalability: It is essential to accomplish reliable scalability to gather and analyze the high volume of audit data correctly from distributed hosts.

- Adaptability: It should adjust over time in order to detect dynamically changing network intrusions.

- Global analysis: In order to detect network intrusions, it should collectively monitor multiple events generated on various hosts to integrate sufficient evidence and to identify for any correlation between the multiple events.

- Efficiency: It should be simple and lightweight enough to impose a low overhead on the monitored host systems and network.

- Configurability: It should be able to configure itself easily to the local requirements of each host or each network component. Individual hosts in a network setting are various.

- Performance: It should incorporate higher ability and performance features.

Using new approaches inspired from nature, particularly the human immune system, it is possible to accomplish an IPS having these features. All of the features mentioned above form part of HIS features and more significantly HIS autonomously applies the security capabilities to protect the human body upon sensing an intrusion. In the following sections, we present an overview of HIS mechanism, immune system applications to IPSs, HIS models inspired to IDS and artificial immune systems of IDS and IPS.

2.5 Overview of Human Immune System

The human immune system (HIS) is a multifaceted network of organs and cells that is an amazing constellation, responsible for protecting the human body against extraterrestrial particles. HIS is based on two main integrated and interrelated mechanisms: Innate Immune system that is the first line of defense, and Adaptive Immune system; both systems have varying number of cells of intelligent behavior. However, HIS has a multi-layered architecture, with protection at many levels as described in Figure 2.5 [42], [43]. The most elementary layer is the skin, which is the first barrier to infection. The second barrier is physiological, where conditions such as PH and temperature provide inappropriate living environment for foreign organisms. Once pathogens entered the body, they are dealt with by the innate immune system and by the acquired immune response system, which is the adaptive immune system. The HIS is capable of detecting and eliminating pathogens, i.e. nonself elements, as quickly as possible. Moreover, HIS can protect against destructive and earlier hidden intruder cells i.e. pathogen active particles. The human immune system mechanisms are described by immunologist as one of a multilevel dynamic system of cells, organs and circulatory systems [44].

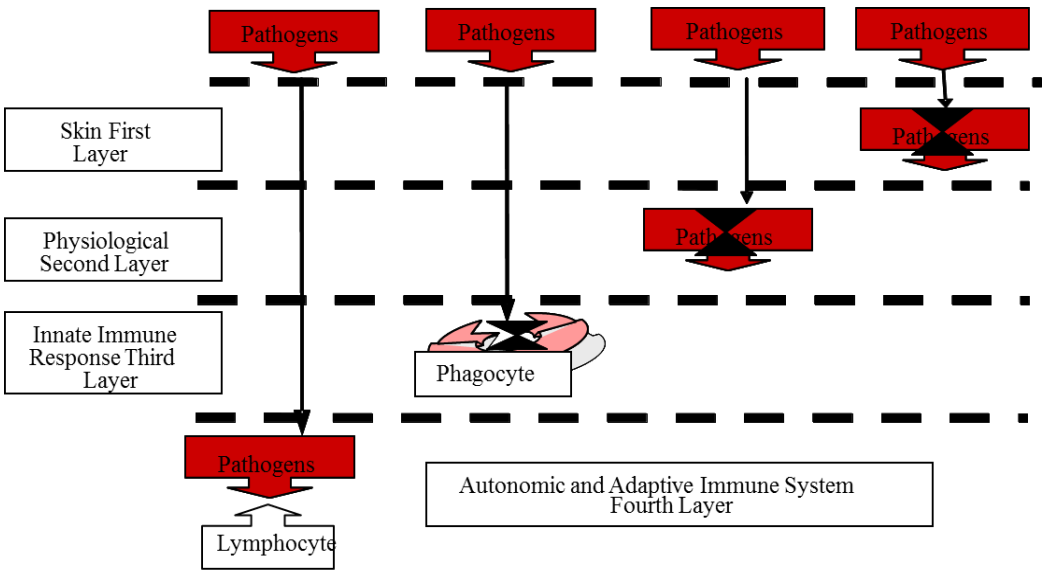


Figure 2.5: Multilayered Human Immune System

These creative and diverse mechanisms are suitably valuable for building a security system that will be able to protect networks from abnormal activities and intelligent

intruders in a similar way. HIS provides the basis for a representation of intrusion prevention system consisting of autonomous agents. Figure 2.6 shows a description of the HIS multilayered systems of organs, cells, and mechanisms. The next section will describe in details the adaptive and innate immune system cells, molecules, tissue organ and circulatory system.

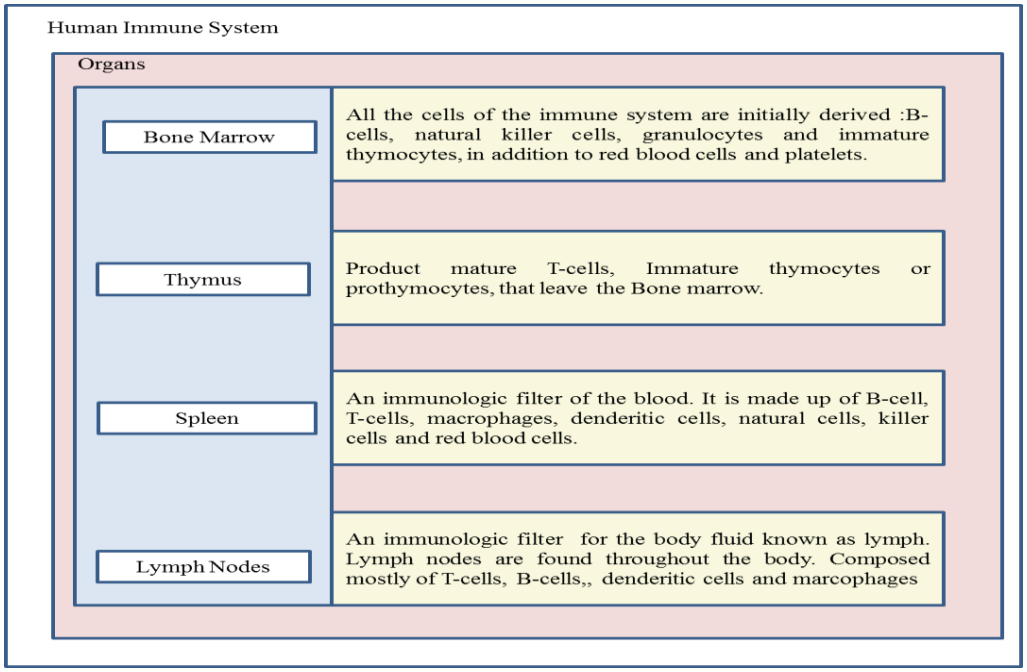


Figure 2.6: Human Immune System Organs and Cells

2.5.1 Adaptive Immune System

The adaptive immune system is known as a formulation of highly specialized, systemic components and direction processes that removes or stops pathogenic tackles. The adaptive or as named by many scientist "specific" immune system is stimulated by the “NonSpecific” and evolutionarily earlier innate immune system. The ability to recognize and remember specific pathogens that generate immunity are the main responsibility of the adaptive immune response, including to build up stronger attack defense each time a pathogen is encountered. The recognition of pathogen for creating pathogen-specific-memory is the main uniqueness of the adaptive immune response.

Another role of the adaptive immune system is the recognition of particular nonself antigens in the presence of self, during the process of antigen presentation. The productions of responses are adapted to potentially eliminate specific pathogen infected cells and to develop immunological memory, in which each pathogen is "remembered" by a signature antibody. These memory cells can be called to quickly eliminate a pathogen should subsequent infections occur [44]-[46]. The adaptive immune cells are produced in organs referred to as lymphoid organs because they are concerned with the growth, development, and deployment of lymphocytes [44].

B-cells that maturated in bone marrow and T-cells that maturated in thymus are the two major classes of lymphocytes. Therefore, they work according to a principle that allows lymphocytes to learn and adapt themselves to specific foreign protein structures, and to ''remember'' these structures when necessary. These principles are implemented by B-cells. Furthermore, to ensure that at least some of the lymphocytes will be able to react to the pathogenic elements, the human body relies on dynamic protection via a continual renewal of diverse lymphocyte receptors [45]. The mechanisms of B-cell and T-cell work in parallel and decentralized control, and both circulate in the human body and react with each other, i.e. self-monitoring with high adaptive, dynamic and distributive response.

A B-cell generates antibodies that differentiates the "nonself" protein called an "antigen," such as a virus, reacts with the antigen, and is eliminated. The B-cell itself does not have the capability to eliminate antigens. However, a B-cell has an antibody molecule like an antenna on the surface of the cell, and the antibody corresponding to the antigen is compounded in large quantities by catching an antigen with the antenna. The B-cell is capable of memorizing the distinguished antigen, and generating many antibodies in a short time if an antigen is distinguished again. A T-cell distinguishes a self-cell that is detected by the B-cell antigen as nonself and eliminates it. A T-cell is generated in the thymus where it is strictly educated to discriminate self and nonself. Specifically, the T-cell is severely tested within the thymus and it is programmed to die. A few T-cells that only distinguish the self are converted into nonself, and do not destroy the self itself i.e. has the feature of self-tolerant.

When B-cells specify antibodies that recognize and react to stimulation and they bind to antigens with a strong affinity, then they are directly activated with the secretion of specific antibodies that recognize and react with appropriate antigens. If they bind to antigens with weak affinity, they need help or co-stimulation from T-lymphocytes called T-helper cells for activation. Forrest et al. [5] conducted research on security that focused on the educational function of this thymus. Their research has enabled the detection of strange illegal entry "nonself", without infringing on regular access "self" [6]. The second type of T-cells is T-killer cells, which destroy cells that are infected by intracellular parasite. T-helpers are needed for the activation of T-killers. This matching between antibodies and antigens explains the core of HIS and most of the first generation of AIS implementations [3]. The whole architecture of adaptive immune system is shown in Figure 2.7.

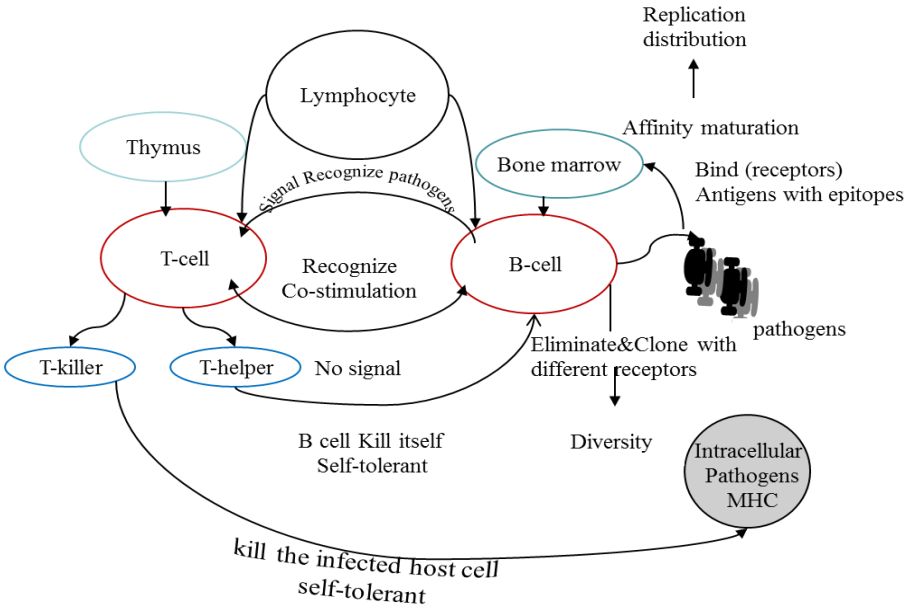


Figure 2.7: Architecture of Adaptive Immune System

2.5.2 Innate Immune System

The mechanism of the innate immune system will be activated when microorganisms or toxins enter an organism. The innate reaction is usually triggered when microbes are recognized by pattern identification receptors, which identify components that are conserved among broad groups of microorganisms, or when damaged, injured or stressed cells send out distress signals, many of which are recognized by the same

receptors as those that recognize pathogens. Innate immune system responds to pathogens, meaning the protection offered by this system is nonspecific. This system does not grant long-lasting immunity against a pathogen.

The innate immune system is the governing system of host defense in most organisms. Cells of the innate immune response are mainly Leukocytes. All white blood cells (WBC) are known as leukocytes, which are different from other cells of the body in that they are not firmly correlated with a particular organ; thus, their function is comparable to independent, single-celled organisms. Leukocytes are able to move freely, interact, and capture cellular debris, foreign particles, or invading microorganisms. Unlike many other cells in the body, most innate immune leukocytes cannot divide or reproduce on their own; they are produced by the bone marrow [47]. The innate leukocytes include: natural killer cells, mast cells, eosinophils, basophils and the phagocytic cells including macrophages, neutrophils and dendritic cells, and function within the immune system by identifying and eliminating pathogens that might cause infections [48].

One of the important innate immune system cells are Dendritic cells (DCs) that are identified as Antigen Presenting Cell (APCs), and perform as natural data fusion agents. In the human body, DCs do not have the adaptive capability of the lymphocytes of the adaptive immune system. However, DCs have a dual role, as trash antennas for tissue debris and as controller of the adaptive immune system. DCs are present in three statuses of differentiations: immature, semi-mature and mature, which determines their exact role [48]. Inflection between the different statuses is dependent upon the receiving of signals at the initial or immature status. Overall, there are two classes of danger signals; those which are generated endogenously i.e. by the body itself, and by exogenous signals which are derived from invading organisms e.g. bacteria. Signals that point to damage cause a transition from immature to mature. Those signals indicating good health in the monitored tissue cause a transition from immature to semi mature. The signals in question are derived from numerous sources, including pathogens, from healthy dying cells, from damaged cells and from inflammation. Each DC has the capability to combine the relative extent of input signals to produce its own set of output signals.

The adaptive and innate immune systems are integrated systems. A comparison between the innate and adaptive immune system is shown in Table2.2. The biological mechanisms discussed in this section are important to this thesis because they provide specific details on how cells are structured and interact. The structure, role, and involvement of DCs in the overall immune system dynamics show many interesting mechanisms that could be modeled for computer security system. The two systems complement one another to build an immune system that can distinguish and eliminate efficiently the intruder that entered the body, because some kinds of immune cells perform their own task while others cooperate with each other. The immunity cells of the two systems are reciprocally activated by stimulating each other, and powerfully remove cells infected by an antigen, as well as the antigen itself.

Table 2.2: Comparison between Adaptive and Innate Immune System [45]

Adaptive immune system	Innate immune system
Pathogen and antigen specific response.	Response is nonspecific.
Delay time between exposure and maximal response.	Exposure leads to immediate maximal response.
Cell-mediated and humoral immune response.	Cell-mediated and humoral immune response.
Experience directs to immunological memory.	No immunological memory.

2.6 Developed Artificial Immune System

Artificial immune system (AIS) has been used to solve many problems in different domains. In 1986, the combination of immune system and machine learning was first published by the researchers in [30] who initiated the field of artificial immune system. This was followed by a considerable number of other researches, which eventually contributed to the development of AIS. Since then, AIS has been built for a

wide range of applications with successful implementation; they include data mining, neural network, fraud detection, document classification, and intrusion detection and prevention systems. The details are shown in Figure 2.8. AIS approaches have gained many successes and accomplishments in many cases.

AISs can be classified into two different implementation mechanisms: network-based model and population based model. Currently, an offspring of the combination of both to construct a hybrid is also being used. Network based model is based upon idiotypic network theory which states that AISs are built on algorithms that recognize that interactions occur between antibodies and antibodies, as well as between antibodies and antigens [49]. Population based model utilizes negative selection or clonal selection as the technique of producing a population of detectors. Most of AISs intrusion detection and prevention systems use the latest model.

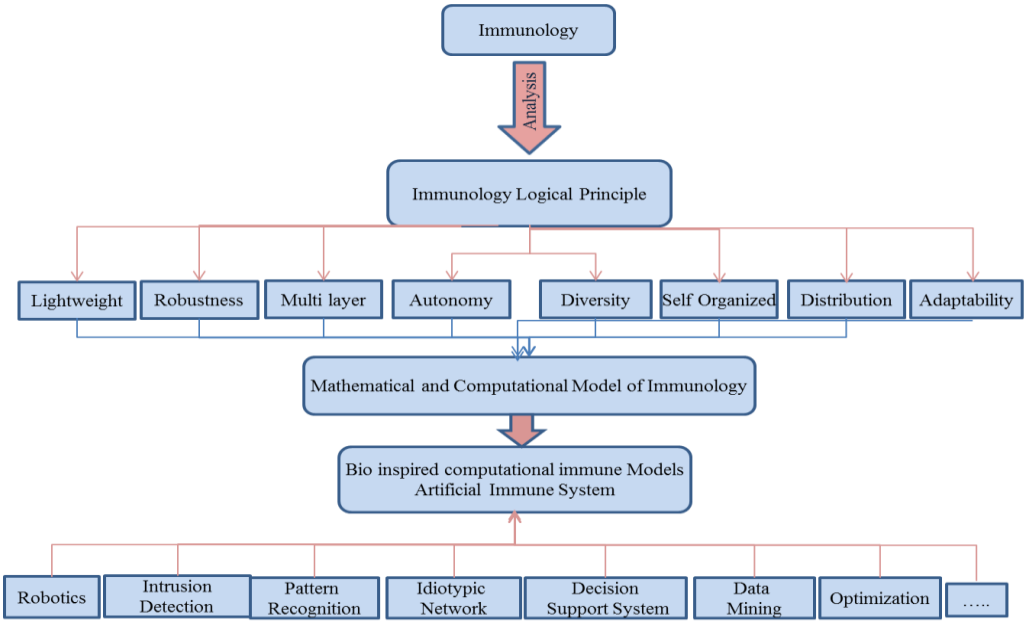


Figure 2.8: Implementation and Research Area of Artificial Immune System

During last two decades, many immune system approaches to IDS and IPS have been introduced. There are three major extractions, and accordingly three different views: conventional algorithm, the negative selection paradigm, and the danger theory.

Conventional algorithm identifies some traits of the HIS that make it attractive for virus detection purposes, and implements them using established algorithms. The

designed algorithm is constructed of five major stages, each inspired by the HIS. Computer viruses can be extended more speedily than traditional signature-based approaches involving manual creation and distribution of signatures, which renders the latter as not effective. Therefore, the developed conventional algorithm is focused on automatic detection of computer viruses and worms. The developers are interested in creating a system that is able to automatically detect and respond to viruses Kephart J.O. et al [50].

J. O. Kephart et al, first proposed a system capable of detecting viruses using either fuzzy matching from a pre-existing signature of viruses, or using integrity monitoring which monitored key system binaries and data files for changes. Despite the modifications that had been performed on subsequences algorithms to resemble an immune mechanism in order to improve the reliability, in some cases, it could be argued that they did not allow all of the possible features of the immune system to be exploited, such as the feature of self-organization or disposability.

When pathogens, disparaging microorganisms such as viruses or bacteria, enter the body, the immune system eliminates them and returns the body to a healthy condition. Thus, the purpose of the immune system is often seen as that of a protector or defender of the body. Since the immune system reacts to the pathogen “nonself” but not to the body “self”, it also seems logical to conclude that the immune system provides this protection by discriminating the “self” from the “nonself”. The defense mechanism characterized by “self-nonself” discrimination has shaped the basis of the majority of immunological models since the middle of the last century, and this view of the immune system is still widely accepted by currently immunologists [51].

Previous models of immunity have been based around the idea that host constituents (self) are ignored by the immune system, while other elements: pathogens, foreign substances or altered self, are reacted to. Based upon the mechanism of discriminating “self” from “nonself”, Forrest et al. [6] proposed a new algorithm known as negative selection (NS) algorithm. The algorithm consists of three phases: defining self, generating detectors, and monitoring the occurrence of anomalies.

In the first phase, it defines “self” in the same way as other anomaly detection approaches establish the normal behavior patterns of a monitored system. It identifies the profiled normal patterns as “self” patterns. In the second phase, it generates a number of random patterns that are compared to each self-pattern defined in the first phase. If any randomly generated pattern matches a self-pattern, this pattern fails to become a detector and thus it is removed. Otherwise, it becomes a detector pattern and monitors subsequent profiled patterns in the monitored system. During the monitoring stage, if a detector pattern matches any newly profiled pattern, then a new anomaly is considered to have occurred in the monitored system.

Negative selection algorithm is most popular; it has diverse features in solving IDS problems, particularly for anomaly detection. However, there are two drawbacks to utilizing the NS algorithm, namely scalability and coverage, and these are the main barriers to its success as effective IDS [3]. When negative selection algorithm was applied to a large and complex data search, it suffered from problem of generating excessive number of false positives [43]. Even though efforts have been made to solve this problem, however, still other unresolved issues are preventing the NS algorithm from being effective IDS. Following Forrest, many researchers working on the “self” and “nonself” concept have developed different algorithms, for examples in [8], [52]-[55].

Danger theory is a different immunology theory for IDS and IPS. The particular characteristic that makes this model different from other immune theories is that according to the danger theory immune response is triggered by unusual deaths of self-cells. Danger theory (DT) recommends that foreign intruders, which are dangerous, will encourage the generation of cellular molecules (danger signals) and initiating cellular stress or cell distress (dies by necrotic or abnormal death). Pathogens, which have damaged the body cell, sends danger signal to the Dendritic Cells (DCs) or Antigen Presenting Cells (APCs). Harmony is their ability to activate APCs and thus drive an immune response; antigen is swallowed from the extracellular environment by DCs in their immature state and then processed internally [11].

During processing, antigen is fragmented and attached to main histo-compatibility complex (MHC) molecules. This MHC antigen complex is then

presented under definite conditions on the surface of the DCs. As well as extracting antigen from their surroundings, DCs also have receptors, which respond to a range of other signaling molecules in their environment. Certain molecules, such as lipopolysaccharide, collectively termed pathogen-associated molecular proteins (PAMPs) are common to the entire classes of pathogens and bind with toll-like receptors (TLRs) on the surface of DCs. While other groups of molecules, termed as danger signals, such as heat shock proteins (HSPs), are associated with damage to host tissue or unregulated necrotic cell death and bind with receptors on DCs. In addition, there is another class of molecules related to inflammation which is called endotoxin, or lipopolysaccharide (LPS). This substance is present in the outer covering of some types of bacteria; also, it interacts with receptor families present on the surface of DCs. The current maturation state of the DC is determined through the combination of these complex signaling networks [56] which can be considered as input signals. DCs themselves secrete cell-to-cell signaling molecules called cytokines, which control the state of other cell types. The number and strength of DC cytokine output depends on its current maturation state.

The proposal presented by Aickelin and Cayzer [11] to DT model has encouraged many AIS, mainly computer security developers, to discover the potentials of danger theory [10], [57], [58]. Our literature review has shown that to date only a few efforts have been made on this new idea. We look for a concrete base for future research and application to AIS based on DT. Our system uses the danger theory as for inspiration to intrusion prevention system integrated with adaptive immune system mechanisms, mainly T-cell and B-cell mechanisms.

2.7. Self – Healing System

The majority of attacks on computing systems occur rapidly enough to discourage manual defense or repair. It appears, therefore, that defense systems must include a high degree of autonomy of repairing attack damages. Recent advances have been directed to an emerging interest in self-healing software as a solution to this problem [59]-[61]. Self-healing mechanism represents a system with the ability to observe, find, diagnose and act in response to system breakdown. Self-healing applications

must be able to examine system failures, evaluate restrictions inflicted by system environment, and apply suitable adjustments. In order to autonomously discover system malfunctions or possible expected collapses, it is necessary to know the expected system behavior.

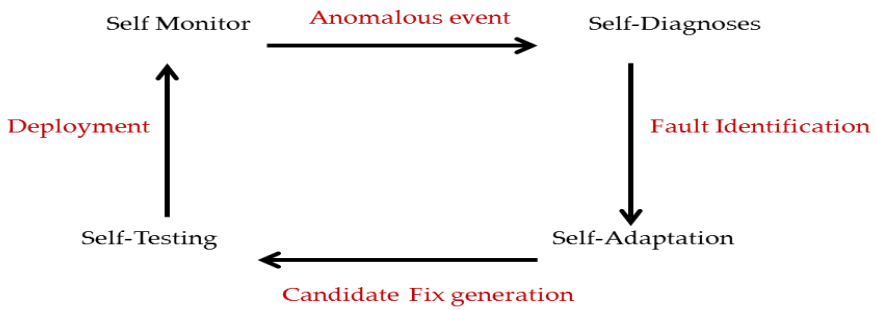


Figure 2.9: General Architecture of a Self-healing System [59]

Autonomic systems must have knowledge of their own behavior in order to determine if the behavior is reliable and expected in relation to the environment. In new environments or in diverse scenarios, new system behaviors can be observed, and the knowledge module must evolve with the environment. Moreover, the self-healing framework enhances the incessant and autonomous monitoring, diagnosis, repair, and remediation of software fault. Applying self-healing properties to network systems could present a way to alter the current fault finding in network systems subjected to various attacks. Many researches of self-healing modeling and techniques have been introduced in [62]-[65].

Figure 2.9 shows the general architecture of a self-healing system. When an abnormal behavior is detected, the system enters a self-diagnosis mode that aims to categorize the fault and extracts as much information as possible with respect to its source, symptoms, and collision on the system. Once these are recognized, the system tries to adapt itself by generating candidate fixes, which are tested to find the best mark state.

Generally, such architecture is composed of two high-level elements: the software service, which integrity and availability we are interested in improving; and the elements of the system that perform the monitoring, diagnosis and healing. The self-healing components can be viewed as a form of middleware and must cover four

aspects: fault-model or fault hypothesis, system-response, system-completeness and design-context.

A fault-model of self-healing system functions to identify what faults or failure to be self-healed including fault duration, fault source such as operational errors, defective system requirements or implementation errors…etc. System-response embraces the characteristics of fault detection, degree of degradation, fault response and an attempt to recovery action or compensation for a fault. Absolute restitution of functionality may not always be possible for a self-healing system after a fault. Such ability is limited by built-in redundancy in self-healing system.

The system-completeness feature deals with actuality of knowledge perimeters, incompleteness in patterns and designs thereof. It also deals with the problem of system self-knowledge and system evolution. Meanwhile, system design concentrates on the problems of abstraction level, component-level homogeneity, system linearity, system-scope, pre-deterministic behaviors, and user involvement aspects [66], [67]. Self-healing systems have been developed primarily as a result of the lacks of other techniques, whether in separation or combination, to present an adequate solution to the problem of software reliability. In particular, self-healing techniques attempt to hit equilibrium between reliability, assurance, and performance; where performance generally is an inverse relationship to the first two.

The main difference between self-healing, and the traditional fault-tolerant architectures and techniques are that, the former aims to identify and remove the origin cause of the fault, while the latter generally only brings the system to a state from which it can resume execution. Thus, while fault-tolerant systems can be viewed as primarily geared for rarely occurring failures [61], self-healing architecture is combined to complement IPS for more autonomous damage repair and system survivability.

2.8 Agents and Autonomous Computing

Autonomic systems are characteristically distributed, complex, and concurrent systems, comprised of multiple interacting autonomic elements that often exhibit

emergent behavior. The design and development of such systems is a non-trivial task that by definition requires specific software engineering approaches, including the use of specialized modeling techniques. An agent is a computer program that acts autonomously on behalf of a person or organization. Agents can be immobile, where they execute only on the system where it has begun implementation; or mobile, where it can be in motion to a remote system for implementation [68]. The main characteristics of agents are:

- Intelligence: it is the capability to acclimatize to status taken by the dynamics of the system.

- Cooperative behavior: it is the ability to contribute information among agents and/or discuss a common policy. This encompasses events that lead to an overall adequate performance.

- Autonomy: it allows agents to implement tasks without users' interaction. An agent displays autonomous behavior if it is able to complete an assigned task by suitably choosing a strategy from a set of probable strategies.

An agent in an autonomic system is proactive, and possesses social ability. For decentralized autonomous cooperative IPS and self-healing system, we use the benefit of those agents; this allows for exploitation of the intrinsic properties of the intrusion prevention and self-healing solution space and of applications in the intended implementation environment.

2.9 Related Works

It is quite necessary to conduct a thorough research and bring forward a realistic solution for coordinated intrusions since they have already become the major threat to the security of network systems. However, at present there are only a few published works that directly address this problem. Nevertheless, several approaches for intrusion detection system that are based on ideas of the human immune system and agent paradigm have been developed.

Morton S. in [69] presented the idea of danger model to autonomic defense systems. Danger model of computer immune system and its application to attack defense, create a fully decentralized model. The main models are co-stimulation using both indications of an attack (knowledge-based or behavior based) with indication of real danger or damage. Using these two detection models enables the probability of an autoimmune feedback in the active defense network to be reduced.

Morton also looked at the characteristics of malware that are most important because they cause damages. He defined the vector characteristics for some given types of malwares. The article discussed the limitation of digital immune system and the performance of discrimination between “self” and “nonself”, and the significance of using danger theory model, which relies on danger signals from injured cell to activate the immune cells (T-cell) and thereby the appropriates B-cells to eliminate the antigen. Concept of ”self “ and “nonself” is still contained in the danger model, albeit in diminished role.

Danger theory assumes that the immune system uses both the sense of self and the immunological memory i.e. it inherits the concept of signature aging, but combines with the danger signals in a new form of co-stimulation in order to respond to the threat. The author realized that co-stimulation through a signal that identifies the threat as dangerous is required to confirm an attack. He used trust danger sensor to be false positive free by design, and to deliver specific information about the attack to confirm its authenticity. The single sensor provides two signals: one for general detection and one for specific evidence. The model proposed by Morton did not deal with password-guessing attack specifically.

All together, the model has been successful to produce a system defense that avoids most drawbacks of central analysis and deployment, and at the same time, it is able to deal with problems of autoimmunity, if sensors are correctly designed and deployed. However, the model still needs to be implemented at larger scale to validate the applicability of the danger model in autonomic network defense.

In [70] the research has outlined a comprehensive workflow for the model with the use of agents and commercial off-the-shelf products to work together to counter malware. The new idea in this work is in providing a self-healing system that is

outside the software architectural descriptive handling (while avoiding the weaknesses of the current models); which utilizes the achievements of the market security products to provide a complete self-healing package from malware in a controlled environment. The research has provided the ground for a complete implementation of a product which can handle programs considered untrustworthy. The agent uses techniques such as sandboxing, processing priority and bandwidth control to quarantine the malicious code and prevent it from spreading.

Another model of network intrusion detection based on artificial immune system and mobile agent paradigms was presented by Azzedine. B and Renato B .M [71]. The structure of the model is based upon registries signature analysis using both syslog-ng and logcheckunix tool. The tasks of monitoring, distributing intrusion detection workload, storing relevant information, and ensuring data persistence and reactivity are carried out by the mobile agents to improve the security of complex computer communication networks, which represent the leukocytes of an artificial immune system.

The presented model is a real time based intrusion detection and communication. It is host based and adopts the anomaly detection paradigm. The intrusion detection tasks are distributed among a number of computer hosts in order to improve accuracy, and thereby allowing the implementation of distributed detection scheme. The IDS system generates three groups of data according to the type of network intrusion detection: attacks, security violations, and security events. The system is based upon the anomaly detection paradigm while it continuously monitors activity registries. It has both passive and active post detection behavior.

The system uses mobile agents which are dependent only on their execution environments and can be designed as an integral component of an intrusion detection system. They have task oriented functions and can be updated dynamically. The results showed that the average anomalous event percentage is around 8.5%. The average false positive was 80.82% and the true positive was 19.8%.

A scalable system that makes use of automated worm detection and intrusion prevention to stop the spread of computer viruses and internet worms using extensible hardware components distributed throughout a network is described in [72]. The

contribution of their work is in the management and configuration of large numbers of distributed and extensible IPSs which, decreases the rate at which worms and viruses spread. They implemented active network management software and extensible hardware systems that can work together in order to protect high-speed networks from fast outbreaks of new internet worm and viruses. Their system can be used to stop an attack and prevent a worm from spreading; this type of protection can be provided with minimal impact on overall network performance.

Robert L. Fanelli presented a hybrid model for network intrusion detection that combines artificial immune system methods with conservative information security methods [73]. The Network Threat recognition with Immune Inspired Anomaly Detection or NetTRIIAD model incorporates misuse-based intrusion detection and network monitoring applications into an innate immune capability inspired by the immunological Danger Model. NetTRIIAD demonstrates recital improvements over a conventional, misuse based network intrusion detection system. The NetTRIIAD model builds upon trusted information security tools, preserving their effectiveness while providing improved performance with the addition of immune inspired components. The positive predictive value of the model was 0.65 which is an improvement from the current widely used intrusion detection system SNORT which is 0.38. The model also has improved in reducing the percentage of the false positive events.

The present work looks for extending NetTRIIAD to move beyond threat recognition and include automated threat response, which would be a step closer towards a computer immune system and would benefit from the improved positive predictive value. The features in a NetTRIIAD antigen contain sufficient information to create firewall rules to block or shape the associated traffic. Such work could extend the adaptive immune metaphor beyond T-cell activation, adding elements inspired by B-cells and antibody production. Improvements to the danger model signal generators, possibly to examine additional external data sources, could gather better evidence of threats and improve detection. Similarly, a mechanism for accurate reactions to “dangerous self”, suggested by the Danger Model, would permit NetTRIIAD to recognize threats hidden in ‘normal’ network traffic. This can be

possible by using the danger theory as a basis; including a prevention mechanism and implementing only misuse detection methodology.

Authors in [74] presented a model of network security based upon the theory of artificial immune system. The concepts and formal definitions of immune cells are given, and dynamic evaluative equations for self, antigen, immune tolerance, mature-lymphocyte lifecycle and immune memory, and the hierarchical and distributed management framework are used to provide an effective solution for network intrusion. Furthermore, the inspiration of dynamic immunological observation phase is applied for enhancing the self-learning ability to adapt continuously to a variety of environments. Their model has the features of real-time processing that provides a promising solution for network surveillance. Agent paradigm is used to implement the model, but no response is taken to prevent or heal the intrusion harmful behaviors.

In [75], the author put forward the design and implementation of an agent-based system, constructed using Java Agent Development Framework (JADE), in which the agent's main task is detecting vulnerabilities and exposures. Each agent can exchange knowledge with others in order to determine if certain suspicious situations are actually part of an attack; this procedure allows them to warn each other about possible threats. In this system, the external source of vulnerabilities is used to keep the agent system updated, The Internet Categorization of Attacks Toolkit (ICAT) Meta base, where a search index of vulnerabilities in computerized systems, is used by the authors. The ICAT binds the users to the diverse public databases of vulnerabilities as well as patch sites, thus allowing us to find and to repair the existing vulnerabilities in a given system. ICAT is not a proper database of vulnerabilities, but an index pointing to some reports of vulnerabilities as well as the information about patches currently available. The system uses agent platforms allocated through the network to scan and interact with each host. The information collected by each agent is then used to build a common knowledge base. The model did not include any mapping from the immune system techniques.

Begnum and Burgess [76] introduced a design for an immune inspired intrusion detection system that combines cfengine and PH. They were motivated by the need to offer a better, decentralized feedback mechanism for the pH system, and enhanced

detection capabilities for cfengine, as well as the necessity to accumulate more detailed data for extended research. By integrating the signals from both systems, they wished to provide a more robust, accurate and scalable anomaly detection system. Their approach is to combine the two systems that aim the possibility of using the pH/cfengine combination to provide an automated reaction method, one that is able to destroy abnormal processes.

A hybrid IDS that combines pure anomaly detection with the provision of higher-level information about the detected anomalies is proposed in [77]. The key feature and novelty of their system is the use of separate components for anomaly detection and attack classification. Detection is about recognizing that a connection is anomalous, while classification is about determining the broad attack type of the connection. The advantage of using separate components is that in the first stage, the system is able to perform pure anomaly detection. This approach should be contrasted with the misuse based approach that looks for signatures of known attacks, since such systems cannot detect attacks for which a signature is not present. The attack detection rate in their approach is on anomaly attacks only, since the use of anomaly detection may provide a further advantage against such attacks. Finally, the technique by Simon T. P and Jun. H did not include any technique for prevention or healing of intrusion abnormal activities.

A distributed intrusion detection system by Wong [68] is a self-monitoring system to identify corrupted intrusion detection system. One way of self-monitoring in IDS is to verify each other. Mobile agents can do this using an immunity-based diagnostic method modeled on idiotypic network theory. In simulations, the credibility of a normal intrusion detection system remains near 1, while it will fall to about 0 for corrupted intrusion detection system, thus enabling identification of the latter. He also confirmed what effects some parameters have on the diagnostic capability. Wong has shown that the most noteworthy advantage of the multiagent paradigm is its ability to scale and adapt to the properties of the decentralized autonomous cooperative system (DACS). Through dynamic changes in behavior of individual agents or the agent population at large, an agent-based DACS will be able to sustain constant performance under a variety of possibly adverse conditions.

Alexander Krizhanovsky presented an approach to an Intrusion Prevention System (IPS), encouraged by the Danger Theory of immunology and tried to solve this problem by analyzing more sources of information [12]. His work showed how to link the entities which contribute in the interactions described by this theory with components of the operating system for synthesizing of IPS. He also introduced a technique inspired by the clonal selection mechanism of the HIS which links the anomaly behavior of system processes with received network traffic, and able to generate new signatures of network intrusions on the fly. His work was too simple heuristically, which needs significant improvements. Also, it is not so clear as how to observe processes created by the suspended running of programs via *atd* (run jobs queued for later execution in linux) and family. His model, also, did not include the healing mechanism.

The researchers in [78], [79] presented a detection system that detects abnormal behavior nodes in a mobile ad-hoc network based on AIS model. Their AIS integrates adaptive-based negative selection with an innate detection mechanisms inspired by the danger model. Danger signals, in their studies, are dreadful conditions in network communication quality computed by the whole of packet defeat, are used to specify normal antigen for use during negative selection training phase and as a co-stimulatory signal to trigger adaptive immune cells. They assessed their AIS on a network simulation, which generated confident results and indicated that their danger signal was powerful to reduce the number of false positive rates to 0.001-0.003.

Dendritic Cell Algorithm (DCA), which is based on the observed function of natural DCs, was initially presented in [13], [80], [81].The purpose of DCA is to correlate disparate data-stream in the form of antigens and signals, and to label groups of identical antigens as ‘normal’ or anomalous. This algorithm is population based with each ‘cell’ expressed as an agent. The DCA presents how anomalous a group of antigens is, not simply if a datum is anomalous or not. This is accomplished by associating a time series of input signals with a group of antigens. The signals used are prenormalized and preclassified data sources, which imitate the behavior of the system being monitored. The co-occurrence of antigen and high/low signal values forms the basis of classification for the antigen data [4]. The DCA has greater confidence on the signal processing aspect by using multiple signal models. DCA

uses expert knowledge to assign input signals to the appropriate category. The DCA also achieves anomaly detection with a comparatively low rate of false positive errors. DCA needs investigation methods for automated signals selection and other time-dependent data.

The Toll- like receptor 'TLR' algorithm is based on two populations of interacting cells namely DCs and T-cells. DCs implemented in TLR collect antigens from antigen store, and process signals. In TLR, DCs are created as immature detectors and sample signals, and antigens for a finite period. If DC receives a signal during antigen collection, it is termed as mature, and conversely, DCs that did not detect the presence of a signal are termed as semi mature. Once a DC life is complete, the cell is transferred to a 'lymph node' in which it is compared against a population of T- cells. The same representation as the antigen, presented by the DC population specifically T-cells, are entrusted to sensors known as "receptors". The T-cell receptor is responsible for the matching and interaction with DCs generated during the training phase. T-cells have two states: activated if a matching antigen is presented by mature DC, or deleted if a matching antigen is presented by a semi-mature DCs. If the populations of T cells are activated, anomalies are detected. The signal used in TLR are referred to as danger signals, implying signals which may represent 'damage'. TLR has been exposed to give an improved performance over negative selection. This incorporates a marked reduction in false positive errors in contrast to a pure negative selection-based approach. The core features of TLR is its requirement for training data performed on multiple types of cell agent which appears to add an extra element of tolerance to the generation of false positive errors. However, training data can be difficult to collect. Both algorithms contain the concept of 'tissue'. Both carry out a kind of temporal association between signals and antigens, and both consist of DCs performing a computational task [56], [80].

S. Liu et al. [82] introduced an immune multi-agent active defense model for network intrusion. The model presented has gained some of the IDPS design features. First, the model has self-learning feature where the model can detect the memory mechanisms of both known and unknown attacks. Second, a multi-layer feature that gives the notion that a vaccine has been captured, and the active defense is achieved in different layers and nodes. Third, the real-time defense feature where the model

can quantitatively assess the danger status faced by the network. Finally, a robust feature that the model uses as distributed architecture, so that the attack on a single node cannot influent the others. The results obtained show that this model changes the isolated and inactive status in traditional network security models. It is a worthy solution to founding dynamic defense for the network security.

2.10 Chapter Summary

The critical services, intrusion prevention system, human immune system, self-healing system and multiagent concepts and mechanisms have been studied and analyzed. In this review, we have discussed the security needs of critical services networks. High performance of IPS can be a desirable security system for critical services networks, which can be accomplished by fulfilling the requirements of IPS design features. Meanwhile, the IPS must be distributed as a hybrid system i.e. host-based and network system. This hybrid system must integrate the anomaly and misuse methodologies of IPS. The mechanisms and features of HIS have been studied and HIS has been found to have good metaphor mechanism for IPS and for meeting the criteria mentioned earlier. The self-healing mechanism is discussed as an enhancement feature of the system to recover from harmful events. The study of multiagent system has provided direction towards accomplishing the criteria of autonomous and decentralized IPS, and self-healing system. Finally, the related works of other developers and researchers have also been studied and discussed. Those related works cover different aspects of IPS, AIS, and self-healing algorithms, mechanism, techniques, and point of views to HIS. Based on this discussion, the analysis, abstract model and specification for the proposed system are established in the next chapter.

CHAPTER 3
BIOLOGICAL INSPIRED INTRUSION PREVENTION AND SELF-HEALING SYSTEM (BIIPSS) ANALYSIS AND ABSTRACTION

3.1 Chapter Overview

This chapter investigates and analyzes the abstraction model of the IPS and self-healing system. This investigation and abstract derivation are later used in designing an efficient IPS and self-healing system. In this work, we are following the derivation model used in artificial immune system proposed by [83]. In order to construct and derive a biological abstract model of the IPS, first, analysis of the IPS techniques and methodologies is required in order to specify the appropriate methodology and technique.

Secondly, we have to select which immune mechanisms are suitable for achieving the requirements of high performance IPS. This is followed by the features that need to be mapped from the human immune system to IPS, before arriving at a new conceptual framework for developing a new artificial immune system. Then, the model for IPS and self-healing system is developed using the new abstract artificial immune system.

3.2Comparison between IPS Techniques and IPS Methodologies

In chapter two, we have defined the techniques and methodologies that are currently being implemented in IPSs. To build a robust IPS, we make a comparison between the *techniques* and *methodologies* for IPS detection and prevention. The comparisons between IPS techniques are shown in appendix A.2. As shown in the TableA.1, there are three different techniques: host-based, network-based and network behavior analysis. The comparisons have been made based upon four criteria: the monitored

information, components required by each technique, security capabilities and management techniques. In Table A.2 Appendix A.2, we compare the methodologies of IPS system based on effectiveness, methods and limitations.

To achieve the design features mentioned in chapter 1, we built our model based on the criteria that combine host based and network based techniques into a hybrid IPS. By using hybrid techniques, we manage to capture high security capabilities and cover the scalability design feature. Meanwhile, misuse detection and anomaly based detection methodologies are applied by default when immune system features are mapped to IPS; this contributes to reducing the false alert, which is one of the main objectives of this study. Throughout this chapter, we will discuss how we determine the technology and methodology that are suitable for validation of the search objectives.

3.3 Derivations of Abstraction Model for IPS and SH

The derivation of the abstract model follows the proposed conceptual framework for artificial immune system presented in [83] and Susan Stepeny et al. [84]. The authors in [84] have suggested that bio inspired algorithms to be built and analyzed in the perspective of a multidisciplinary conceptual framework that presents biological models and well-analyzed principles. Figure 3.1 shows a probable arrangement for such a conceptual framework.

In the framework, the probe like inspection and experiment are used to afford a sight of the complex biological system. A straightforward abstract representation model of the biology can be built and authenticated using this sight.

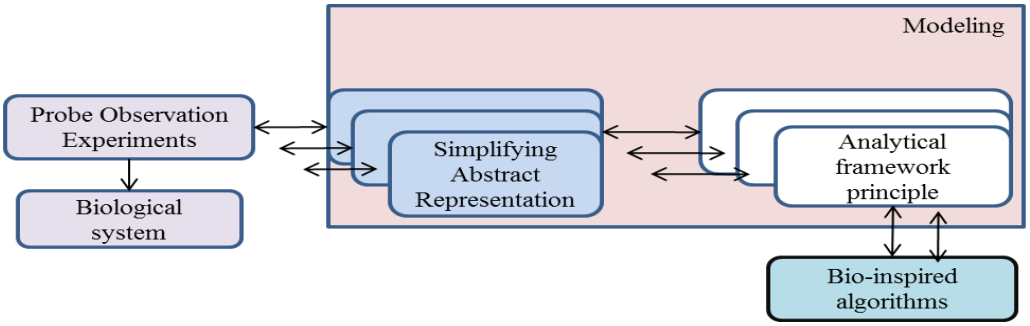


Figure 3.1: A Conceptual Framework for Biologically Inspired Algorithms [84]

From these biological models, we build and authenticate the analytical computational framework. Authentication may use mathematical analysis, benchmarking and engineering demonstrators. These frameworks provide the principle for designing and analyzing bio-inspired algorithms applicable to non-biological problems. To follow these frameworks, many questions in the research area have been suggested based upon the design features that are thought to affect complex behavior of the proposed system in general. These areas are related to Candidness, Diversity, Interaction, Structure and Scale (CDISS). Now we can briefly make a relationship between the design features and the CDISS.

Candidness: The communication between the immune system and the host is of one carriage system in a dynamic equilibrium coupled to an ever-changing environment. This biological feature is related to the design features: autonomy and self-organizing. This is mapped to distribution feature of the proposed IPS as a hybrid system i.e. host-based and network-based, and they can apply self-organizing design feature when using agents. As mentioned in section 2.5.2, chapter 2, this feature is mapped from the innate immune system that presents the mechanisms for controlling dynamic allocation of resources among of population of the agent's design feature i.e. distribution.

Diversity: The type and state of the human immune cell and receptor cell are diverse. The innate and adaptive immune systems have different sets of cells to guide the stimulation of distinct groups of functionally similar agents. The proposed IPS design feature: diversity and light weight.

Interaction: The innate and adaptive immune systems show how computation is largely communication, with immunity arising from cytokine network of signaling interaction between intercommunicating tissue cells. Both immune cells are specialist to access different informational levels. IPS design features: multilayered, adaptability, and self- organizing.

Structure: The innate immune system is composed of distinct subsystems. Function similarities can be seen between the innate and adaptive immune subsystems. At the same time, they are composed of interacting populations of human agents. Cells differentiation pathway provide an even more fine grained division of cells into types. IPS design features: self-organizing, distribution, and self-repair.

Scale: A diverse population of huge number of cells is the main feature of the immune system. One of the challenges in developing AISs is the need to simulate large population of agents. The utilization of emergent properties from different population of large numbers of simple agents, rather than a smaller number of more complex agents, along with distributed and parallel architectures for AISs may provide a way forward [85], [86]:IPS design features: distribution and diversity.

Many researchers have made some great efforts to define a general AIS engineering framework; in [85] the authors presented a general framework for engineering AISs. The description of this framework is shown in Figure 3.2. The application domain manipulates the representation of the AIS. They recommended shape-space representation, where the problem domain basically stands for different patterns named antigens which the AIS cooperates with using its own set of patterns termed antibodies. The extent of relationship between antigens and antibodies is assessed using a number of different possible affinity measures. Immune algorithms, using this problem representation, model specific immune mechanisms which control the production of antibodies. AIS solutions can be built in for the application domain by following the three steps: engineering framework of representation, affinity measures and immune algorithms.

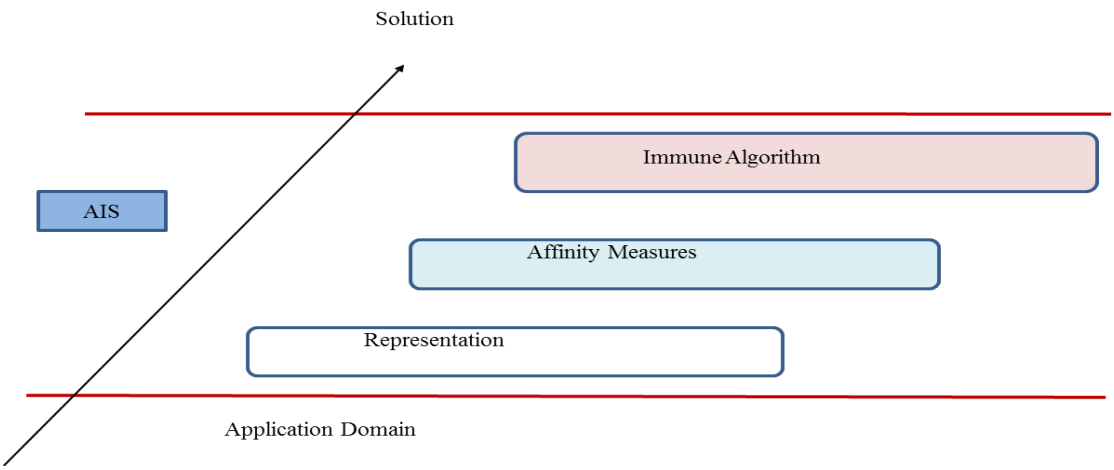


Figure 3.2: Layered Framework for Engineering AIS [85]

To develop the abstract IPS and SH model, we follow the steps detailed in Figure 3.3. The steps outlined in this figure implement integration between the conceptual framework and framework of engineering AIS. From this integration, we aim to present a new conceptual framework for AIS. The abstract metaphor model is then

established. The design specification and algorithms of the model that will be discussed in the next chapter will complement the abstract representation of the system.

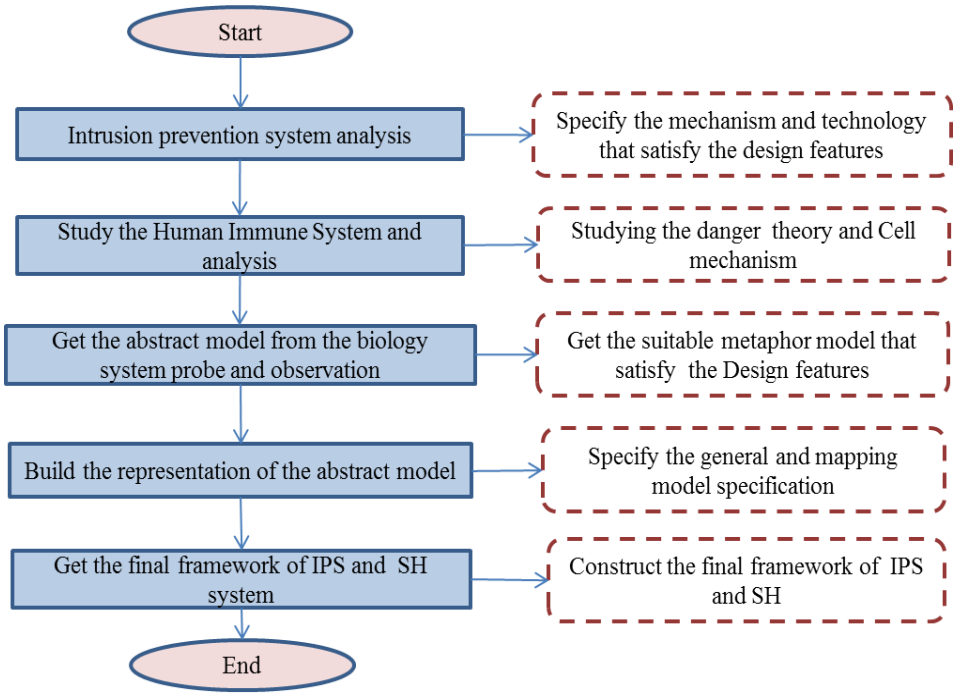


Figure 3.3: Development of Abstract Model of IPS and SH

3.4 Why Abstract and Engineering Frameworks

When starting to analyze the inspiration of IPS and SH from the human immune system to obtain the abstract model, we realize that it is important to find a standard framework for abstracting immune inspired computational metaphor. This necessity has motivated many researchers to follow and develop a conceptual framework for AIS. In our study, we follow most of the conceptual framework described by [84], [80]; in addition, we also introduce some improvements to the framework due to some reasons which are discussed below.

A multidisciplinary field: The AIS is a multidisciplinary field that needs the knowledge of computer science, mathematics, physics, biology, immunology, and other inspiring fields. Therefore, the research in this area is comparatively difficult and has been rather slow to develop. It requires repeated learning cycles at various stages. Therefore, it is important to know the sequences of the research in hand

through an appropriate framework. This will be useful in the search for fundamental multidisciplinary knowledge and minimize idleness along the way in AIS research.

Inexpert computational model: Artificial immune system is a relatively new branch of application in the field of computation. At present, this area is suffering from the lack of obvious guidelines for abstracting descriptions; therefore, the growth of the field is relatively slow. We believe that certain distinguished aspects of experienced frameworks for abstracting AIS descriptions can be integrated to get a good general framework. Our framework might be considered a useful contribution in this regard. This framework might be capable of providing guidelines to future developers of AIS.

Pass up the unnecessary: During this study and research investigation, we have discovered that it is actually complicated to maintain stability of multidisciplinary knowledge in one model. At one time, we find ourselves tending more on biology, while at other times more on computation and less consideration of mathematical derivation. We reduce this gap between biology and computing by introducing a new framework that puts computational model in consideration.

Important requirements: Considering that what we are looking for is important i.e. the aim of the model and requirements, we will have to concentrate on the boundaries of the multidisciplinary knowledge that are closer to the metaphor. A high-quality framework must underline what is necessary of high quality metaphor through related knowledge in multidisciplinary areas. Therefore, an appropriate framework will guide us to concentrate on the most urgent requirements for abstracting the metaphor.

Prefect control of framework development: It is quite inadequate to control and manage things without having prior experience in the field. We have developed a framework that improves the research process management and control. It should also be helpful for abstracting more than one metaphor at a time. This dictates the scalability of the framework.

Systematic development and faster inspiration: Due to improvements in the process of abstraction through framework guidelines, we expect a good metaphor in

relatively less time. A good metaphor will need less alteration efforts subsequently. This will save the researcher's time. This might also result in a number of good metaphors from are searcher or team of researchers. The framework must provide a way of knowledge and experience sharing. This kind of knowledge sharing is vital for the growth of multidisciplinary research like AIS. We have spent a significant amount of time for AIS metaphor abstraction. This framework will help avoid a reinvention of the wheel through knowledge sharing.

Developing the area of AIS science: To develop new framework is a contribution for AIS science. Any developers can obtain their own new experience through the newty of this framework. Our work is expected to give new impact in AIS research.

The foregoing discussions are the motivation for this research and for contributing the proposed framework. In the next section, we will demonstrate the steps involved in developing our new abstract framework. These steps are grouped to introduce a new multidisciplinary framework for new AIS applications.

3. 5 Observation of Biological System

The first step towards obtain the abstract model is to observe a suitable biological system from which we can draw some inspiration for the IPS and SH security systems. From an overview of the human immune system presented in chapter 2, we concluded that it is possible to achieve the design requirements of IPS and self-healing system by constructing an abstract model with mechanisms inspired by the integration of innate and adaptive immune systems, more specifically dendritic cell, T-cell, and B-cell based on danger signal model among these three cells. The three mechanisms are analyzed and how they cooperate to defend the human body from the aspect of *candidness* and *diverse* features mentioned above. For the IPS and SH system, we will introduce IPS immune agents that will work in the same manner as specific HIS cells defending the human body; the immune agents will be responsible for defending critical network systems that must have high security services. Next, we study immunology based on the danger theory as the second step towards developing the abstract model.

3.6 Immunology Based on Danger Theory

The approach of using Danger Theory is to understand the intrusion detection and prevention mechanisms in HIS and to capture the fundamental nature of these mechanisms through an abstraction process. In order to construct such immune-inspired IPS algorithms, the understanding of correlations of signals processed by the DCs of innate immune system, which includes characterization of the danger signals, is the key prerequisite. In this section, we have summarized the main principles of the Danger Theory which is used as the starting point for our work. These principles could be used for designing an intrusion prevention system. In the model, we have concentrated on the three mechanisms: DCs, T-cells and B-cells, which are considered as the key agents within the Danger Theory. In sections 2.5.2 and 2.6, we have discussed how the DCs in the innate immune system receive and input signals. Table 3.1summarizes the definitions and causes of these signals [15], [80], [81].

Table 3.1: Signals Definition in Innate Immune System

Signal	Definition
Safe	A result of normal cell death i.e. death for regulatory reasons.
Danger	A consequence of unintended necrotic cell death. The presence of danger signals may or may not indicate an anomalous situation.
PAMP	Pathogen - associated molecular proteins(PAMP), an indication of an anomalous state detectable by DCs. Protein expressed exclusively by bacteria.
Inflammation	The outcome of injury which amplifies the above three signals but is insufficient to stimulate DCs alone.

DCs collect the antigens whilst being exposed to environmental signal molecules by cell death and other events. The combination of signals determines the interaction with T-cells. T-cells process the information handed over by DCs and they are either

being cloned or die subsequently. Meanwhile, by using receptors on their surface, B-cells combine to antigens (they identify antigens) with a strong affinity above the threshold (there are two different levels of affinity between epitopes on the antigen surface and the receptors of the B-cell, below and above the threshold correspondingly). Upon activation, they secrete specific antibodies that distinguish and react with the proper antigens. If they bind to antigens with a weak affinity, i.e. below the threshold, they need help (co-stimulation) from T-helpers (a kind of T-cells) for activation. The second type of T-cells, T-killers destroy cells which are infected by intracellular parasite. T-helpers are also needed for the activation of T-killers. T-cells are therefore co-stimulated from both B-cells and DCs [15], [81], [87]-[89]. We look at the integration between innate and adaptive system as a suitable metaphor for creating an autonomous IPS. An overview of the different danger signals, which are immune system reaction on pathogens, can be simplified into a model with four types of signals as shown in Figure 3.4. Pathogen that has damaged the cell sends danger signal zero "PAMP" or "necrotic cell death" to DC or APC which may generates two signals: signal one is antigens recognition and signal two is co-stimulation confirmation that "this antigen is really dangerous" (danger signal). By receiving both signals one and two, the T-helper is activated, but it dies if it receives only the first signal.

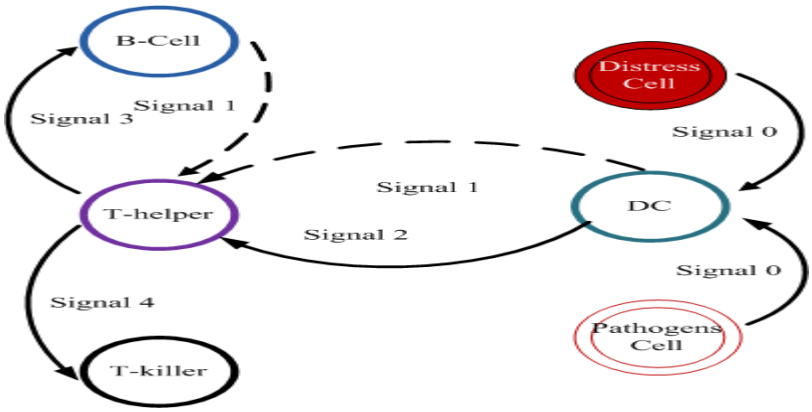


Figure3.4: Danger Theory Signals Model

In this model, the T-helper can receive signal one randomly, but APC sends the second signal only when it receives signal zero. Whilst B-cells send signal one to T-helper for activation, if they bind to antigens with a weak affinity i.e. below the threshold. The T-helper activates the T-killer by sending a four signal if it recognizes

an antigen presents in APC. The T-killer is able to destroy the infected cell. T-helpers co-stimulate B-cells if they recognize the antigens sent by B-cell with a weak affinity signal three [12], [80].

To decide the exact state of differentiation, the comparative concentration of output signal is used; expressed by the production of two molecules, namely the mature and semi-mature output signals. Revelation to PAMPs, danger signals and safe signals causes the increased creation of co-stimulatory molecules and a follow-on removal from the tissue and migration to a local lymph node. DCs interpret the signal information received in the tissue into a context for antigen presentation, i.e. the antigen presented in an overall “normal” or “anomalous” context. The antigen collected while in the immature phase is expressed on the surface of the DC.

Meanwhile DCs look for T-Lymphocytes (T-cells) in the lymph node and try to bind expressed antigen with the T-cells changeable area receptor. T-cells with a high enough affinity for the presented antigen are influenced by the output signals of the DC. DCs exposed to mainly PAMPs and danger signals are termed ‘mature DCs’; they produce mature output signals, which activate the bound T-cells. On the contrary, if the DC is exposed to predominantly safe signals, the cell is termed semi-mature and antigens are presented in a safe context as little damage is evident when the antigen is collected.

The balance between the signals is interpreted via the signal processing and correlation ability of these cells. The overall immune system response is based on the average systemic maturation status of the whole DC population on a per antigen basis. Figure 3.5 shows the input, output, and DC states. The link between IPS and Danger Theory can be explained as follows.

DCs carry out the function of antigen presentation, where distress found on cells or tissues are collected by DCs, processed to form antigen and presented to the adaptive immune system in combination with context information. The output information is derived through the DCs processing of various signals that are

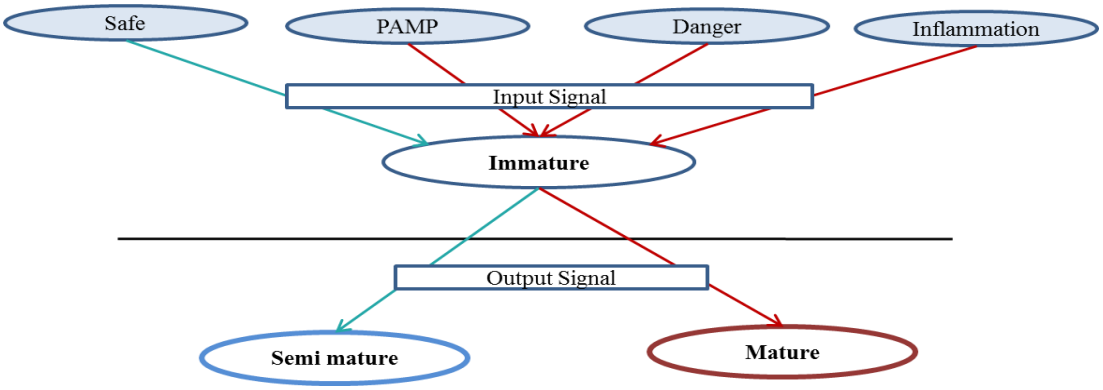


Figure 3.5: Dendritic Cell States and Input-Output Signals

produced due to the binding between antigens category and receptors. We used this mechanism from the innate immune system integrated with T-cell and B-cell to provide a new approach to intrusion prevention. Similarly, for DCs functionality, we observe the behaviors of network traffic in the form of a system call sequences or input data packet, and consider the antigens for the attributes and features such as network protocol headers, and port and socket as good categories with parameters. When the attributes, features and parameters are changed i.e. the data are subjected to abnormal behavior, then a danger alert is produced and prevention must established. Otherwise, the normal behavior is maintained and the network system will continue receiving the data packet or system calls. This is mapped from the Danger Theory model where the DC produces two signals: danger and safe in two different maturation states (mature or immature according to the antigens). For example in the "system process" category, each system process has a life span like biological cells; the disconnection in a network system could be either normal (like apoptosis, the normal death of cells) or abnormal (necrosis processes). Another example is TCP sessions, which can also die abnormally and feel distress in the case when a segment for an inappropriate port is received.

For our system, we could identify the biological cell for the category with parameters and processes, and antigen for any external input to the category. The external input could be network traffic, command line argument or environment variables. We can identify the 0 signal i.e. danger as the first sense of deviation from specific trained rules, T-helper for process behavior analyzing module or for scanning

the port or socket. Similar to signal activation from T-helper to T-killer (to kill pathogen cells), the module with first sense of deviation can generate activation signals to prevent intrusions that might stop access to the network system and block the network traffic. In Figure 3.6, we compare the similarities between human organ and network system for different critical services.

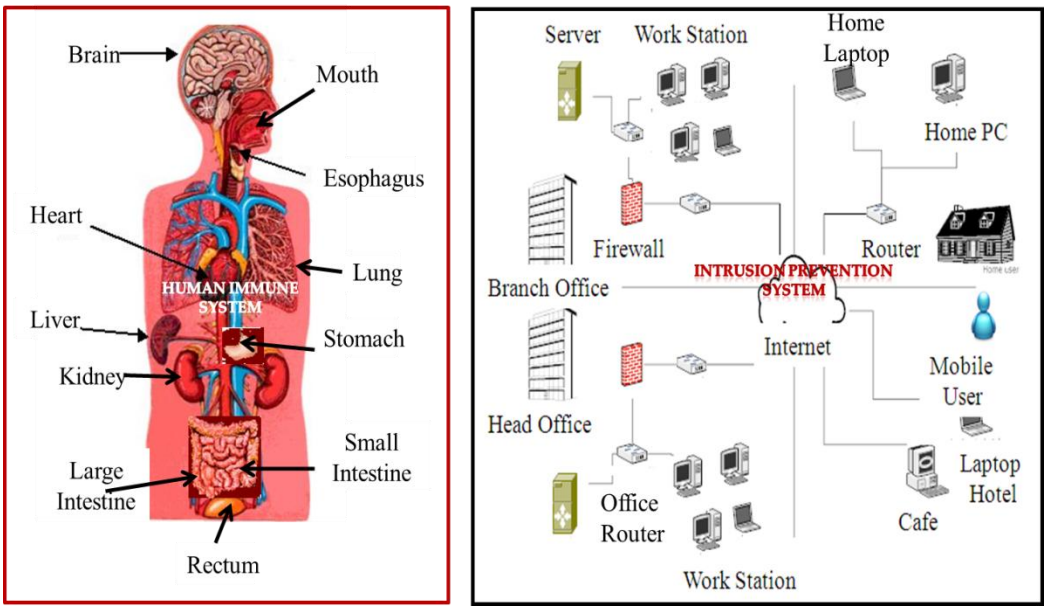


Figure 3.6: Human Body Organ and Network System Components

The sensitivity of each process category in the system is determined during the learning and training period. It is possible to increase the accuracy of intrusion detection and prevention by associating received network with categories behavior. If we add a direct call or categories flag, and features for any unspecified program to the legitimate user signature, false positive alerts will be prevented. Similarly, in many cases, by adding rules for accessing any system or connection to the signature, false negative alerts can be avoided.

During the training and rules learning period for network traffic, we have to identify the type of signature whether it is normal or dangerous signature. This is one of the similarities between HIS and IPS. When there is any activation of danger signal i.e. detection of intrusion like T-killer, the intrusion will be prevented immediately by the network traffic either by blocking or by disconnecting the network connection. A relationship between behavior category and network traffic makes dynamic generation of network signatures possible by clonal selection mechanism. For this

purpose, we classify danger zone as a combination of network received by observed categories and time, during which traffic is monitored.

This link properties mapped to IPS and SH multiagent system. First, the sense agent (DCs), based upon sensing the traffic, will produce signal 0 to the sense module (T-killer) to prevent the abnormal behavior, and immediately produces signal 1 to the analysis agent (T-helper), which in turn sends out signal 2 to the adaptive agent (B-cell). Adaptation agent (B-cell) activates clonal selection if it receives signal 3 from analysis agent (T-helper). In our research, the adaptive agent will produce another signal 2 to the self–healing agent, which keeps continuity of the system by healing the failure or damages caused by the intrusion. These mechanisms, mapped from HIS to IPS, ensure that the required design features are applied. Table 3.2 shows the mapping from human body to IPS and SH system.

Table 3.2: Mapping between Human Body and Network Systems

IPS and SH	**Biological System**
Network systems	Human body
Hosts	Organs
Security system	Immune system
SEA Agent	Dendritic Cell
Analysis Agent	T-helper
Prevention	T-killer
Adaptive agent	B-cell
Self-healing Agent	Cell-regeneration
Data features	Proteins
Classification features	Antigens
Abnormal behavior	Pathogens and damaged cell
Data classes	Signals
Multi layer IPS and SH system	Multi layer immune system

3.7 Autonomic Systems Properties and Approaches

The proposal for the abstract model is presented in this section. One of our main design features and research objectives is to develop an autonomous IPS with SH. Autonomic computing is a term first used by IBM in 2001 [89], [90] to describe computing systems that were said to be self-managing. Nevertheless, the concept of self-management and adaptation in computing systems has been around for some time. In the current state-of-the art in autonomic systems, the concept of self-management is usually set into having four essential properties: self-configuration, self- optimization, self-healing and self-protection.

Autonomic computing arises due to the need to reduce cost and complexity of owning and operating an IT infrastructure. A brief description of those properties given by J.O. Kephart et al. [19] is presented here.

• *Self-configuration*: An autonomic computing system configures itself according to high-level goals, i.e. by specifying what is most wanted, but not necessarily how to achieve it. This can mean being able to install itself based on the needs of a given platform and the user.

• *Self-optimization*: An autonomic computing system optimizes its use of resources. It may decide to initiate/introduce a change to the system proactively (as opposed to reactive behavior) in an attempt to improve performance.

• *Self-healing*: An autonomic computing system detects and diagnoses problems. The type of problems detected can be interpreted broadly: they can be as low level as a bit-error in a memory chip (hardware failure) or as high-level as an erroneous entry in a directory service (software problem). However, the significant aspect is that as a result of the healing process the system is not further harmed. Fault-tolerance is an important characteristic of self-healing. Typically, an autonomic system is said to be reactive to failures or early signs of a possible failure.

• *Self-protection*: An autonomic system protects itself from malicious attacks but also from end users who inadvertently make software changes, e.g. by deleting an important file. The system autonomously tunes itself to achieve security privacy, and data protection. Thus, security is an important aspect of self-protection, not just in

software, but also in hardware. A system may also be able to anticipate security breaches and prevent them from occurring in the first place.

Self-management requires that a system monitors its components (internal knowledge) and its environment (external knowledge), so that it can adapt to changes that may occur, which may be known changes or unexpected changes where a certain amount of artificial intelligence may be required. The approaches to autonomic computing system are classified orthogonally into two categories: intelligent multiagent systems and architecture design-based autonomic systems. In this research, the multiagent architecture is used to achieve the main design feature of our proposed bio inspired intrusion prevention and self-healing system.

Complex autonomic systems that are not composed of a single self-managing component can be built using multiagent. Every agent has its own goals, which drive its decisions. An agent in an autonomic system is proactive, and possesses social ability. The latter can potentially lead to instabilities of the overall system due to chain reaction of agents instructing other agents to change behavior. Moreover, it is complicated and not easy to define the individual goals of the agents such that the desired global goals of the system under development are accomplished [68], [92]. In an autonomic system, we want to be able to provide goals in the form of high-level notions, and expect the agents themselves to determine what behavior is necessary to reach these goals.

Multiagent system is the proper technique for studying immunology because: first, the agent behaviors can straight forwardly incorporate biological facts about low-level components, even if they cannot be expressed mathematically. Second, information from multiple experiments can be combined into a single simulation, to test for consistency across experiments or to identify gaps in our knowledge. Finally, the immune system is a multifaceted biological system with many diverse interacting mechanisms, and many biologically related values that cannot be measured directly.

In our model, in order to accomplish an autonomous and decentralized IPS and self-healing system, three techniques have been integrated namely DCs, T-cells and B-cells mechanisms, which form the intelligent multiagent system. The incorporation of agents in IPS allows the features of candidness, diverse, scale and structures to be

accomplished. The multiagent system for IPS and SH represents the abstract model for engineering framework of AIS.

3.8 Autonomous IPS and SH Abstract Model

Following the conceptual frameworks for AIS, we develop an abstract model for intrusion prevention and self-healing based on a biological system, namely the human immune system. At this stage, we will deduce the abstract model from the observations and probings of the organs and cells of HIS from the previous sections. As mentioned before, in order to achieve the design features and to accomplish the requirements of our new IPS and SH system, the mechanisms of the system are inspired from the mechanisms of the Dendritic cell, T-cell and B-cell. The proposed model consists of three layers of multiagent system combined with self–healing mechanism.

The model that has been developed can be considered as a hybrid model that integrates host-based with network based IPS techniques. The IPS and SH system can be resided to monitor both network traffic and each individual host, but the difference lies in the type of data monitored. As explained in section 2.4.2, the same model is applied to both techniques but using different types of detected input data.

A host-based IPS monitors the characteristics of a single host and the events occurring within that host for suspicious activity. Host-based IPSs provide a variety of security capabilities such as logging of data related to detected events. This data can be used to settle the validity of alerts, to inspect incidents, and to correlate events between the host-based IPS and other logging sources. A network-based IPS monitors network traffic by scrutinizing network fragments or devices and analyzing network, transport, and application protocols to identify suspicious activity. Network-based IPSs typically offer extensive and broad detection capabilities. Most network-based IPSs use a combination of signature-based detection, anomaly-based detection, and stateful protocol analysis methodologies and provide a wide variety of security capabilities [27].

One of the paradigms in autonomous system is multiagent system which, is a self-governing system. The agents will have a training phase of alert settings to produce alerts in principle messages between the agents. These messages can contain the features and information about abnormal behavior at the instance of detection. In traditional IPS, code viewing and editing are the needed customization features, where some IDS and IPS technologies permit administrators to see some or all of the detection-related code. This is usually limited to signatures, but some technologies allow administrators to see additional codes, such as programs used to perform stateful protocol analysis. Viewing the code can help analysts to determine why a particular alert is generated, and is helpful for validating alerts and identifying false positives.

The proposed system is designed in such a way that the information is extracted autonomously by detection and prevention action taken immediately. Once an event is detected, an alert will be sent to an analysis agent in a formulation of agent messages. From the information in the message, the analysis agent has to specify whether the abnormal activities are either a misuse or an anomaly by searching in the knowledge base. All the procedures of getting the information and alerts are performed autonomously between the agents. The self-healing component uses the information analyzed by the IPS component to specify the damage caused by the intrusion. This feature stops the spread of malicious activities and maintains system continuity. The cooperation among mechanisms mapped from HIS are effective in the intrusion detection prevention, and specification of an intrusion route for the network security [93].

In Figure 3.7 the abstract design and model components of IPS and self-healing system are shown. The main agent components of our IPS model are: sense agent (SEA), analysis agent (ANA) and adaptive agent (ADA). Self-healing agent (SHA) is incorporated into the IPS as additional enhanced mechanism for self-healing purposes. One of the central features of the model is that it needs both expert knowledge and training data.

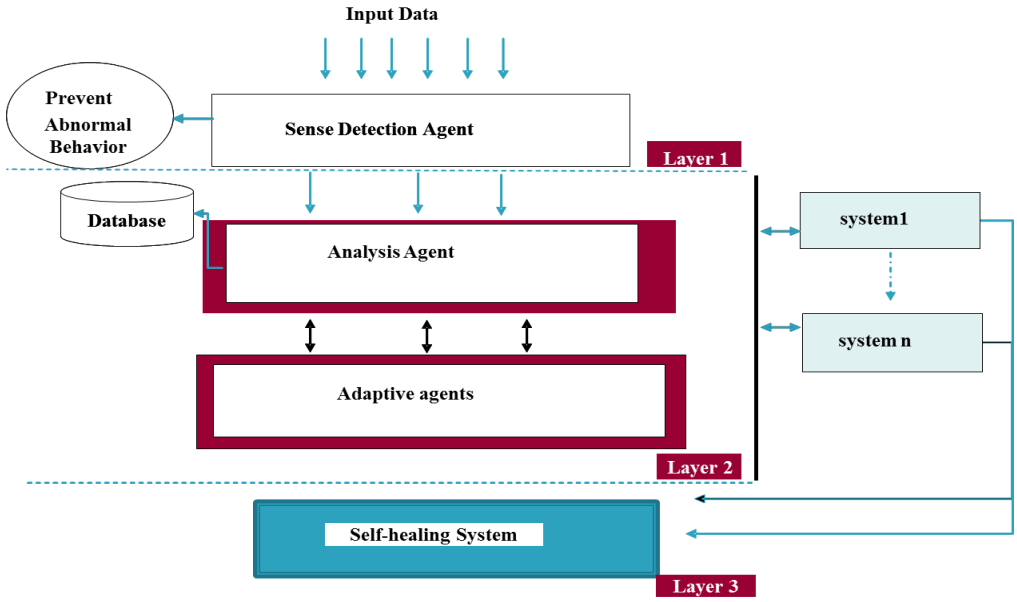


Figure 3.7: IPS and SH Abstract Model

Each agent in the model performs training on multiple types of input data within a specific period. However, with many unsupervised learning systems, training data can be difficult to collect.

The main aim of the model is building an expert knowledge base that assigns input signals and rules to appropriate category. We use two knowledgebase systems, one for misuse attack and the other for self-healing purposes. Each agent must training using the rule-based system to enable to carry out its role. In order to achieve the proactive, self-organizing and autonomous features of the immune system, multiagent paradigm is constructed. The main characteristics of agents are: intelligent, cooperative and autonomy. In an autonomous multiagent system, every agent has its own goals, which drives its decisions. The individual goals of each agent must be specified such that the preferred universal goals of the whole system are achieved [93]. The developed model has been designed to be applicable as a security check in order to prevent malicious attacks such as malware and code-based attacks such as buffer over flow attack initiated by worms.

3.9 New Conceptual Framework of Artificial Immune System

Through our development of IPS and SH abstract model, we have integrated conceptual frameworks in [84] with layered framework for engineering [85] to introduce a new conceptual framework to develop an AIS. Figure 3.8 shows the main steps of this conceptual framework for development of AIS system. First, the domain of application needs to be specified, for example security system, robotic and management information system followed by objectives setting of the system. Next, the requirement to achieve these objectives must be established. This is followed by investigations and observation of a suitable biological system that would contribute towards achieving the objectives.

After the probing and observations, the differences and similarities between the biological and computational systems must be established. These comparisons will be the principles in the construction of the abstract model, which needs to be proven mathematically. A mathematical model can be easily converted into a computational model and algorithm presentation. Finally, the computational model and algorithms must be tested and simulated to get the solution that the AIS developer looks for. The developer of AIS can repeat and recycle the steps from probing to simulating and testing to refine the results until the objectives for the new AIS are accomplished.

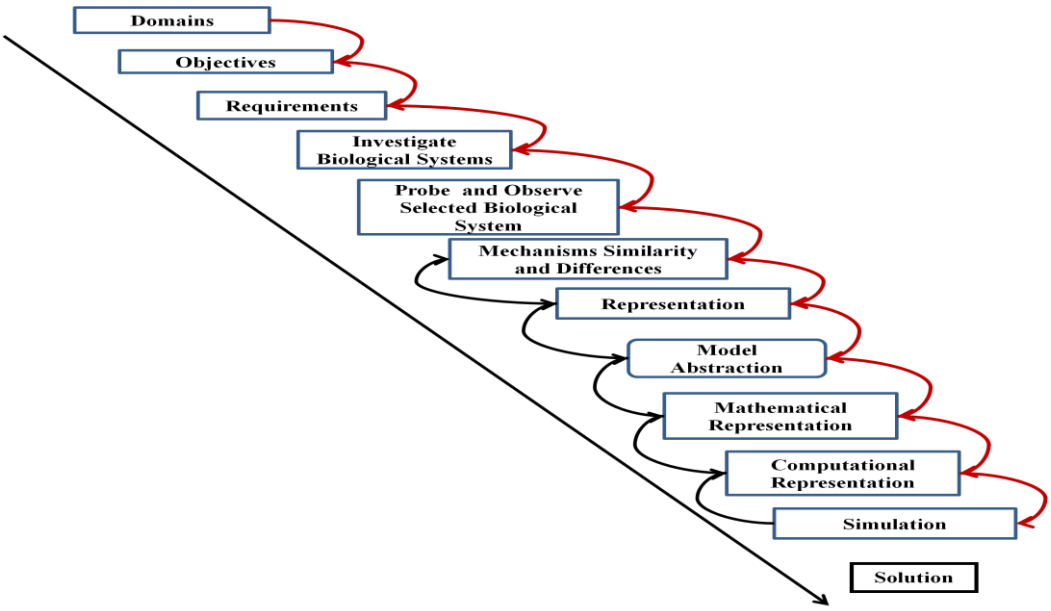


Figure 3.8: A New Conceptual Framework for AIS Development

3.10 Chapter Summary

An abstract model for a new intrusion prevention and self-healing system has been established in this chapter. The abstract model has been developed by making inferences and analyzing two conceptual frameworks developed by previous researches in [84],[85].The mechanisms in the abstract model has been developed by observing suitable mechanisms of HIS and mapping those mechanisms into the IPS and SH system. The features of the IPS and SH system are derived after analyzing and observing the DC, B-cell and T-cell mechanisms, and finding the similarities between these mechanisms and the design features required for the IPS and SH system.

Based on the similarities between the desired features of IPS and HIS, an abstract model consisting of multilayer agent system combined with SH agent has been constructed for the proposed IPS. Each agent has been designed based on the inspirations from specific mechanisms from HIS system. Thus, a new conceptual framework for developing AIS for diverse applications and domains has been introduced.

CHAPTER 4
BIIPSS DESIGN SPECIFICATION

4.1 Chapter Overview

In this chapter, we continue to follow our conceptual framework to establish the design specification for the biological intrusion prevention and self-healing system. In the last chapter, we have presented the abstract model of the intrusion prevention and self-healing system, which consists of four agents. The design and specification of the multiagent system as whole must be constructed to realize the robust IPS and SH system. In addition to that, the design and specification of each agent must be established so that each agent will be able to function as intended.

4.2 The IPS and SH Architecture

The main components of our IPS architecture are: sense detection agent (SEA), analysis agent (ANA) and adaptive agent (ADA). The three agents form three different function layers, each having specific roles, responsibilities, and interactions. In an autonomous multiagent system, every agent has its own goals that drive its decisions. The individual goals of each agent must be specified such that the preferred universal goals of the whole system can be achieved [68], [93]. In the paradigm of agent design, we have to identify the states and transitions of each agent in our architecture according to the design logic. This is an essential step in the designing of proposed IPS and SH system, which is accomplished by creating our own specification language. The derivation of this specification language uses the logic design formulation with set theory notation and Z-notation.

Specification language is mostly suitable to prove the logic, communication and mathematical model for the simulation, and implementation of IPS and SH system as autonomous system.

4.3 IPS and SH Specification Language

The specification language for any system needs: firstly, to apply logic and simple mathematics to computing; secondly, to use formulas i.e. symbols and rules; thirdly, to gain understanding through analysis rather than experiments i.e. testing; and finally, applicable to behavior or specification, structure design or "refinement" implementation. Specification language is most useful in projects that are new, difficult, or critical. Generally, any specification language is not directly executed, instead it describes the system at a much higher level than a programming or execution language because the specification is meant to describe the *what*, not the *how*. A common elementary hypothesis of many specification approaches is that programs are modeled as algebraic structures that include a collection of sets of data values together with functions over those sets. This level of abstraction is adequate with the view that the accuracy of the input/output behavior of a program takes precedence over all its other properties. The IPS and SH specifications must be focused on a process of enhancement before they can actually be implemented. The result of such an enhancement process is an executable algorithm, which is formulated either in a programming language, simulation process, or in an executable subset of the specification language at hand. Most specification languages:

- Explicitly describes behavior in terms of a model using well-defined types (set, sequences, relations, functions) and defines operations by showing effects on the model.
- Specification includes: type-syntax of object being specified, model-underlying structure and invariant properties of modeled object, pre/post conditions and semantics of operations.

In the next sub sections, we create the specification language for each agent. First, we categorize the roles, function and responsibilities for each agent that will enable the agent to attain its goals. This is followed by specifying the notation for each agent. These specifications are analyzed and proved using Petri net in section 4.3. The IPS

and SH specification language is developed by using a combination of set theory and Z-notation.

All notations of the specification language for the whole system are explained in the table of symbol. These specifications are later used for the construction of the IPS and SH system with distributed immune agents that have the abilities of self-learning, expert knowledge, memory, work autonomously and decartelized learning.

4.3.1 Specification of SEnse Agent (SEA)

The main goal of the SEnse agent (SEA) is to prevent the occurrence of abnormal behavior in order to ensure the security of critical services network system. In order to achieve the goal, the SEA must be able to perform the followings:

- Dynamically learns and trains to build a generic knowledge about the whole network system. In the training period, all features and flags (similar to proteins and antigens in HIS) and classes of data, either normal or attack (signal), are defined according to the specific scanning criteria.

- Senses all input to the network system and classifies them according to the dataset that SEA has been trained, and then decide whether it is a source of malicious activities. The mechanism is analogous to the mechanism performed by dendritic and tissue cells to sense or capture the danger signal.

- Prevents malicious activities upon detection of abnormal behavior.

- Sends a detection message to ANA.

From the definition of SEA functions above, we then specify the roles, functions and responsibilities of SEA using the specification language that has been developed earlier as follows:

The set of roles ($\mathbf{R_{SEA}}$) of SEA is:

$\mathbf{R_{SEA}}$= *{sense input data, prevent abnormal behavior}*

The set of function (**F_{SEA}**) of SEA is:

F_{SEA} =*{features and behavior classification, block abnormal behavior}*

The set of responsibilities (**P_{SEA}**) of SEA is:

P_{SEA}= *{detect malicious activities, send DetectionMsg*ANA*}*

Then we formalize these specifications of SEA with our specification language as follows:

Consider H as the set of hosts in a network system defined as:

$H_n ::= \{h_1|h_2|, \ldots\ldots\ldots\ldots, |h_n\}$,

where;

n is the number of host in the network system and h_1, h_2 ...h_n are n hosts.

Next, we create the first SEA agent,

$\forall\, h_i \in H_n \;\; \exists \;\; SEA$,

where;

h_i is host *i* in H and $i \in \mathbb{N}$,

where; $\mathbb{N}$ is the set of natural numbers.

Then we configure SEA to the set of features and behavior (BH) for any host system. The SEA is trained dynamically to familiarize the dynamic change in the behavior of the host system.

$\forall\, h_i \;\; \exists \;\; \{bh: BH | bh \in \; BH_{nb} \vee BH_{ab}\}$,

where;

$BH_{nb} ::= \{BH_f, \forall\, h_i \in H_{n,}\}$,

$BH_f :: function\ of\ the\ normal\ behavior\ in\ each\ host$

$BH_{ab} = \{BH_{f'}, \forall\, h_i \in H_{n,}\}$

$BH_{f'} :: function\ of\ abnorml\ behavior\ at\ this\ instance$

Then in each host,

$$SEA \xrightarrow{Train} BH.$$

Upon receiving input data, SEA senses the data to decide whether it is normal or abnormal behavior.

$$SEA \xrightarrow{sense} DP,$$

where; $DP \Leftrightarrow InputData$

Here, the SEA has to perform one of two events and change the state of the input data (DP) into one of these two states: prevention and stop communication or continue the connection. If the input data at that instance is denoted by DP_i, according to the data attributes, then SEA will classify the Input Data. The definition of the states can be denoted by:

State1: Prevention and block communication.

If the attributes of DP_i is classified as abnormal behavior with function$BH_{f'}$, SEA will prevent DP_i.

$$If\ DP_i \in BH_{f'}\ ,$$

$$SEA \xrightarrow{prevent} DP_i$$

Then SEA blocks the communication (COM).

$$SEA \xrightarrow{block} COM$$

Finally, SEA sends the detection message to ANA.

$$SEA \xrightarrow{Send} DetectionMsg"BH_{f'}" \uparrow ANA$$

State2:: Normal behavior and continue the connection.

If the attributes of DP_i is classified as normal behavior with function BH_f SEA will continue receiving DP_i.

$DP_i \in BH_f$,

The communication will continue receiving the input data.

$SEA \xrightarrow{continue} COM$

By this, the specification of the abstract model of SEnse agent (SEA) is now complete.

4.3.2 Specification of ANalysis Agent (ANA)

The main goal of the ANalysis agent (ANA) is to analyze the DetectionMsg which contains the abnormal behavior function and attributes. It is able to achieve the main goal by performing the following functions:

- Receives DetectionMsg from SEA, and then analyzes the received information to extract any malicious signature.

- Searches for a match of the signature in the misused abnormal behavior database. If the signature matches are cord in the database, then it is labeled as misused abnormal behavior. In such cases, ANA sends MisusehealMsg to SHA which checks the system behavior and if any abnormal activities are detected. Else, ANA will consider the abnormal behavior as an anomaly and sends AnomalyMsg to ADA.

- Updates database records of detected anomalies. ANA waits for RecognitionMsg from ADA, which contains the recognition information of the anomaly, and then updates the database records.

- Sends Anomalyhealmsg to SHA.

Using our specification language, we specify the roles, functions and responsibilities of ANA as follows:

The set of roles ($\mathbf{R_{ANA}}$) of the ANalysis agent ANA is:

$\mathbf{R_{ANA}}$= *{analyze abnormal behavior}*

The set of functions ($\mathbf{F_{ANA}}$) of the ANalysis agent ANA is:

$\mathbf{F_{ANA}}$ = *{distinguish misuse attack from anomaly attack, analyze attack behavior}*

The set of responsibilities ($\mathbf{P_{ANA}}$) of the ANalysis agent ANA is:

$\mathbf{P_{ANA}}$= {receive *DetectionMsg from SEA, send AnomalyMsg to* ADA *agent, receive RecognitionMsg from ADA agent, call self-healing system}*

These formal specifications of ANA using our specification language are as follows:

$\forall\, h_i \in H_n \;\exists\; ANA$

As ANA receives the DetectionMsg from the SEA, ANA starts to analyze the message content and decide whether the abnormal behavior $BH_{f'}$ is misuse or anomaly. ANA has dynamic knowledge base MD of all misuse abnormal behavior signatures. This can be denoted as follows:

$MD ::= \{m_{a1}, \ldots., m_{an}\} \;\; , \; n \in \mathbb{N}$

where; MD is the set of misuse Abnormal behavior signatures database.

As ANA receives *DetectionMsg*, it starts to scan in the MD database and compare if the detection message content has been included in the MD. These two events are denoted as follows:

$ANA \xrightarrow{receive} DectectionMsg"BH_{f'}" \triangleleft\downarrow \text{SEA}$

$ANA \xrightarrow{scan} scan\ "BH_{f'}"\ in\ MD$

As a result of scanning, ANA enters into one of the following states:

State 1:: Misuse behavior.

If ANA identifies the abnormal behavior $BH_{f'}$ as misuse, ANA then stipulates the features of the misuse abnormal behavior and send a MisuehealMsg message to SHA. These events are denoted as follows:

If $BH_{f'}$ is Misuse abnormal behavior exist in MD,

$$BH_{f'} \exists MD,$$

Then ANA stipulates the features

$$ANA \xrightarrow{analyze} analyze\ "BH_{f'}"$$

Finally, ANA sends the MisuehealMsg message to SHA.

$$ANA \xrightarrow{send} MisusehealMsg \uparrow SHA$$

State 2::Anomaly abnormal behavior

If the scanning result shows that the abnormal behavior is not in the knowledge base, then ANA specifies the behavior as anomaly and send AnomalyMsg, which contains the anomaly behavior features, to ADA.

If the abnormal behavior $BH_{f'}$ is an anomaly, i.e. it does not exist in MD,

$$BH_{f'} \nexists MD$$

Then, ANA sends anomaly message "AnomalyMsg" to ADA.

$$ANA \xrightarrow{send} AnomalyMsg \uparrow ADA$$

ANA waits for the recognition message "RecognitionMsg" of the anomaly abnormal behavior $NBH_{f'}$.

$$ANA \xrightarrow{wait} RecognitionMsg : ADA$$

$ANA \xrightarrow{receive} RecognitionMsg"NBH_{f'}" \triangleleft\downarrow$ SEA

As ANA receives the recognition message with abnormal behavior function, it updates the knowledge database MD.

$NBH_{f'}$ :: *new abnormal function*

State 3:: Updates the MD with the anomaly behavior.

ANA analyzes the new abnormal behavior function and sends a healing message "AnomalyhealMsg" to SHA,

$ANA \xrightarrow{analyze} analyze\ "NBH_{f'}"$

ANA denotes the misuse abnormal behavior function.

$MD \otimes NBH_{f'}$

ANA sends the AnomalyhealMsg to SH agent

$ANA \xrightarrow{send} AnomalyhealMsg \uparrow SH$

4.3.3 Specification of Adaptive Agent (ADA)

The Adaptive agent ADA is specified to perform the following functions:

- Receives the AnomalyMsg from ANA and triggers adaptation method to address the anomaly behavior.
- Recognizes and registers the anomaly behavior signature.
- Sends RecognitionMsg to ANA that identifies the anomaly abnormal behavior features and contains information required for knowledge database MD to register the anomaly behavior.

Based on the functions of the ADA defined above, using our specification language we specify the roles, functions and responsibilities of ADA as follow:

The set of roles ($\mathbf{R_{ADA}}$) of the adaptive agent ADA is:

$\mathbf{R_{ADA}}$= *{adaptationToanomaly}*

The set of function ($\mathbf{F_{ADA}}$) of the adaptive agent ADA is:

$\mathbf{F_{ADA}}$= {recognize anomaly}

The set of responsibilities ($\mathbf{P_{ADA}}$) of the adaptive agent ADA is:

$\mathbf{P_{ADA}}$= *{anomalysignature,feedback to analysis agent }*

The specifications of ADA have been formalized using the specification language as follows:

Define ADA in each host,

$$\forall\, h_i \in H_n \;\exists\; ADA$$

As ADA receives AnomalyMsg from ANA, ADA will start the adaptation process on the anomaly.

ADA receives the anomaly message AnomalyhealMsg from ANA,

$$ADA \xrightarrow{receive} AnomalyhealMsg \downarrow SH$$

The features and characteristics of the anomaly abnormal behavior are formulated in set F_C,

Let $F_C :: \{f_c | \forall\; BH_{ab} \exists\, f_{cab} \;\wedge\; f_{cnb}\}$

Where;

f_{cab} is charataristics of the abnormal part.

f_{cnb} is characteristics of the normal part.

ADA carries out the adaptation process to the anomaly function to recognize the anomaly behavior,

$$ADA \xrightarrow{adapt} f_{cab} \oslash f_{cnb}$$

ADA fixes the adaptation process,

$$ADA \xrightarrow{fixadapt} f_{cab}$$

As a result of the adaptation process, ADA recognizes the target and features of the anomaly behavior,

$$ADA \xrightarrow{recognize} BH_{ab} \supset f_{cab}$$

As recognition is completed by ADA, ADA sends RecognitionMsg to ANA,

$$ADA \xrightarrow{send} RecognitionMsg \uparrow ANA$$

4.3.4 Specification of Self-healing Agent (SHA)

The functions of SHA are the followings:

- Receives MisusehealMsg and Anomalyhealmsg about harmful malicious activities from agent ANA.

- Diagnoses the system behavior, captures the fault identification, and extracts anomaly activities configuration.

- SHA is expert knowledge and trained to adapt to abnormal activities using the mechanism inspired by cell regeneration mechanism.

- SHA finds a suitable healing component candidate for each fault that can repair the specific damages caused by the harmful activities. SHA fixes generation of the candidate for testing according to the diagnosis of the damages.

- Finally, SHA performs self-testing for the new regenerated damaged component and deploys it.

Based on the above functions of SHA, next we specify the roles, functions and responsibilities of SHA using our specification language as follows:

The set of roles ($\mathbf{R_{SHA}}$) of the self-healing agent (SHA) is:

$\mathbf{R_{SHA}}$= *{self-healing for abnormal activity damages}*

The set of function ($\mathbf{F_{SHA}}$) of the self-healing agent (SHA) is:

$\mathbf{F_{SHA}}$= *{diagnoses, faultadaptation, testing}*

The set of responsibilities ($\mathbf{P_{SHA}}$) of the self-healing agent (SHA) is:

$\mathbf{P_{SHA}}$= *{receive msg, faultidentification, candidatefixgeneratin, deployment}*

Then, these specifications of SHA are formalized with our specification language as follows:

$$\forall\, h_i \in H_n \;\exists\; SHA$$

As self-healing agent SHA receives *MisusehealMsg* or *Anomalyhealmsg*,

$$SHA \xrightarrow{receive} AnomalyhealMsg \vee MisusehealMsg \downarrow SH$$

SHA diagnoses the fault from the messages,

$$SHA \xrightarrow{extract} f_{cab}$$

Then, SHA analyzes and identifies the damaged function,

$$SHA \xrightarrow{damage} f_{cab_i}\,\{\forall f_{cab}\,\exists\, f_{cab_i} \oplus DMG\}$$

SHA fixes the damage DMG,

$$SHA \xrightarrow{FIX} DMG$$

SHA has to search in Heal Knowledge database (HK) for the Healing Candidate (HC) to fix the damage (DMG).

SHA searches for the heal component,

$$SHA \xrightarrow{HealSearc\ h} HK :: \{\forall DMG \in HK \wedge \exists\, HC \ni HK : DMG \leftrightarrow HC\},$$

SHA selects and fix a heal candidate,

$$SHA \xrightarrow{HealFix} Fix :: DMG \bowtie HC,$$

SHA tests the fixed candidate heal component,

$$SHA \xrightarrow{Test} HC == \neg DMG,$$

Finally, SHA regenerates the tested heal candidate,

$$SHA \xrightarrow{Regenerate} HC \boxplus BH_{nb}$$

4.4. Multiagent System Architecture

Many researchers and developers of autonomic systems have used different architectures and methodologies to establish multiagent system for many types of applications. The construction of autonomic security systems based on multiagent system varies among developers of intrusion detection and prevention systems. Different developer use different requirements, architecture and methods. We can refer to these references to discover the methods and architectures [71], [94], [95].

In the previous section concerning abstract architecture model, we have shown that an agent behaves with respect to changes in its environment. Continuing with the design specification, here we need to establish the design for every agent based on its own environment, and this environment is defined by the nature of the agent. The environment consists of a set of states S. The agent can undertake a set of actions, A and a set of percept, P. The details of these are specified for IPS and self-healing

system. We consider a multiagent system as a discrete event system because we focus on the interactions among the agents and their environments [95].

For each agent, there are specific sets of states (S) for environment, actions (A) and precept (P) which has the behavior represented by the function action:

$P \mapsto A$

And perception function,

$S \mapsto P$

and deterministic behavior of an environment can be represented by the function,

$env: S \times A \mapsto S$

A Petri net base for the abstract model of IPS and SH system is defined as a five-tuples (P,T,A,W,M_0).

Where;

P is a finite set of places.

T is a finite set of transitions.

$A \subseteq (P \times T) \cup (T \times P)$ is a set of arcs.

$W: A \mapsto \{1,2,3,\ldots\}$ is a weight function.

$M_0: P \mapsto \{1,2,3,..\}$ is the initial marking.

According to these formulas, a Petri net for the four agents has been built and represented graphically, and proven against three behavior properties: free of deadlock , boundedness and liveness using linear algebraic. Petri net is a tool for proving and analyzing any synchronize, concurrent, and a synchronize system mathematically. Communication systems can be analyzed and designed used Petri nets. We refer to these references to obtain the final design and analysis of the multiagent system of IPS and SH [95], [96].

A multiagent system can be analyzed to assess system properties by using Petri net model. For example, an inspection of the reachability graph of a Petri net model can

indicate if the model is alive and bounded, while liveness and boundedness properties can be assessed using invariant analysis. In our analysis, we use P-invariants, T-invariants obtained from the incidence matrix, which give information regarding token conservation, and transition firing sequences that leave the marking of the net unchanged. Petri net design model for each agent *i* has been constructed according to the procedures given in [94], [95] as follows:

Definitions:

S_i is defined to be the set of environment states of agent *i.*

Where;

$s_{ij} \in S_i$ be the j^{th} environment state of agent *i.*

The set of actions for agent *i* is defined *to* be A_i.

where;

$a_{ik} \in A_i$ be the k^{th} action of agent i.

Step1

For each component of S_i, place p_{ij} $for\ s_{ij}$ is labeled.

For each component of A_i, action T_{ik} $for\ a_{ij}$ is labeled.

Step2

At the instance, the environment function is defined by:

$S_i \times A_i \mapsto S_i$,

$s_{ij} \times a_{ik} \mapsto S_{il}$,

An arc leaving placeP_{il} and ending T_{ik} is added, followed by an arc leaving T_{ik} and ending in place p_{il} .

Step3

All arcs are labeled with weight $w = 1$ and a token in place represent the initial state of the environment is added.

The combination of all agents in one system is based on their indirect interactions i.e. any agent *i* action will change an environment of state of the agent *j*. Such communication between agents is regarded as a steady event in the final comprehensive model of multiagent system.

To complete the assessment of the system properties: deadlock inspection, liveness and boundendss using Petri net model, an analytical model has to be built for the multiagent system as follows:

For each arc from transition t_i to place p_i, the weight is:

$$a_{ij}^{+} = w(i,j)$$

For each arc from place p_i to, the weight is:

$$a_{ij}^{-} = w(j,i)$$

The incidence matrix A of a Petri net has |T| number of rows and |P| number of columns.

A P-invariant is a vector that satisfies $I_x = 0$ (4.1)

T invariant is a vector that satisfies $I_y^T = 0$ (4.2)

The Petri net model for each agent must prove the following conditions to satisfy the properties: deadlock inspection, liveness and boundendss:

1. For each agent, the Petri net is covered by P-invariant if and only if for each place in the net, there exist a positive P-invariant *x* such that:

$$x(s) > 0 \quad (4.3)$$

and

$$Ix = 0 \quad (4.4)$$

2. Bound structure of Petri net is captured if it is covered by P-invariants and the initial mark M_0 is finite.
3. For each agent the Petri net is covered by T-invariant if and only if for each place in the net, there exist a positive T-invariant *y* such that:

$$y(t) > 0 \quad (4.5)$$

and

$$I^T y = 0 \quad (4.6)$$

4. Petri net is live and bounded if it is covered by T-invariant.

The above conditions have been captured and proven for each agent in the proposed system. The design and analysis of each Petri net agent will be shown in the following sections.

4.4.1 The Design of Sense Agent (SEA) Model

Set of states= *{configure, train, scan, complete prevention}*

S_{SEA}= *{s_1, s_2, s_3, s_4}*

Set of actions= *{configure completed, detect, block, permit, detection message}*

A_{SEA}= *{a_1, a_2, a_3, a_4, a_5}*

Set of percepts=*{training, detection, prevention, communication with analysis agent}*

P_{SEA}= *{p_1, p_2, p_3, p_4}*

The Petri net model of SEA is shown in Figure 4.1.

For SEA each j^{th} environment has the state:

$s_{SEA_j} \in S_{SEA}$

Similarly, A_{SEA} be the set of actions of SEA;

$a_{SEA_k} \in A_{SEA}$

Where, k^{th} are the actions of SEA.

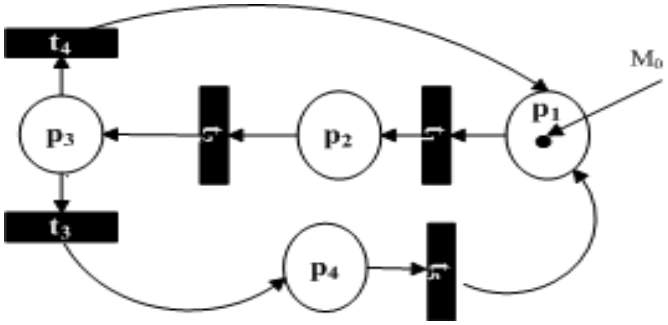


Figure 4.1: Petri net Model of SEA

These definitions have been used to build the Petri net sub–model of SEA. The incidence matrix for SEA is obtained from the Petri net graph, and both *P*-invariant and *T*-invariant satisfy the conditions mentioned above. The T_{SEA} and P_{SEA} invariants for SEA vectors are:

$$I_{SEA} = \begin{bmatrix} & P_1 & P_2 & P_3 & p_4 \\ t_1 & -1 & 1 & 0 & 0 \\ t_2 & 0 & -1 & 1 & 0 \\ t_3 & 0 & 0 & -1 & 1 \\ t_4 & 1 & 0 & -1 & 0 \\ t_5 & 1 & 0 & 0 & -1 \end{bmatrix},$$

$x(s)^T{}_{SEA} = [1\ 1\ 1\ \ 0]$; verify equation 4.5

$y(t)^T{}_{SEA} = [0\ 1\ 0\ 0\ 1]$; verify equation 4.6

4.4.2 The Design of Analysis Agent (ANA) Model

Set of states = *{configure, monitor, analyze, decide, wait, update}*

$S_{ANA} = \{s_1, s_2, s_3, s_4, s_5, s_6\}$

Set of actions= *{configure completed, DetectionMsg, scan, MisusehealMsg, send AnomalyMsg, Receive RecognitionMsg, AnomalyhealMsg, register }*

$A_{ANA} = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8\}$

Set of percepts= *{monitoring, receiving detection, analyzing, decision, updating, triggerheal}*

$P_{ANA} = \{p_1, p_2, p_3, p_4, p_5, p_6\}$

For ANA agent each j^{th} environment has the state:

$s_{ANA_j} \in S_{ANA}$

Similarly, A_{ANA} be the set of actions of ANA*;*

$a_{ANA_k} \in A_{ANA}$

where; k^{th} are the actions of ANA.

We use these definitions to build the Petri net sub–model of ANA. The incidence matrix for ANA is obtained from the Petri net graph, and both *P*-invariant and *T*-invariant satisfy the conditions mentioned above. The T_{ANA} and P_{ANA} invariants for ANA vectors are:

$$I_{ANA} = \begin{bmatrix} & p_1 & p_2 & p_3 & p & p_5 & p_6 \\ t_1 & -1 & 1 & 0 & 0 & 0 & 0 \\ t_2 & 0 & -1 & 1 & 0 & 0 & 0 \\ t_3 & 0 & 0 & -1 & 1 & 1 & 0 \\ t_4 & 0 & 0 & 0 & -1 & 1 & 0 \\ t_5 & 1 & 0 & 0 & -1 & 0 & 0 \\ t_6 & 0 & 0 & 0 & 0 & -1 & 1 \\ t_7 & 1 & 0 & 0 & 0 & 0 & -1 \\ t_8 & 1 & 0 & 0 & 0 & 0 & -1 \end{bmatrix},$$

$x(s)^T{}_{ANA} = [1\ 1\ \ 0\ 10\ 1]$; verify equation 4.5

$y(t)^T{}_{ANA} = [1\ 0\ 0\ 1\ 0\ 1\ 1\ 1\ ]$; verify equation 4.6

The Petri net model of ANA is shown in Figure 4.2.

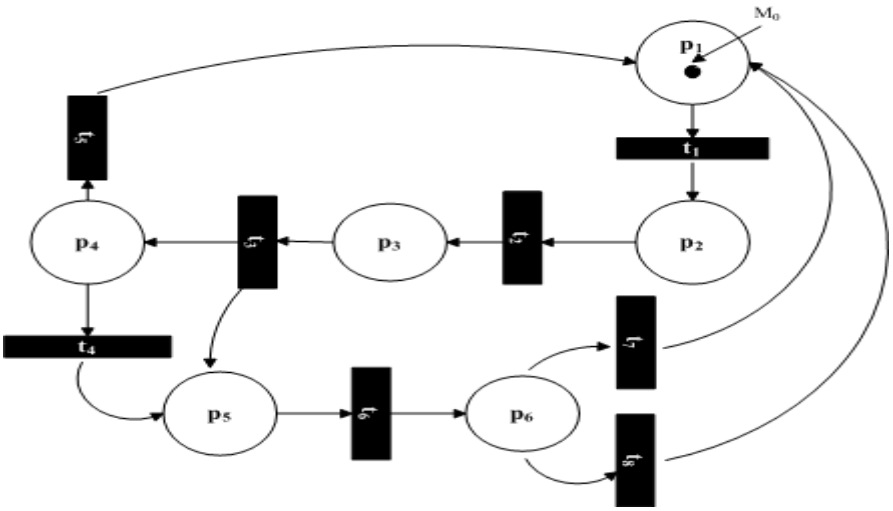


Figure 4.2: Petri net Model of ANA

4.4.3 The Design of Adaptive Agent (ADA) Model

Set of states = *{configure, monitor, adaptation, recognize}*

$S_{ADA} = \{s_1, s_2, s_3, s_4\}$

Set of actions=*{configure completed, received AnomalyMsg, wait, fix adaptation, send AnomalyhealMsg }*

$A_{ADA} = \{a_1, a_2, a_3, a_4, a_5\}$

Set of percepts= *{monitoring, adaptation, recognition, sending}*

$P_{ADA} = \{p_1, p_2, p_3, p_4\}$

For *ADA* agent each j^{th} environment has the state:

$$s_{ADA_j} \in S_{ADA}$$

Similarly A_{ADA} be the set of actions of ADA,

$$a_{ADA_k} \in A_{ADA}$$

where; k^{th} are the actions of ADA.

We use these definitions to build the Petri net sub–model of ADA. The incidence matrix for ADA is obtained from the Petri net graph. Both *P*-invariant and *T*-invariant satisfy the conditions mentioned above. The T_{ADA} and P_{ADA} invariants for ADA vectors are:

$$I_{ADA} = \begin{bmatrix} & p_1 & p_2 & p_3 & p_4 \\ t_1 & -1 & 1 & 0 & 0 \\ t_2 & 1 & -1 & 0 & 0 \\ t_3 & 0 & -1 & 1 & 0 \\ t_4 & 0 & 0 & -1 & 1 \\ t_5 & 1 & 0 & 0 & -1 \end{bmatrix},$$

$x(s)^T{}_{ADA} = [1\ 1\ \ 0\ 1]$; verify equation 4.5

$y(t)^T{}_{ADA} = [0\ 1\ 0\ 1\ 0]$; verify equation 4.6

The Petri net model of ADA is shown in Figure 4.3.

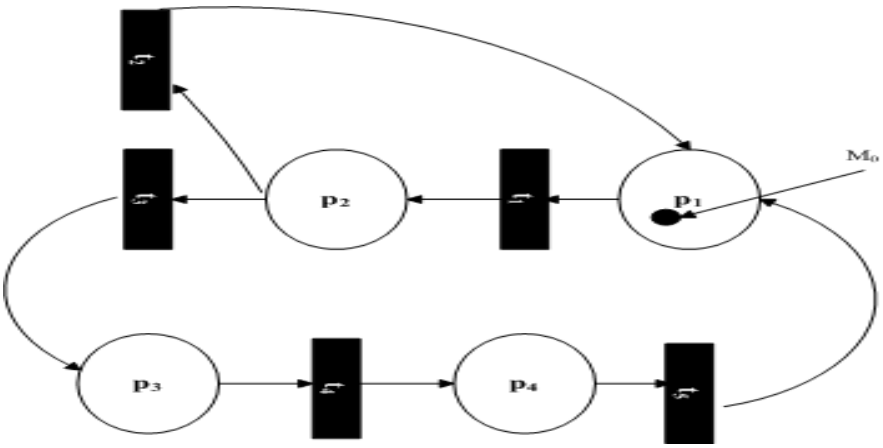


Figure 4.3: Petri net Model of ADA

4.4.4 The Design of Self–healing Agent (SHA*)* Model

Set of states= *{configure, wait, search, fault diagnosis, fault adaptation, self-test}*

S_{SHA}= *{*s_1, s_2, s_3, s_4, s_5, s_6*}*

Set of actions= *{configure complete, received anomaly message, received misuse message, fix daignosis, fault identification, candidate fix generation, deployment}*

A_{SHA}= *{*a_1, a_2, a_3, a_4, a_5, a_6, a_7*}*

Set of precepts= *{training, receiving, fault identification, fault adaptation, testing, deployment}*

P_{SHA}= $\{p_1, p_2, p_3, p_4, p_5, p_6\}$

For SHA agent each j^{th} environment has the state:

$$s_{SHA_j} \in S_{SHA}$$

Similarly, A_{SHA} be the set of actions of SHA*;*

$$a_{SHA_k} \in A_{SHA}$$

where; k^{th} are the actions of SHA.

These definitions have been used to build the Petri net sub–model of the SHA agent. The incidence matrix for SHA is obtained from the Petri net graph, and both *P*-invariant and *T*-invariant satisfy the conditions mentioned above. The T_{SHA} invariant and P_{SHA} invariant for SHA vectors are:

$$I_{SHA} = \begin{bmatrix} & p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \\ t_1 & -1 & 1 & 0 & 0 & 0 & 0 \\ t_2 & 0 & -1 & 1 & 0 & 0 & 0 \\ t_3 & 0 & -1 & 1 & 0 & 0 & 0 \\ t_4 & 0 & 0 & -1 & 1 & 0 & 0 \\ t & 0 & 0 & 0 & -1 & 1 & 0 \\ t_6 & 0 & 0 & 0 & 0 & -1 & 1 \\ t_7 & 1 & 0 & 0 & 0 & 0 & -1 \end{bmatrix},$$

$x(s)^T{}_{SHA} = [0\ 1\ 1\ 1\ 00]$; verify equation 4.5

$y(t)^T{}_{SHA} = [1\ 1\ 0\ 1\ 1\ 1\ 1]$; verify equation 4.6

The Petri net model of SHA is shown in Figure 4.4.

The design for each agent has been established. We look for a multiagent system where the agents communicate with each other autonomously. In the next section, we will present the design of this communication and interaction using the foundation for intelligent physical agents, FIPA.

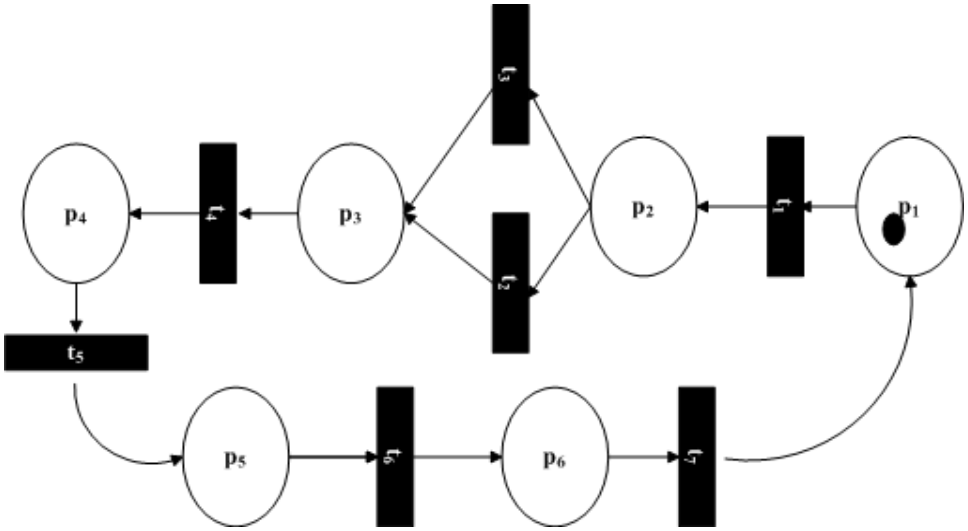


Figure 4.4: Petri net Model of SHA

4.5 Immune Agents Communication and Interaction

The Foundation for Intelligent Physical Agents (FIPA) is an international organization that is enthusiastically promoting the development of intelligent agents by explicitly developing specifications that support interoperability among agents and agent based applications. FIPA semantic agent communication language is required for concrete specifications of agents.

FIPA semantic language is used as the strong basis for developing multiagent system models, particularly for models of different technologies, representation of models, and programming models more specifically for security systems. It may also be essential for concrete specifications, including implementers of agent platforms, agent systems, and gateways between agent systems [97].This standard language of establishing communication is used to build the interactions and communication between the agents in our proposed biological inspired IPS and SH system. Figure 4.5 shows the details of the communication model based on FIPA SL language.

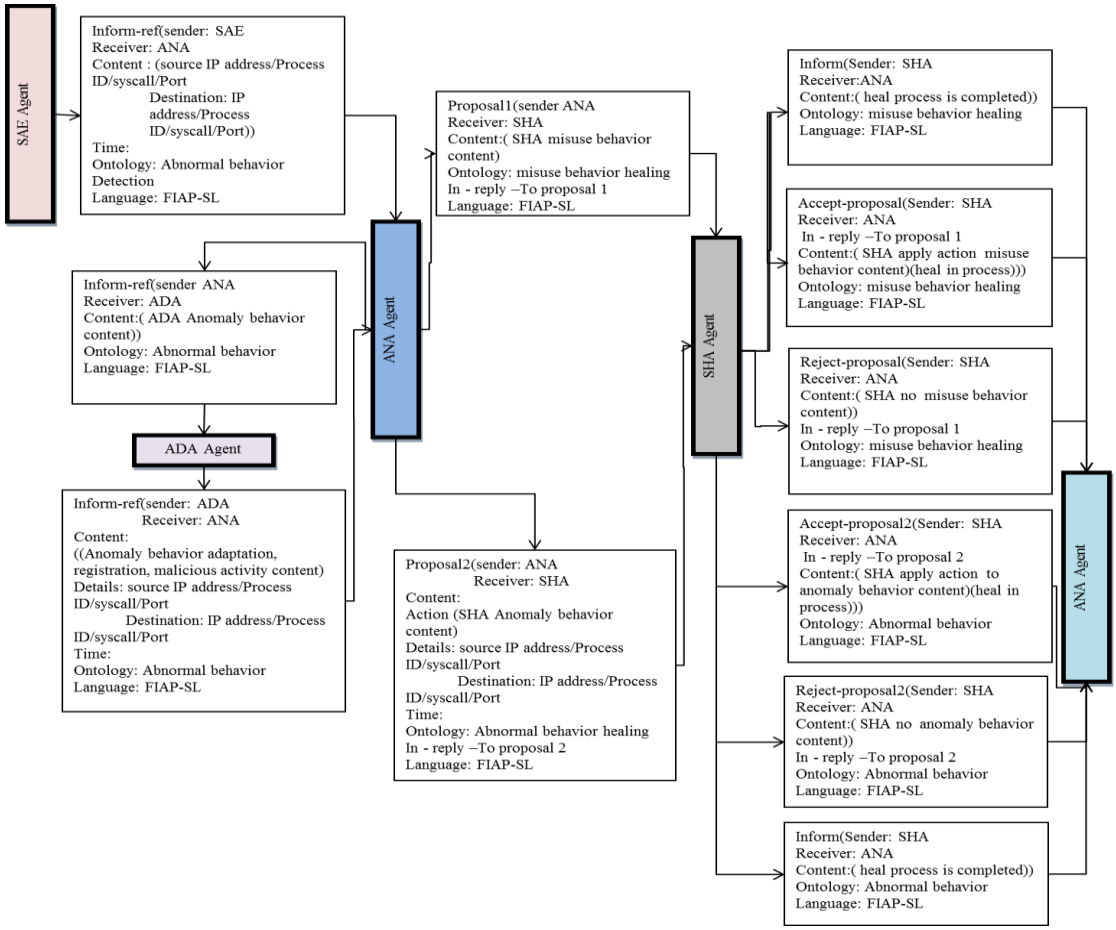


Figure 4.5: The Interactions and Communication between Agents

4.6 Chapter Summary

The IPS and SH system have been designed as a multiagent system to accomplish the first objective of this research i.e. an autonomous IPS and SH system. As the principal step in designing the new systems, a new specification language for the IPS and SH agents is constructed the roles, functions, responsibilities, events and states of each agent have been specified and analyzed using set theory and logic design. The analytical and design models have been established using Petri net for each agent in the system. The required properties of the agent design have been proven. Finally, FIPA semantic language has been used to explore the communication between agents in order to construct the communication between them. In the next chapter, the mathematical models of IPS and SH will be established, which is an important step for validation of the system.

CHAPTER 5

BIIPSS MATHEMATICAL AND COMPUTATIONAL MODEL

5.1 Chapter Overview

Danger Theory (DT) integrated with negative selection provides a considerable shift in perspective about the main objectives of human immune system (HIS). This perspective though controversial among immunologists, may enable artificial immune system (AIS) researchers to extract benefits of the theory. To justify the potentials of DT for AIS and within the perspective of the conceptual framework of the proposed AIS system introduced in chapter 3, computation modeling plays an important role in establishing the aspects of the biological intrusion prevention and self-healing system.

In this chapter, we present the computational model of IPS and self-healing system. This computational model accomplishes the abstract specification and design criteria that have been set in the previous chapters i.e. chapters 3 and 4. For each agent, we propose an algorithm that should be implemented in order to achieve the design features that we look for. The integration of DT in AIS system as proposed in this work will give us a significant advantage over AIS based on negative selection only. This chapter presents expressive details of the most important stage of the framework. The details given in different sections are structured to gradually follow the abstract model. Each of the subsequent sections describes different algorithms that have been built upon the basic mechanisms of HIS, relevant theoretical background, and logical mapping that have been derived in chapter 4.

5.2 Computational Model of IPS and SH

The computational model of IPS and SH are established based on the four mechanisms as shown Figure 5.1. These algorithms are designed to realize the four

main goals of our system: detection of abnormal behavior, prevention of abnormal behavior, adaptation to abnormal behavior, and finally healing of any damages caused by intrusion. The subsequent sections explain the four algorithms.

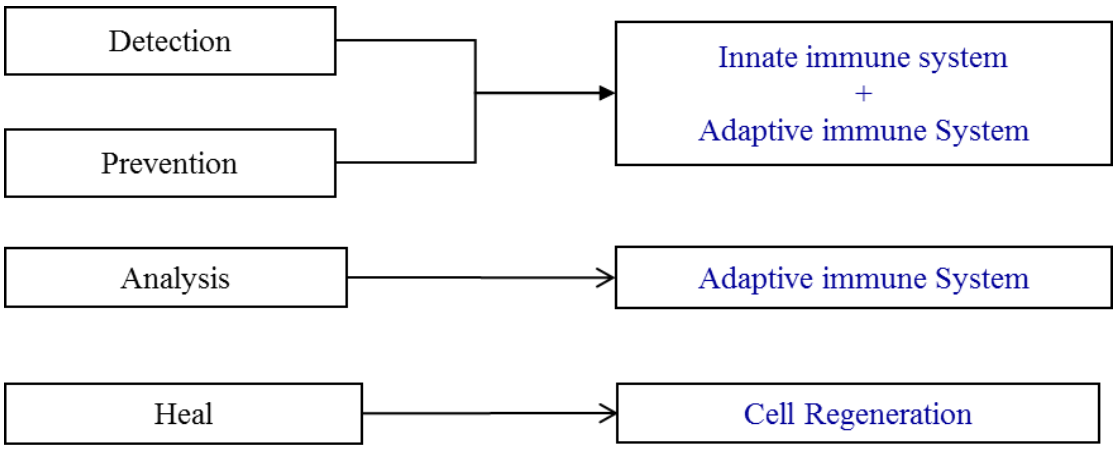


Figure 5.1: The Main Algorithms of IPS and SH System

5.2.1 Detection Algorithm

The Detection Algorithm (DA) is mapped from the mechanism of DC and represented by SEA, which applies three algorithms to classify the input data in the network system as either normal or abnormal behaviors. Julie et al. [15] introduced a DCA as mentioned in the literature review and Thomas at el. [17] has discussed the geometrical insight of DCA. They explained that in spite of the good result obtained by this algorithm in intrusion detection application; there are several limitations of the DCA. They found that the DCA is a collection of linear classifiers, which pose severe limitation on the datasets assessable by the algorithm. This is made worse by the fact that the gradients of the linear boundaries are constant.

However, it is possible to improve the accuracy of intrusion detection algorithm by using non-linear classification methods. The authors explored how the DCA was applied using derivation of linear classification and how that affected the accuracy of the algorithm. From this point of view, we consider the DC as a classifier, and hence we would be able to use an algorithm that applies non-linear classification for significant accuracy of detection.

The limitation of the DCA has been explored by T. Stibor et al. [17] and Robert O. et al. [91]. They discovered that the DCA model has three main limitations: firstly, the model assumes that the middle co-stimulatory fragment signal is steady i.e linear; secondly, it is only likely to make predictions for a single cell and the model only takes into account the signal processing element of the DCA. Finally, attempts to explore the antigen-presenting phase have never been made before.

Feng Gu et al. [95] discussed the integration of Real-Time analysis with DCA. They proposed that ideally, the intrusion detection should be performed in real-time to continuously detect misused behaviors as soon as they occur since the analysis process of DCA is performed offline. The authors proposed a real-time analysis component to be integrated in the DCA to improve validation.

Their original step for improvement is to implement segmentation to the DCA. Two segmentation approaches were introduced and tested, namely antigen based segmentation (ABS) and time based segmentation (TBS). The outcomes of the experiments suggested that implementing segmentation produced different and considerably better outcome, when compared to the standard DCA without segmentation. Based on their study, we conclude that segmentation is applicable to DCA for the purpose of real-time analysis. From these different points of view and evaluation of DCA, we are presenting a new algorithm inspired from DC mechanism to overcome the deficiency of DCA in order to achieve a more accurate and high performance detection.

Firstly, in our research we present a new non-classification algorithm to intrusion detection by introducing SEnse agent (SEA). The newly developed algorithm uses a non-linear classification method to classify and detect misused and anomaly abnormal behaviors. Secondly, we need to prevail over the existing limitations of DCA and implement our detection algorithm in real time without the need of segmentation process.

Our approach takes the advantages of Cluster-k-Nearest-Neighbor, k-means and Gaussian mixture methods, which are able to give highly accurate and fast classifier detection system that requires less training data. Generally, classification systems investigate and categorize the data into known classes; in our case, we have two

classes representing normal behavior or abnormal behavior. Each class of data is labeled with a known class label. Classification techniques are functional for a wide variety of real time applications dealing with large amount of data [26], [99]. Some of the applications are: Network intrusion detection, Insurance/Credit card fraud detection, Industrial Damage Detection...etc. A review of some classification algorithms for intrusion detection is given is the next subsection.

5.2.2 Classification Algorithms in Intrusion Detection

Classification technique is defined as a training technique i.e. the technique of classifying a group or set of labeled data instances known as training instances and then, classifying a test instance into one of the classes using the learnt training model testing data [100]-[102]. Classification based intrusion detection techniques operate in a similar two-phase fashion. The training phase learns a classifier using the available labeled training data, while the testing phase classifies a test instance as normal or anomalous using the classifier. In DC, antigens are used to differentiate between danger and safe signals, but in our case, we use features of the labeled data used during the training phase to classify normal and abnormal behaviors. Similarly, protein transformation in DC is mapped to classification of testing data during the testing phase.

Classification based intrusion detection techniques operate under the general assumption that a classifier can distinguish between normal and anomalous classes from the learnt feature i.e. similar to antigens stimulation in DC. Based on the labels available for training phase, classification based anomaly detection techniques can be grouped into two broad types: multiclass and one-class anomaly detection techniques. Multiclass anomaly detection techniques assume that the training data contain labeled instances belonging to multiple normal classes Stefano et al. [103]. Such anomaly detection techniques learn a classifier to distinguish each normal class from the rest of the classes. A test instance is identified as anomalous if it is not classified as normal by any of the classifiers. Some techniques in this sub-category correlate to confidence of the prediction made by the classifier Barbara et al. [104]-[106]. If none of the

classifiers is confident in classifying the test instance as normal, the instance is declared as anomalous.

One-class anomaly detection techniques suppose that all training instances have only one class label. Such techniques learn a discriminative boundary around the normal instances using a one-class classification algorithm, e.g. one-class SVMs, one-class Kernel Fisher Discriminates [107], [108]. Any test instance that does not fall within the learnt boundary is declared as anomalous. There are varieties of intrusion detection techniques that use different classification algorithms to build classifiers.

In [109], [110] are some examples of classification based anomaly detection technique using neural network technique. Anomaly detection technique using neural networks for multiclass generally works in two stages. First, a neural network is trained on the normal training data to learn the different normal classes. Second, each test instance is provided as an input to the neural network. If the network accepts the test input, it is normal and if the network rejects a test input, it is an anomaly.

Another technique for anomaly detection is Bayesian Network based, which is applied also for multiclass anomaly detection. Bayesian networks estimate the subsequent probability of monitoring a class label, given a test data instance. The class label with largest subsequent is selected as the expected class for the given test instance. The likelihood of monitoring the test instance for a given class, and the class probabilities are estimated from the training dataset. The basic technique can be generalized to each test instance, and a class label can be assigned to the test instance using the cumulative value. Several alternatives for the basic technique have been proposed for network intrusion detection [111]-[113]. Anomaly detection for one class is applied by using Support Vector Machines. Such techniques use one class learning techniques that learn a region containing the training data instances. Kernels, such as radial basis function (RBF) kernel, can be used to learn complex regions. For each test instance, the basic technique determines if the test instance falls within the learnt region. If a test instance falls within the learnt region, it is declared as normal, else it is declared as anomalous. Several system developers have used this technique [114]-[116], where novelty intrusion detection systems have been accomplished.

Rule based anomaly detection is another classification technique that learns rules that hold the normal behavior of a system. In such a system, a behavior is regarded as an anomaly when a test instance is not covered by any such rule. Rule based techniques are different from other techniques since they have been applied in multiclass as well as one-class setting. Associations of rule mining based techniques have been used for network intrusion detection as in [117]-[120], and for system call intrusion detection [121], [122].

In this work, the training data are clustered into subclasses where each subclass is represented by one data, which contribute to a reduction of the classification time. We get the benefits of the Nearest- Neighbor (NN) classification algorithm to classify by using representative data. By using this classification method, we are able to answer the challenges that most developers have attempted before, which are:

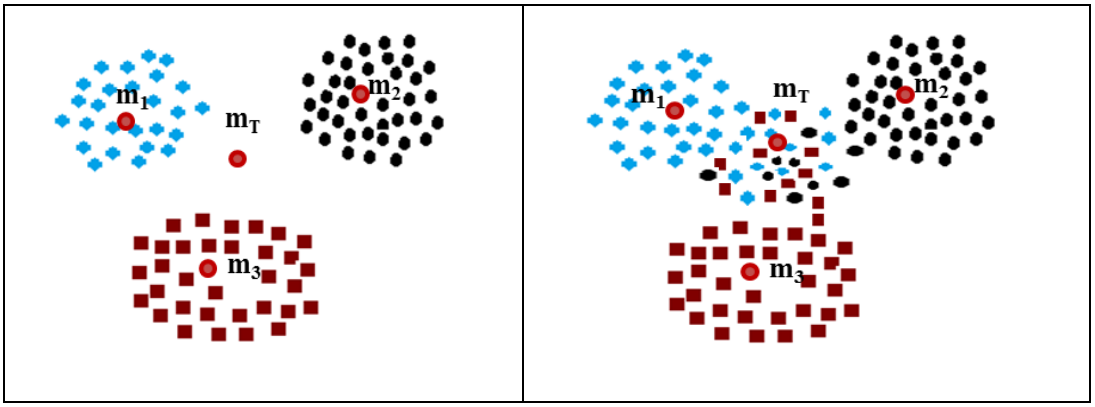
- How many outliers are there in the data?
- Fast and accurate real-time detection.
- Misclassification cost is very high.

This technique can be applied to multiclass as well as one-class classification.

5.2.3 Extraction of the Best Features

Before starting the classification algorithm and clustering of data, an algorithm to extract the best and powerful features for the best classification was implemented. The extraction algorithm reduces classification time and training time, which increase the accuracy of classification and make strong discrimination or classification of the data. This algorithm follows a method that depends on the mean and variance of each class in the training phase.

Suppose m_1, m_2, and m_3 are the mean of class1, class2, and class3 respectively, and m_T is the total mean of all classes; then we obtain two metrics as shown in Figure 5.2(a).



(a)Three classes without overlapping (b)Three classes with overlapping

Figure 5.2: Classes Overlapping

Figure 5.2 (a) shows that the coefficients of the best features extracted from the classes are good for classification process that will lead to increase system's accuracy. On the other hand, Figure 5.2 (b) is showing that the coefficients extracted from the classes are not good for classification process due to overlapping classes that will lead to increase the probability of error. To extract the best features, we need to calculate the total mean m_T as in equation (5.1).

$$m_T = \frac{\sum_{i=1}^{n} m_i}{n} \qquad (5.1)$$

Where ; m_T is total mean, m_i is class mean, i is index of class, and n is the size of class "i".

So the variance of m_i is:

$$var_i = \frac{\sum_{j=1}^{n} (x_{ij} - m_T)^2}{n} \qquad (5.2)$$

Where x_{ij} belongs to class'i'.

The metric obtained from Figure 5.2 (b) is not efficient for extracting the features of intrusion detection dataset due to overlapping classes. Therefore, another metric namely Var_{mod} has been is used.

$$Var_{mod} = \frac{1}{n}\left(\sum_i \frac{(m_i - m_T)^2}{var_i}\right) \qquad (5.3)$$

or

$$Var_{min} = \min_i \left\{\sum_i \frac{(m_i - m_T)^2}{var_i}\right\} \qquad (5.4)$$

Where; var_i is variance of the class i.

Now the way to select coefficients of the desired features will be as follows:

If $Var_{mod} \leq 1$, we delete all coefficients belonging to this column, and the coefficients will not be considered; otherwise they will be kept. The details of calculating the features extraction are explained in Appendix B.

5.2.4Classification Analysis

Classification techniques that use k-nearest neighbor (k-NN) or Gaussian Mixture Model (GMM) almost have the common sense that they believe the neighboring data. These data are represented as the new pixel vectors, where any new pixel vector for example x will be classified to neighboring k-cluster class and the classification will be more accurate compared to NN technique [123], [124]. This is because they are more competent in overlapping area as these methods take more consideration of training data samples that are less numerous.

In order to reduce the classification time of k-NN technique, we need to cluster our space i.e. organizing the training data into subclasses, where each subclass will be represented by one datum. According to the number of subclasses, we can select two or more representatives. This is followed by applying the classification algorithm NN or k-NN using representative data. The data in the subclasses are random, where they are relatively close to each other. This procedure of classification is known as cluster-k-NN (C-k-NN), which is comparable to 'variable k'-NN.

The estimation the classification times for NN and k-NN respectively are:

$$X = O(N), \qquad \frac{X}{N} \to K \; as \; N \to \infty$$

$$x = o(N), \qquad \frac{x}{N} \to 0 \; as \; N \to \infty$$

where; N is the training data size.

The time of classification is subclass number m_i dependent, where m_i is the number of subclasses in class C_i. Therefore, the classification time is reduced by clustering the space. Generally, m_i is a small number that does not depend on the training data size. This has been considered in the number of Gaussian functions to estimate a

probability density. The estimation of probability density according to GMM method, in general, is bounded with respect to the variable *N*.

Non-parametric density is commonly estimated by k-NN. The rule used by k-NN technique is influential and able to generate highly nonlinear classification even with limited data [117]. To classify a pattern *x*, first we have to find the closed k examples in the dataset, and select the predominant class C_i among those k-neighbors. Problems arise if there are two or more predominant classes. One drawback of k-NN is that the training data must be stored, and a large amount or processing power required for evaluating the density of a new input pattern. However, C-k-NN correct those drawback points

The original C-k-NN classifier is based on the Euclidean distance between a test sample *x* and specified training samples, but in the new C-k-NN we add the following metric in order to get a better estimation of density probability:

$$d(x, x_{ij}) = \frac{d^{s}_{euclidean}\ (x,\hat{x}_{i,j})}{n_{i,j}}, \forall_x \in R^d \qquad (5.5)$$

where;

s is a positive number, we note $C_{i,j}$,for all *s*,

$x_{i,j}$ belongs to the subclass j of class *i*,

$\hat{x}_{i,j}$ is the representative of$C_{i,j}$,

$n_{i,j} = card(C_{i,j})$or the variance of the set $C_{i,j}$.

Figure 5.3 explains how we can distribute the data and find the nearest neighbors by calculating the distance as in equation (5.6). In addition, Figure 5.4 shows how the data is clustered by finding the nearest neighbors.

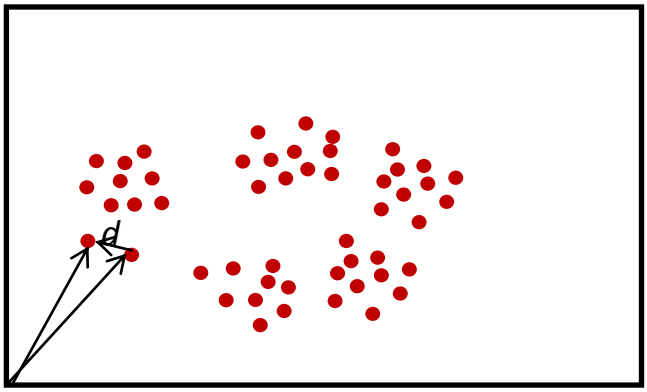


Figure 5.3:Distribution of The Data and Find The Neighbor Using The Euclidean Distances.

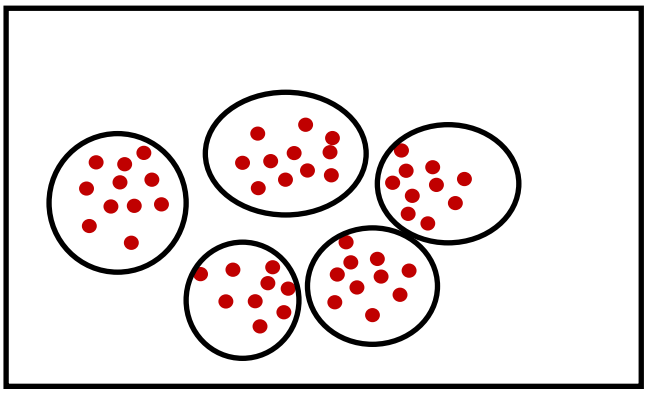


Figure 5.4:Cluster The Data After Estimates The Nearest Neighbors.

The Gaussian mixture model is used as a parametric method that is classified as a semi-parametric density estimation method since it defines a universal class of functional forms for the density model. In a mixture model, the probability density function is expressed as a linear combination of basic functions [126]. A model with M components is explained as a distribution mixture according to equation (5.6).

$$P(x) = \sum_{j=1}^{M} P(j)P(x \,/\, j), \qquad (5.6)$$

Where; $P(j)$ are the mixing coefficients and $P(x/j)$ are the component density functions each mixture component is defined by a Gaussian parametric distribution in d dimensional space.

$$P\left(x/_j\right) = \frac{\exp\left\{-\frac{1}{2}\left(x-\mu_j\right)^T \Sigma_j^{-1}\left(x-\mu_j\right)\right\}}{(2\pi)^{\frac{d}{2}} |\sum_j|^{\frac{1}{2}}} \quad (5.7)$$

The parameters to be estimated are the mixing coefficient $P(j)$, covariance matrix and mean vector μ_j.

In C-k-NN, each Gaussian function $P(j)\ P\left(x/_j\right)$ can be approximated by:

$$\frac{1}{cst + d(x,\mu_j)} \quad (5.8)$$

where; *cst* is any small number that is added to avoid the division by *0*.

The estimation of the number of subclasses and their representatives for C-k-NN (or the number of Gaussian function, M and their means μ_j for GMM) can be derived by k-means cluster. The number of subclasses is needed as input to the k-means cluster algorithm. To fix the number of clusters, we iterate the number of clusters starting with one and under the following conditions it stops:

a. All the representative centroids $(\mu_{i,j})$ have to be closer to their class $C_{i,j}$ than to other classes. This is to reduce misclassification.(i.e. no error in the classification of the training data)

b. The variance of each classvar, does not reduce significantly in comparison to previous iteration. We define the variance of each class as:

$$var = \sum_{i=1}^{n} var_i \quad (5.9)$$

Where; var_i is the variance of subclass i.

Considering that:

$$\frac{\Delta var}{var} \leq \alpha, \qquad 0.0 \leq \alpha \leq 1.0 \quad (5.10)$$

The criteria for obtaining smooth function of var with the number of subclasses are variable. The best value of $\propto$ will be considered in the simulation to obtain the best accuracy. By this judgment of $\propto$, we have completed our dictionary for the training phase. The relation between the variance and k is shown in Figure 5.5.

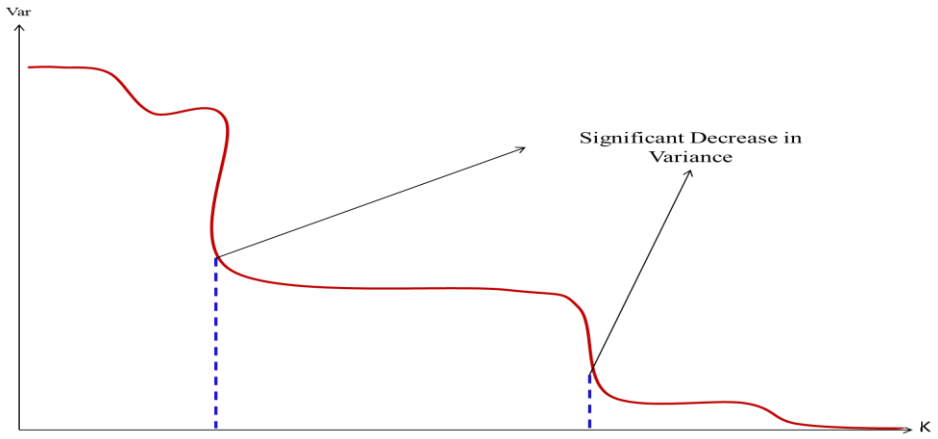


Figure 5.5: Variance in Terms of Number of Cluster "k"

For each class, C_i is represented by:

$$\{\mu_{i,1}, \mu_{i,2}, \ldots\ldots, \mu_{im_i}\}, \qquad 1 \leq i \leq cn \tag{5.11}$$

Where; m_i is the number of subclasses for class C_i and cn is the number of classes.

To classify a new pattern *x*, we use k-NN algorithm on the dataset:

$$\{\mu_{i,j} : 1 \leq i \leq cn, 1 \leq j \leq m_i\} \tag{5.12}$$

By using this technique, we reflect on the minimum rule and assign *x* to class C_i, which is verified by:

$$C_{\hat{\imath}} = arg\left(\min_{\substack{1 \leq i \leq cn \\ 1 \leq j \leq m_i}} \{d(x, m_{i,j})\}\right) \tag{5.13}$$

where;

$$arg(d(x, m_{i,j}) = C_i, \qquad \forall\ 1 \leq cn.$$

However, the k-means cluster is unstable because the result is dependent on random choice of k initial vectors. Therefore, to achieve the stability of k-means cluster, we introduce a different initialization for the algorithm, which gives a better result than the classic k-means cluster where:

$$Var_{modified\ \ algorithm} \leq Var_{classic\ \ algorithm} \tag{5.14}$$

In the modified algorithm, the significant decrease in the value of Var increases the accuracy of the classification.

5.2.4.1 Classification Method

Each class C_i must be clustered to several subclasses and each subclass will be represented by its means $\mu_{i,j}$.

$$C_{i,j} \text{ with } 1 \leq j \leq m_i \qquad (5.15)$$

Thus, each cluster seeks to identify a set of groups, which is minimized within the group variation and maximized between group variation.

In order to cluster each class, the k-means cluster is applied. This is followed by estimating the number of subclasses for each class and the initial k-vectors to initialize the k-means cluster algorithm. To locate the best suitable number of subclasses, the iteration is started with 1 and two conditions are specified to stop the iteration. These conditions are:

a. All representatives $\mu_{i,j}$ should be closewith respect to the metric d in their class C_i, i.e. if we classify all the representatives $\mu_{i,j}$,we will find 100% accuracy. For any misclassifications of $\mu_{i,j}$, we have to decrease the parameter α by multiply it by the factor ά which is less than 1.

b. The variance var_i of each class C_i, does not decrease considerably in comparison to the previous iteration.

$\frac{\Delta var}{var} \leq \alpha$ is used as a criterium to check : if there is a decrease or if it is still approximately constant. In certain cases, it is better to stop the iteration if the condition $\frac{\Delta var}{var} \leq \alpha$ has been checked twice or more, i.e. after the variance has been smooth.

For the initialization of k-means cluster algorithm, in general, the aleatory k-vector that fits our class of data is selected. This makes the algorithm unstable in the sense of final variance,

$$var_{C_i} = \sum_{j=1}^{m_i} var_{C_{i,j}} \quad (5.16)$$

which depends on the initial vector.

At this point a logical question is raised, "How to choose the initial vector in order to find a minimal variance?" To answer this question, two algorithms are presented: near-to-near and near-to-mean, which add significant modifications to the related current application. The next section explores the details of these algorithms.

5.2.4.2 Near-to-near algorithm

The algorithm first calculates the distance $d(x_{i,n}, x_{i,m})$ between test samples x_i for all $x_i \in C_i$ then starts to cluster or class the samples into $(N_i - 1)$ subclasses;
where;

$$card(C_i) = N_i \quad (5.17)$$

Then we put the two closest data into the same subclass.

$$C_i = \{x_{i,n_0}, x_{i,m_0}\} \quad (5.18)$$

where;

$$\min_{n \neq m}\{d(x_{i,n}, x_{i,m})\} = d(x_{i,n_0}, x_{i,m_0}),$$

The next step is to put other data into separate subclasses,

$$C_{i,j} = \{x_{i,j}\}, \forall j \in \{1, \dots N_i\} - \{n_0, m_0\} \quad (5.19)$$

The following index n_i and m_i are considered for which

$$\min_{\substack{n \neq m \\ (n,m) \neq (n_0, m_0)}} \{d(x_{i,n}, x_{i,m})\} = d(x_{i,n_1}, x_{i,m_1}) \quad (5.20)$$

If x_{i,n_1} and x_{i,m_1} belong to the same subclass $C_{i,r}$, then this subclass is split into two other subclasses,

$$C_{i,r+1} = C_{i,r} - \{x_{i,n_1}, x_{i,m_1}\} \quad (5.21)$$

$$C_{i,r} = \{x_{i,n_1}, x_{i,m_1}\} \quad (5.22)$$

If x_{i,n_1} and x_{i,m_1} belong to two different subclasses C_{i,r_1} and C_{i,r_2} respectively. Then we put x_{i,n_1} in subclass C_{i,r_2}, if $card(C_{i,r_2}) > card(C_{i,r_1})$.

And

if $card(C_{i,r_2}) \leq card(C_{i,r_1})$, then x_{i,m_1} is put in (C_{i,r_1}).

To use the cardinally of set, the distance between the vector to the set is used as *d(vector ,mean of set).*

When k-subclass is obtained, the iteration stops and the initial k-vector will be the mean value of each class.

5.2.4.3 Near-to-mean Algorithm

This algorithm is almost the same as near-to-near algorithm but it deals with the mean of subclass $C_{i,r}$.

At the start, the class is split into two subclasses:

$$C_{i,1} = \{x_{i,n_0}, x_{i,m_0}\} \quad (5.23)$$

$$C_{i,2} = \{x_{i,j} \mid j \notin \{n_0, m_0\}\} \quad (5.24)$$

where;

$$d(x_{i,n_0}, x_{i,m_0}) = \min_{n \neq m}\{d(x_{i,n}, x_{i,m})\} \quad (5.25)$$

C_i is updated by replacing x_{i,n_0} and x_{i,m_0} by their average, i.e.

$$C_i^1 = \{.., x_{i,n_{0-1}}, s_0, x_{i,n_{0+1}}, \dots, x_{i,m_{0-1}}, s_0, x_{i,m_{0+1}}, \dots\}, \quad (5.26)$$

where; $S_0 = {(x_{i,n_0}, x_{i,m_0})}/{2}$

Next x_{i,n_1} and x_{i,m_1} are considered such as,

$d(x_{i,n_1}, x_{i,m_1}) = \min_{n \neq m}\{d(x_{i,n}, x_{i,m}) | d(x_{i,n}, x_{i,m}) \neq 0\}$,

We replace all data in C_i^1 that are equal to x_{i,n_1} or x_{i,m_1} by s_1 which is the mean of the union of the two subclasses where x_{i,n_1} and x_{i,m_1} belong to

$$s_1 = \frac{C_{n_1} x_{i,n_1} + C_{m_1} x_{i,m_1}}{C_{n_1} + C_{m_1}}, \qquad (5.27)$$

where; C_{n_1} is the number of repetition of x_{i,n_1} inside C_i^1 and C_{m_1} is the number of repetition of x_{i,m_1} inside C_i^1 as illustrated in Figure 5.6 where Each cluster has a centroid, the mean between each to nearest is estimated, and each two nearest data are replaced by their means.

The algorithm stops once the number of distinct vector inside C_i^r is equal to k.

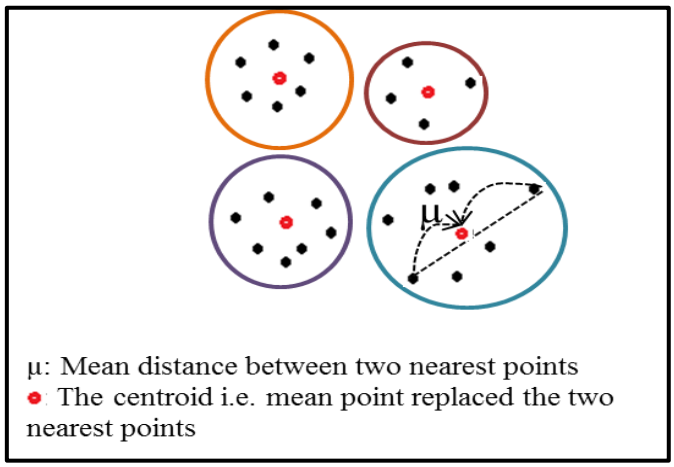


Figure 5.6: **Near to Mean Algorithm**

These classification algorithms do not require all data to be kept; this one of the strong features of these classification algorithms. Instead, they need only the average of each subclass. To classify a new data or vector *x*, we use k-NN algorithm i.e. we assign *x* to class $C_{\hat{\imath}}$ for which,

$$\hat{\imath} = \underset{i}{arg}\min_{i,j} d(x, \mu_{i,j}), \qquad (5.28)$$

where; $\underset{i}{arg} d(x, \mu_{i_0,j_0}) = i_0$.

More examination of the k-NN algorithm is required to find the closest *j*-examples in the dataset and to select the predominant class. The smallest and closest examples in the dataset can be captured and the predominant class that has exact k examples can be selected. The result of the classification is shown in Figure 5.7 where each class cluster contains subclasses represented by the mean of the data in the subclass and one centroid.

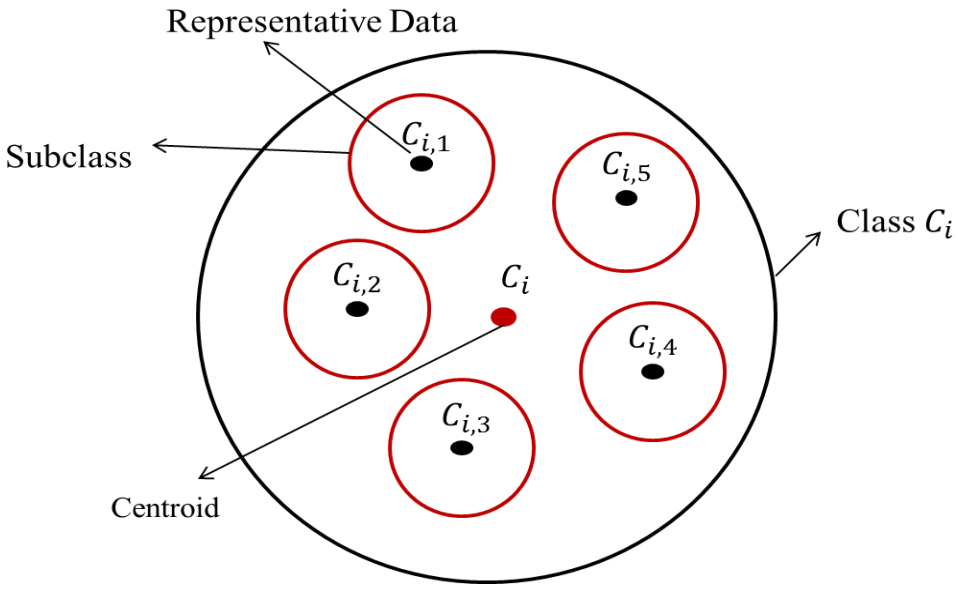


Figure 5.7: Class, Subclasses and Representatives Data

5.3 Prevention Algorithm

The main role of SEA is to respond to any abnormal behavior in a system by preventing the detected abnormal behavior. Many preventive actions can be performed to stop the abnormal behavior in the network system. The response techniques has been introduced earlier in section 2.4.2.3. Now for more clarification in our study, we categorize the response actions into host-based prevention capabilities and network prevention capabilities.

5.3.1 Host-based Prevention Capabilities

The host-based prevention capabilities can be categorized into four categories as follows:

Code Analysis: The code analysis technique can prevent code from being executed, including malware and unauthorized applications.

Network Traffic Analysis: This can stop inward network traffic from being processed by the host, and outgoing network traffic from exiting it. This might be done to stop network, transport, and application layers attacks, as well as to stop the use of unauthorized applications and protocols. Analysis can also identify malicious files being downloaded or transferred, and prevents those files from being placed on the host.

Network Traffic Filtering: This category works as a host-based firewall; it can stop illegal access and use adequate rule violations. It is effective only for stopping activity that is identifiable by IP address and TCP port, UDP port, or ICMP type and code.

File System Monitoring: This prevents files from being accessed, modified, replaced, or deleted, which could stop malware installation, including other attacks involving inappropriate file access.

5.3.2 Network-based Prevention Capabilities

Network-based prevention capabilities can be categorized into three categories: monitoring behavior, passive, inline or hybrid i.e. inline and passive [25].

5.3.2.1 Passive Only

Ending the Current TCP Session: This is will attempt to end an existing TCP session by sending TCP rearranged packets to both endpoints. The sensor does this to both endpoints by making it to appear to each endpoint that the other endpoint is trying to end the connection. The goal is for one of the endpoints to terminate the connection

before an attack can succeed. Since this technique is only applicable to TCP, it cannot be used for attacks carried in other types of packets, including UDP and ICMP.

5.3.2.2 Inline Only

Performing Inline Firewalling: Most inline IPSs offer firewall capabilities that can be used to decline doubtful network activity.

Throttling Bandwidth Usage: If a specific protocol is being used inappropriately, such as for a denial of service (DoS) attack, malware distribution, or peer-to-peer file sharing, some inline IPS can set the percentage of network bandwidth that the protocol can use. This prevents the action from harmfully affecting the bandwidth usage for other resources.

Altering Malicious Content: Inline IPS can disinfect part of a packet, which means that the malicious content is replaced with benign content, and the disinfected packet is then sent to its destination. IPS that acts as a proxy might perform automatic normalization of all traffic, such as repackaging application payloads in new packets. This has the effect of disinfecting some attacks involving packet headers and some application headers, whether or not the IDPS has detected an attack. Some sensors can also strip infected attachments from e-mails and remove other discrete pieces of malicious content from network traffic.

5.3.2.3 Both Passive and Inline

Reconfiguring Other Network Security Devices: IPS can train network security devices such as firewalls, routers, and switches to block certain class of activities or route it elsewhere. This prevention procedure is useful only for network traffic that can be discriminated by packet header characteristics typically recognized by network security devices, such as IP addresses and port numbers.

Running a Third-Party Program or Script: Some IPS can run a script or programwhen certain malicious activity is detected. This could trigger any prevention action desired by the administrator. Third-party scripts are most commonly used when

the IPS does not support the prevention actions that the administrators want to have performed.

Since we are looking at hybrid intrusion prevention system, we implement our simulation of the prevention part according to the occupant of the IPS system. The algorithm is designed to switch between different types of prevention categories as illustrated in figure 5.8. We can assume that:

1- C_{abi}, abnormal behavior is returned by the detection algorithm in real time and has a vector of feature such that:

$$C_{abn} = [f_{abn} \mid \forall\, n \geq 1] \tag{5.29}$$

where; f_{ab} is the set of features of an abnormal behavior.

2- C_{nbi} normal behavior is returned by the detection algorithm in real time and has a vector of feature such that:

$$C_{nbn} = [f_{nbn} \mid \forall\, n \geq 1] \tag{5.30}$$

where; f_{nb} is the set of features of normal behavior

Then, the output signal $outp$ is examined against the detection results:

$input :: C$ class identified in communication session.

$if\ result == C_{nbi}\ \ then,$

$outp = 1$

$else,$

$result == C_{abi}$

Signal::*Prevention response*

$end.$

f_{ab} is different according to which IPS prevention capabilities is specified by the user during configuration setting i.e. host-based or network-based.

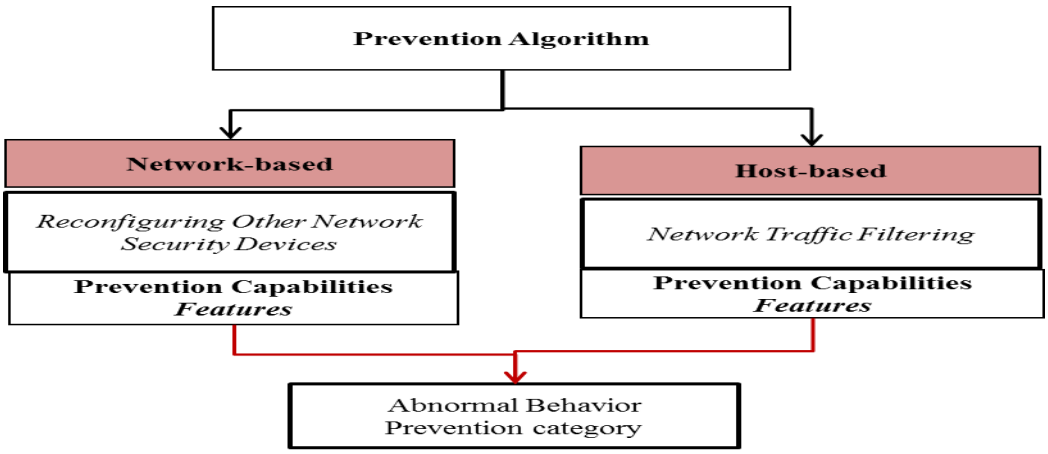


Figure 5.8: Prevention Algorithm Categories

5.4. Analysis Algorithm

The main function of analysis agent is to analyze detection message, in terms of abnormal behavior, sent by Sense Agent. The analysis agent has a knowledge base of all misused abnormal behaviors. When analysis agent receives the danger message, it starts searching in the database whether the abnormal behavior context f_{abi} is a misused or an anomaly.

$input :: DetectionMsg, f_{abi}$ $which$ Contains the subclass and the representative features

$search\ in\ knowledge\ base$

$if\ f_{abi}$ ∃ DB [f_{mab}] then,

$Outp:: MisusehealMsg, f_{mab}$

$else,$

$outp :: AnomalyMsg, f_{anab}, MisusehealMsg$

$end.$

The analysis agent analyze the features of abnormal behavior to identify whether it is a misused abnormal behavior or an anomaly abnormal behavior. If it is a misused, the healing system is able to respond easily and within a short time since the abnormal behavior activity and damage are known. This is inspired from the immune

memory cell where the antigens are matched with antibodies in the B-Cell, and the antigens are presented to T-Cell to identify the type of abnormal behavior.

5.5 Adaptive Algorithm

When the adaptive agent receives the $AnomalyMsg$, it starts to recognize the anomaly abnormal behavior f_{anab} and calculates the distance between the nearest normal behavior context f_{nbi} and the abnormal behavior context f_{anabi} that is given by:

$$y(f_{nbi}, f_{abi}) = \sqrt{\sum_{i=1}^{n}(f_{nbi} - f_{anabi})^2} \qquad (5.31)$$

Where; y is the deviation distance from the normal behavior.

The value of y give identification features of the anomaly abnormal behavior f_{anab} and allow discrimination of the abnormality from other types. Meanwhile, these deviation features are recognized and sent to the analysis agent.

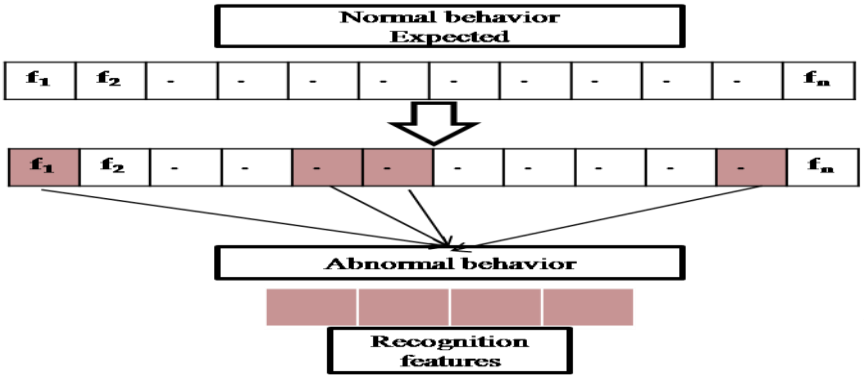


Figure 5.9: Recognition Features of Abnormal Behavior

The mechanism of the adaptive agent is mapped from the mechanism of the B-cell, which produces adaptive antibodies and continues producing antibodies until it recognizes the pathogens. A similar mechanism is created for the adaptive agent by generating a classification index for each class in the training phase. For example, if the abnormal behavior is classified as denial of service attack of type "*smurf* " , then it should have an index in the vector of the classes. This step is repeated by ADA until the abnormal behavior is fully recognized. The recognition process is simplified

in Figure 5.9. ADA agent sends these features and identity of the abnormal behavior of the anomaly to ANA agent for updating of the misuse database.

$input::AnomalyMsg, f_{anabi}$

$while\ i \leq n$

$f_{anabi} = y_i$

end

$outp :: RecognitionMsg, f_{anabi}$

5.6 Self-healing Algorithm

The algorithm of self-healing system is implemented if there is any predicted damage due to the abnormal behavior. The self-healing process performed by SHA agent follows the steps shown in Figure 5.10.

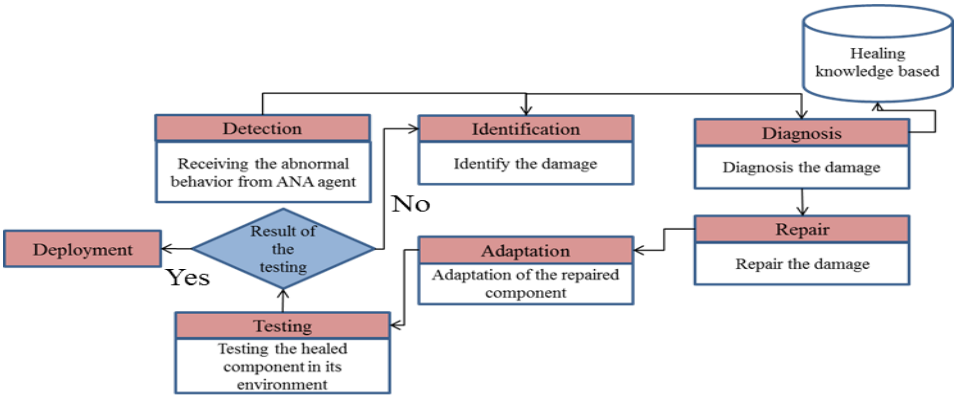


Figure 5.10 Self-healing Algorithm Steps

The healing system has replicated the actual system components into three typical dataset. The first one is to be subjected to periodic scanning to investigate whether any damage is found. The scanning is performed by equivalent comparison between the first copy of the dataset. If any damage is captured, the equivalent one in the copied dataset replaces the damaged component. These steps implement the detection, identification diagnosis and repair phases of self-healing algorithm. Secondly, the second copy performs equivalent comparison with third copy to verify that no damage is found in the two copies. Finally, the dataset of the repaired system component

performs another equivalent comparison with the third copy to validate that the healing has been performed perfectly.

As SHA receives the healing request messages from ANA i.e. $AnomalyhealMsg$ and $MisusehealMsg$ containing the abnormal behavior, characteristics and the damage behavior analysis, then SHA will start to perform the periodic healing process as described above and the algorithm is as follows:

$input :: MisusehealMsg, f_{mabi}$

$input :: AnomalyhealMsg, f_{anabi}$

$SYS = \{sys_n | 1 \leq n \leq \infty\}$

Where; SYS is the set of system components.

Replicate SYS twice.

Scan $f_{anabi} \vee f_{mabi}$ in the first copy.

SYS'=DI($f_{anabi} \vee f_{mabi}$)

where;

SYS' is the set of damaged system components.

DI is diagnosis function.

Search the R_i in DB_{heal}

where; R_i is replacement function and DB_{heal} is healing component in the second copy

$\forall DI \text{ in } DB_{heal} \exists R_i$: *Replace* *// repair state*

$Compare\ between\ second\ copy\ and\ third\ copy,$

$R_i(SYS') == SYS$

$Fix\ the\ generation\ of\ the\ heal\ component$

Scan in the first copy with third copy to verify adaptation.

$if\ SYS\ is\ adapted,$

$Deploy$

$Else, go\ to\ DI\ repeat\ the\ scan.$

$End.$

In order to perform self-healing, systems should have the ability to modify their own behavior in response to changes in their environment and discover alternatives; therefore, the healing function is tested until the healed component has adapted to the system. The Self-healing technique used is shown in figure 5.11.

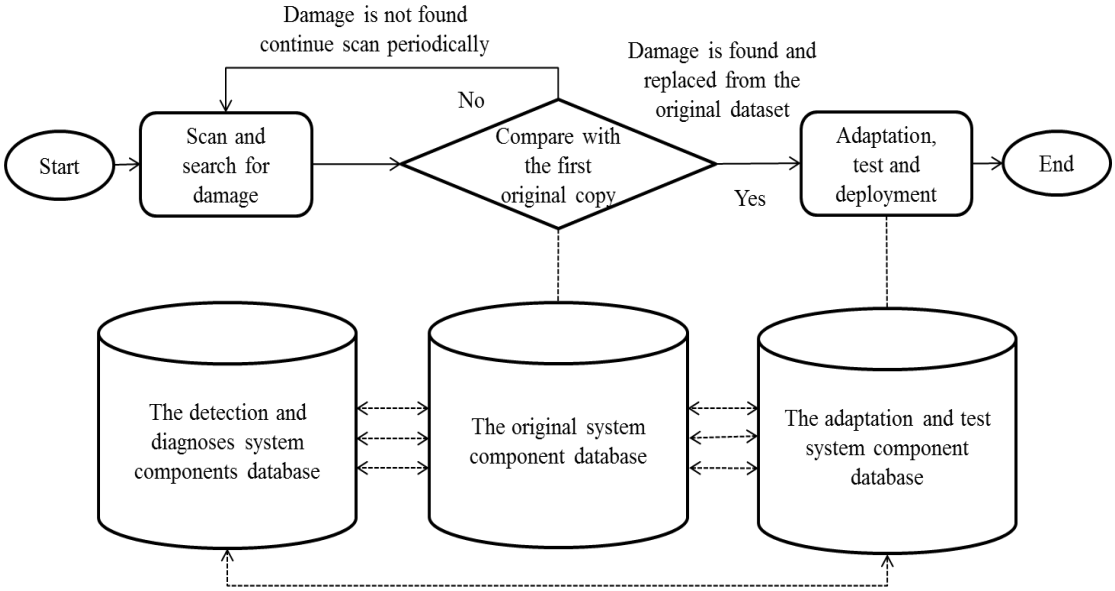


Figure 5.11:The self-healing Technique.

5.7 Chapter Summary

In this chapter, the description and derivation of the intrusion prevention and self-healing system algorithms has been explored. The basic principles of the intrusion detection and prevention systems use machine learning and pattern recognition classification. The classification algorithm is based on a new improvement of variance of the best feature, k-means cluster and Gaussian mixtures methods. The analysis and adaptation algorithms are established, as well as, the intrusion identification and recognition step. The self-healing computation and healing algorithm are accomplished using equivalent check between replicates of the system components datasets. In the following chapter, the system is simulated and the algorithms are implemented. The biological intrusion prevention and self-healing system will be tested using standard intrusion detection dataset and process behavior datasets to validate the established new algorithm.

CHAPTER 6
BIIPSS VALIDATION

6.1 Chapter Overview

In this research, we have developed an application system with specific design features mainly autonomous feature, and presented new algorithms for detection, prevention and healing mechanisms. To validate this system, we need to measure the effectiveness of this new approach in a real environment from the point of views of both the developer and user of the application. In security systems, developers consider metrics as fundamental for measuring the efficiency and cost of complex security controls. Security enhancement begins by identifying the metrics that measure various characteristics of security for the endeavor. Security metrics, which describe a measure for the security of an entire organization, are quite a new area of research. Without commonly established security metrics, it would be very complicated to improve the security of network system. In this chapter, we will cover validation of the algorithms against the metrics for network security from the perspective of IPS and SH model, and present the major technical-operational metrics used by large security system developers and researchers.

6.2Validation Metrics and Measurement

Managing the security of critical services network system has become a considerable issue. As any other procedure, security cannot be managed, if it cannot be measured. The need for metrics is important for assessing the current security status, to develop operational best practices and for guiding future security research. The issue is essential at a time when the researchers and developers need to simulate the implementation, and test their models. Metrics emphasize on the main concerns of security systems such as: abnormal behaviors, vulnerabilities and risks. Metrics build

application area information assets based on either quantitative or qualitative measure. The current policies for evaluating or validating IT systems and network security are focused on examining the results of security assessments which include diffusion testing, vulnerability scanning, and other means of probing defenses for weaknesses in security, and on examining the building blocks, processes, and controls. Metrics assessable standards observe the effectiveness of aims and objectives established for network security system. They measure the implementation of security strategy, the effects of security services and the impact of security events on the presented security objectives and mission.

Network security metrics can be obtained at different levels within our developed IPS and self-healing system. Detailed metrics, collected at the system and network level, can be aggregated and rolled up to progressively higher levels, depending on the size and complexity of the implementation area. If measurements are instant snapshots of a fussy measurable parameters, then metrics are more complete representations, typically comprised of several measurements, baselines, and other supporting information that provide context for interpreting the measurements. Perfect metrics are objectives oriented and should have the following features: precise, measurable, comparable, manageable, repeatable, and time dependent. In this chapter, we divide the metrics for the IPS and SH into: detection metrics, prevention and adaptation metrics. Self-healing metrics, and IPS and SH combination metrics are new which are introduced in the next chapter. The following section explains these metrics in details.

6.3 Detection Metrics

A fundamental problem in intrusion detection is what metrics can be used to evaluate IPS and an intrusion detection system (IDS) in terms of its ability to properly classify network data packet events as normal or intrusive. There are two main issues facing intrusion detection system: they may possibly raise a huge number of alarms and false alarms. Thus, the objective is to decrease the number of false alarms as explained in [127], [128]. The lack of a single unified metric makes it difficult to fine-tune and evaluate IPS. Traditional metrics are as defined by Guofie et al. [129]:

False Negative (FN) rate: The chance there is no alert,$\rightarrow$ A, when there is an intrusion, I.

$$FN = P(\rightarrow A|I)$$

$$= \beta \quad (6.1)$$

True Positive (TP) rate: The probability that the IDS outputs an alarm, A, when there is an intrusion, I.

$$TP = P(A|I)$$

$$= (1 - \beta) \quad (6.2)$$

False Positive (FP) rate: The probability that the IDS outputs an alarm, A, when no intrusion occurs, $\rightarrow$ I.

$$FP = P(A| \rightarrow I)$$

$$= \delta \quad (6.3)$$

True Negative (TN) rate: The chance there is no alert, $\rightarrow A$, when there is *no* intrusion, $\rightarrow I$.

$$TN = P(\rightarrow A| \rightarrow I)$$

$$= (1 - \delta) \quad (6.4)$$

Up to date researchers and developers are unable to define a single metric that seems sufficient to measure the capability of intrusion detection systems.

6.4 Prevention Metrics

An Intrusion Prevention System (IPS) capability requires assembling data packets received in real time network connection to restore the original network system protocols, rules and flags to gain an awareness of traffic behavior. In order to achieve this, a significant amount of processing power must be available. Performance measurement of IPS prevention capabilities is complicated by the lack of a standard on what intrusions must be caught. Therefore, when evaluating the performance of

IPS, a number of metrics need to be considered to evaluate the prevention events. The present model defines the prevention metrics according to the design specification as follows:

Positive predictive value (PPV): The probability of a chance that an intrusion I, is present when an IPS outputs an alarm and response, A.

$$PP = P(I|A) \tag{6.5}$$

As PPV is higher, this indicates that the prevention response is the correct and confident response.

Negative predictive value (NPV): The probability of a chance that there is no intrusion$\rightarrow I$, when an IPS does not output an alarm and response, A.

$$NP = P(\rightarrow I| \rightarrow A) \tag{6.6}$$

As NPV is high the permission to continue receiving data is the correct and confident response.

Throughput: The measurement of the amount of data an IPS can process per second/Bytes of data. The data can be packaged in one large, or many smaller packets meaning that the related measurement of how many small packets can be handled per second is a key to reflecting the true throughput value.

Consider dat is the amount of data that can be measured in bit or byte in each prevention process. Then:

$$thr_{put} = \frac{dat}{t_{p_{ab}}} \tag{6.7}$$

where;

thr_{put} is the throughput.

$t_{p_{ab}}$ is the time to process instance of abnormal behavior .

Concurrent Connections: This value identifies the number of sessions that an IPS is able to maintain.

Consider the set of abnormal processes in the concurrent sessions is p_{abn}.

$$p_{abn} = \{p_{ab1}, p_{ab2}, \dots, p_{abn}\} \qquad 1 \leq n \leq \infty$$

Then, the concurrent connectioncon_{cont} can be calculated as:

$$con_{cont} = \frac{\sum_{n=1}^{\infty} p_{abn}}{t_{allp\ ab}} \tag{6.8}$$

where; $t_{allp\ ab}$ is the time to process all abnormal behavior instances.

Latency: A measurement of the time taken to process a single abnormal behavior packet within the system. Here the term latency refers to the delay between detecting and preventing abnormal behavior packet.

Consider the process being p_{ab} as the process of a single abnormal behavior, then:

$$t_{p_{ab}} = t_{prev} - t_{det} \quad \forall\ p_{ab}\ instance \tag{6.9}$$

where;

$t_{p_{ab}}\ is\ latency;$

$t_{prev}\ is\ the\ prevention\ time;$

$t_{det}\ is\ the\ detection\ time.$

The throughput and concurrent connection metrics, both are affected by the prevention time measurement, which is the latency metrics.

The measurement of latency and concurrent connection metric need real implementation of the whole IPS system in the network system, which is out of the scope of this research. The latency metric is based on the bandwidth, routing system and other network engineering parameters, and the settlement of the networks. At this stage of developing and simulating, IPS algorithm has limited scope; only deployment of the IPS suggested in this research is able to help the network designer and vendors of IPS system to achieve the best for their critical services network. In next chapter

introduced two new metrics for prevention predictability trust which can be used to evaluate the present system and any developed IPS.

6.5 Analysis and Adaptation Metric

The analysis to detect abnormal behavior is to identify whether it a misuse or anomaly by scanning the misuse knowledge base. Our metric here is concerning classification of the type of abnormal behavior. For the purpose of this research, we define a new metric for abnormal behavior analysis which measures the system ability to discriminate between misuse and abnormal behavior from the extracted signature during the detection process. From chapter 4, when the analysis agent scans a detected abnormal behavior, $\mathrm{BH_{f'}}$, there are two possibilities: either the abnormal behavior is already existing in the misuse database or not. Our metric is used to measure the scanning accuracy; analysis accuracy is to investigate whether the analysis and adaptation have been completely done. This was simulated during the test stages according to the algorithm of analysis and adaptation. Using the data classification, specific type of abnormal behaviors is represented by j of sub classes, C_j^i and each subclass data has representatives instance of data, $P_{C_j^i}$. The features of $P_{C_j^i}$ is pointed towards the discrimination and adaptation criteria of the detected abnormal behaviors. The measurement of the adaption and analysis is the error in classifying the right type of abnormal behavior, and can be defined as Cl_{ER} as given by:

$$Cl_{ER} = \frac{\text{number of errors of abnormal behavior classes}}{\text{the total number of abnormal behavior instances}}$$

Cl_{ER} is a metric that can be used to measure the analysis and adaption performance of the analysis and adaptation agents, and is considered as an additional contribution in this research.

This metric is calculated and measured as the result of the system simulation. The next section demonstrates the simulation processes and results.

6.6 Validation of the IPS and SH System Algorithms

To validate the proposed new algorithms, three different datasets were used. The aim was to examine if the system is able to detect some abnormal behaviors after its training period. The first dataset is a version of the KDD Cup 1999 DARPA intrusion detection evaluation dataset generated and managed by MIT Lincoln Labs published in [130]. The second is system process dataset collected by University of New Mexico (UNM) which was used to test computer immune systems [131]. Finally, the self-healing algorithm is simulated using grid network dataset to investigate the system behavior in grid network system. The self-healing system result and reliability will be discussed in chapter 7. In order to accomplish the validation and performance testing of the algorithms, we consider the following limitations:

1. Due to the complexity of designing a network system for specific critical services, the system is tested using standard real time statics datasets, which are real trace datasets and are used to validate many algorithms of intrusion detection and preventions, and have been used by many researchers.
2. The measurement of actual prevention response needs real implementation and computer engineering design, which are out of the scope of our research since we implement software engineering design. The possible solutions of prevention are merely our suggestion for consideration by the network designer and/or vendor of the critical services who use the system. They must test and decide which solution is the best for their services requirements. Due to security policy and confidentiality of some critical information, the system could not be implemented in the university campus. The healing system is simulated using trace data for the system components usage information. Such as the information of memory usage, processing time needed and CPU consumption time. Although in this work, SH system simulation is static and not real time but, the same will happen in the real time; but the difference is only how long will be the period to trigger the SH system autonomously, which also depends on vendor requirements and network design capabilities.

The following subsections explain the KDD Cup and UNM datasets, which are used in the simulation.

6.6.1 MIT Lincoln DARPA Intrusion Detection Evaluation Dataset

The DARPA KDD Cup Intrusion Detection Evaluation Program was formulated and managed by MIT Lincoln Labs. The aim of DARPA was to study and assess research in intrusion detection and test the performance of intrusion detection and prevention systems. Audited standard set of data is used in the simulation, which is comprised of a wide variety of intrusions simulated in a military network environment, was provided. The 1999 KDD Cup intrusion detection contest had used a version of this dataset, and summarized it into network connections with 41-features per connection [130], [132]. This arranged dataset became the benchmark in international knowledge discovery and data mining tool competition. The KDD Cup intrusion detection dataset is a log connection traffic from MIT Lincoln Labs which contains connection details in its network, such as traffic data (tcpdump, inside and outside network traffic), Basic Security Module (BSM) audit data, file systems data and many related information. Lincoln Labs set up a situation to obtain nine weeks of raw TCP dump data for a local-area network (LAN) simulating a typical U.S. Air Force LAN. They operated the LAN as if it were a true Air Force environment, but scattered it with multiple abnormal behaviors. The original training data were about four gigabytes of compressed binary TCP dump format from simulations of network traffic conducted in a period of seven weeks. Those simulation data were then processed into approximately five million network connection records. Two weeks of test data yielded around two million connection records. Abnormal behaviors fall into four main classes [133]:

- DOS- denial-of-service: This class has 6 categories which are smurf, Neptune, back, teardrop, pod and land. The algorithms were tested to classify all6 categories.
- R2L- unauthorized access from a remote machine: This class has 8 categories which are warezclient, guess_passwd, warezmaster, imap, ftp_write, multilop, phf, and spy. The algorithms were tested to classify 4 of these categories due to the lack of training data of the other 4 categories.
- U2R- unauthorized access to local super user (root) privileges: This class has 4 categories which are buffer_overflow, rootkit, loadmodule and prel. The algorithms were tested to classify all 4 categories.

- Probing- surveillance and other probing: This class has 4 categories which are satan, ipseewp, portsweep and nmap.

The process of collecting the dataset resulted in 41 features for each connection between the dataflow from source IP address to destination IP address. For each connection, the features are clustered into four categories [134]:

Basic Features: Basic features can be derivatives from packet headers without inspecting the payload.

Content Features: Domain knowledge is used to assess the payload of the original TCP packets. This includes features such as the number of failed login Attempts.

Time-based Traffic Features: Properties that mature over a two second temporal window are captured as time based traffic features such as counting the number of connections to the same host over the two second interval.

Host-based Traffic Features: Exploit a historical window assessed over the number of connections, in this case 100, instead of time. Host based feature are therefore designed to assess attacks, which span intervals longer than two seconds.

Details of the types, features of the dataset are shown in Appendix D.1. The data used in this research are a part of 10% of the whole training data detailed in [133], [134]. Each class of data is tested individually and another test is taken for the whole class against normal class. It is important to note that the test data are not from the same probability distribution as the training data, and they include abnormal behavior types not in the training data. Both training and testing data are taken randomly from the original dataset. This makes the task more realistic. Some intrusion experts consider that most anomaly abnormal behaviors are alternatives of known abnormal behaviors, and the "signature" of known abnormal behaviors can be sufficient to catch innovative alternative abnormal behaviors. The dataset contains a total of 23 training abnormal behavior types. For the purpose of this research, the abnormal behavior data have been divided into the four mentioned categories. Each abnormal behavior category dataset is classified against normal data using the classification algorithm.

6.6.2 Process Behavior Dataset

The University of New Mexico (UNM) research group of computer immune system collected several datasets consisting of system call traces for many processes on SUN SPARC stations running unpatched SUNOS 4.1.1 and 4.1.4. These collections of datasets are publicly used in processes behavior detection domain, specifically, have been used to train and test several process behavior intrusion detection systems [131].

In UNM datasets, the processes and corresponding system calls were accumulated from different types of programs (e.g. programs that run as daemons and those that do not), programs that diverge widely in their size and complexity, and different kinds of intrusions (buffer overflows, symbolic link attacks, and Trojan programs). In these datasets, only those programs that run with privilege have been included, because misuse of these programs has the highest possible harm to the system. Some of the normal data are "synthetic" and some are "live".

Synthetic traces are collected in production environments by running an arranged script; the program selections are chosen exclusively for the purpose of exercising the program, and not to meet any real user's requests. Live normal data are traces of programs collected throughout normal usage of a production computer system. In various cases, data for the same program have been collected from multiple locations and/or multiple versions of the program. Each of these is a dissimilar dataset; normal traces from one set can be quite different from those of another. Each trace is a sequence of (process id, system call number). System call numbers are stored in the order in which it is executed. There is a mapping file that associates the system call numbers to the corresponding system call names. The set includes normal traces and abnormal traces. Sequence lengths differ because of differences in program complexity and because some traces were daemon processes and others were not. The samples of each datasets are shown in Appendix D.2.

To comport the simulations to exam process behavior; sendmail daemon was used for studying the classification of normal behavior and to detect anomalous process behavior. The syslog intrusions in UNM dataset are simple examples of buffer overflows in sendmail. Though patches are currently available for most of the vulnerabilities, sendmail and the buffer overflow attacks on Sendmail are worthy

examples for simulation result. Sendmail daemon was examined for detection of buffer overflow attacks.

In order to construct a good classifier, we need to gather sufficient amount of training data and identify the set of meaningful features. Due to the unavailability of enough varieties of intrusion trace data, further simulation runs were conducted using datasets with specific arrangement, where the given input sequences of process system calls have to be divided into separate classes known as sequence sets. The input sequence for each process is converted into frequency components of the system calls. The assembling information of neighboring system calls in the input sequence it misplaced and only the frequency of each system call in the sequence is conserved. Intrusion in this representation is defined according to frequency count of system calls. 69 common system call frequencies are used as the classification features.

6.7 IPS and SH Simulation Results

In order to regulate the training and test datasets of different categories of abnormal behavior, variance features and problem representation on the dynamics of the classification and self-healing algorithms, validation of the algorithm is examined across a range of different parameter settings. While the classification algorithm works efficiently for real time applications, when evaluating the system performance over many runs such as examining different dataset categories and testing the effects of variations in multiple parameters, simulating the algorithms can be quite time consuming. As an example, we have 41 different features in KDD Cup dataset that need to be evaluated for the best results. Due to the requirement of intrusion prevention for validation against different metrics, the algorithm was run for different train and test dataset for the two different standard datasets described in section 6.6.1 and 6.6.2. For the simulation purpose, matlab2009b software has been used and the simulation was run in normal PC. The source code of the main part of simulation demonstrated in Appendix E. Each dataset has been trained and tested individually for different parameter settings under the different conditions and arrangement of the two standard datasets. The parameters considered in the simulation process are defined as follows:

Features variance: Estimated best feature that gives maximum value of the new variance metric which contributes to high classification accuracy. This is mapped from HIS when we need to identify the antigen that binds to specific type of pathogen molecules, death cells and antigens.

Sub class number: The number of sub class in each class of data.

α: The significant decrease in variance. These values have been explained in chapter 5 by equation (5.10) as given below. This is mapped to the affinity maturation of the dendritic cell when it binds to specific pathogens and migrates to lymphocyte nodes to present the antigens to T-cell.

$$\frac{\Delta var}{var} \leq \alpha,$$

where; $0.0 \leq \alpha \leq 1.0$.

Size of sub class: The size of data processed in each subclass; represents the number of normal behavior and abnormal behavior in each class, this is mapped from the immune system; the number of safe signal and danger signal presented by dendritic cells.

Error position: Represents the error position of the classification, i.e. which normal data are classified as abnormal and vice versa.

Number of representative in each sub class: The number of data that can represent each class. This is mapped from the number of dendritic cells that can bind to the same type of pathogen.

These parameters and their variations affect the simulation results and system reliability. The next subsections discuss the simulation condition, arrangement and results. The result is the metrics mentioned in sections 6.2, 6.3, and 6.4.

6.7.1 Validation of IPS and SH algorithms Using KDD Cup Datasets

To test the IPS algorithms, 4 smaller subsets of the KDD cup dataset have been created randomly for each abnormal behavior classes with normal data. The ratio

between the normal records to the abnormal records is around 5:1 for probe, denial of services and R2L classes while for U2R around 2:1. Both trained and tested datasets for each individual class have the same ratio. The test for each sub dataset was run twice: first, all the features were tested to identify which features will give maximum accuracy; secondly, the test was run again with iteration stop at the first feature that gives the maximum accuracy. The results of classification are as follows:

First, the algorithms were applied to denial of services DoS abnormal behavior dataset. The simulation produced the results detailed in Table 6.1. The metrics mentioned in section 6.2 were calculated.

Table 6.1: Results of Detection Accuracy Using Denial of Service Dataset

Maximum Accuracy %	TP %	FP %	FN %	TN %	Identified abnormal behavior
99.79	99.86	0.07	0.14	99.93	6

The maximum accuracy is 99.79% and the result shows very low false positive error in detection. As mentioned in section 5.2.4 the accuracy is depending on the best features of classification and the value of α. To estimate the features and characteristics of each feature, please refer to Appendix D.1. The result demonstrated in Table 6.1 shows very low false positive and false negative error rates.

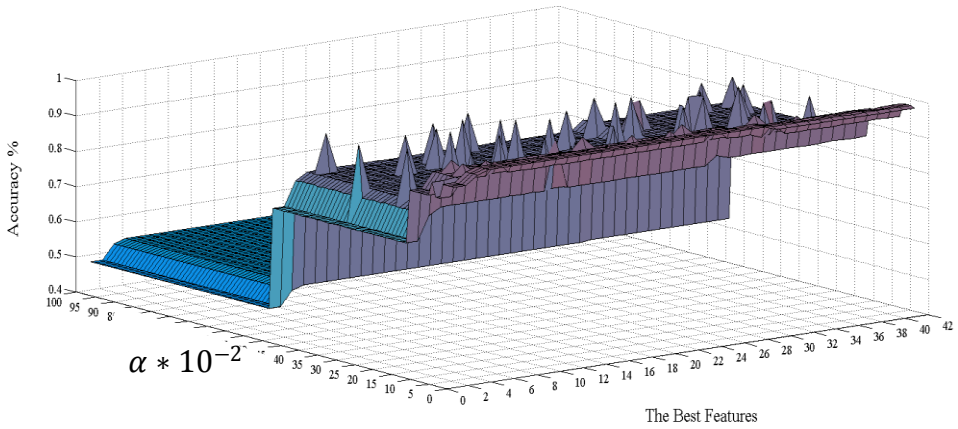


Figure 6.1. Denial of Services Simulation Test - values of accuracy against values of the best features and *α*.

In the first run of the algorithms using DoS dataset, which includes 6 different categories of DoS abnormal behavior, the variation of the accuracy due to the change of the features and α is shown in Figure 6.1. Changes in the accuracy value depends on the variances of the features i.e the best of the features, and according to this the values of α. The maximum accuracy is reached at different features. From this result, the fast and best feature (4) that gives the maximum accuracy is estimated and this value was applied in the second run.

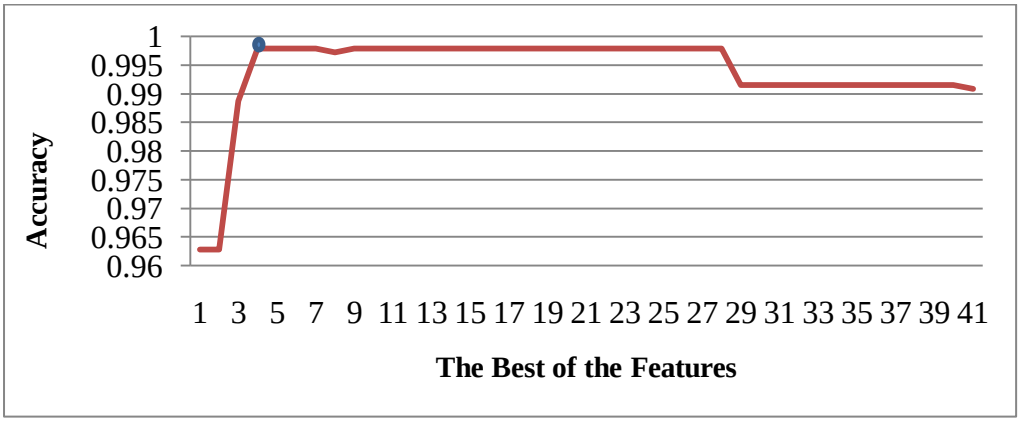


Figure 6.2: Denial of Services Dataset Test-the accuracy values against the best features.

In Figure 6.2, the change of accuracy at this specific value of feature is clearly shown. In the second run at the best feature, the variation of accuracy due to changes in the value of α at the best classification feature is obtained and shown in Figure 6.3. Accuracy has range 99.79-99.58% at the best feature (4).

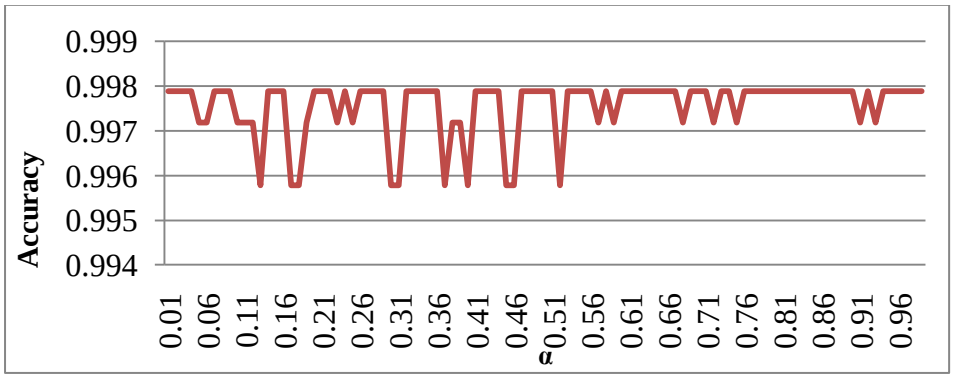


Figure 6.3: Denial of Services Dataset Test-accuracy values at best feature with α.

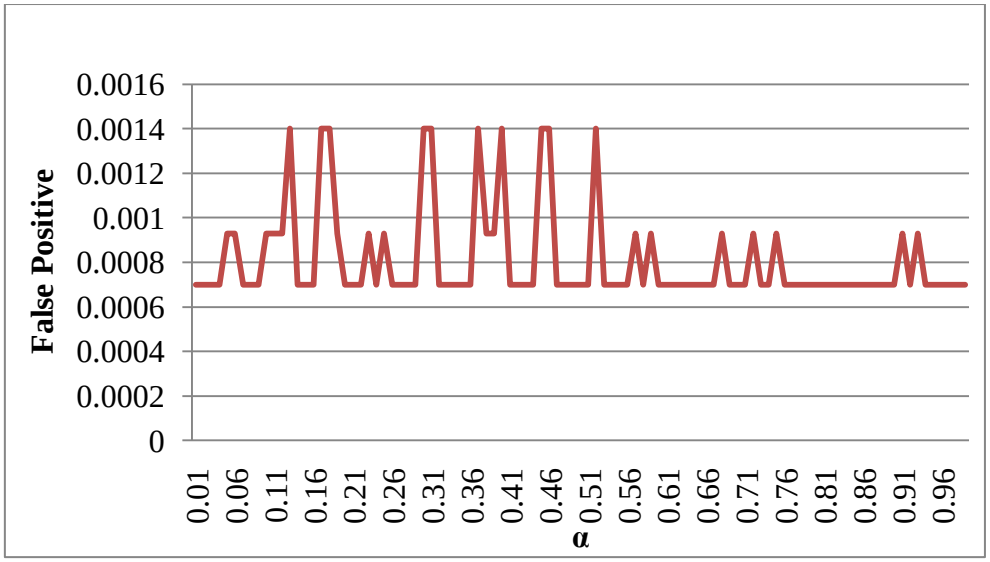


Figure 6.4: The Denial of Services Dataset Test-the false positive errors value with α.

α values also has a substantial effects on the values of the false positive and false negative. As shown in Figures 6.4 and 6.5. The minimum value of the false positive is 0.07% at maximum accuracy and feature number (4) and has range 0.07-0.14%. The value of the false negative is 0.14% and has range 0.14-0.28% at the best feature (4).

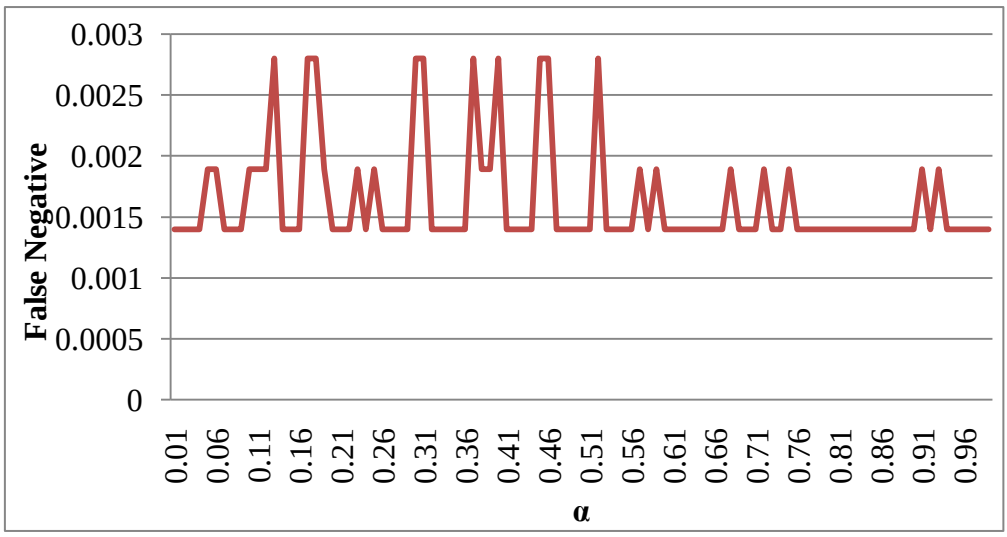


Figure 6.5: The Denial of Services Dataset Test-the false negative error values with α.

Table 6.2: Results of Detection Accuracy Using Probe Dataset

Maximum Accuracy %	TP %	FP %	FN %	TN %	Identified Abnormal behavior
99.92	100.00	0.08	0.00	99.92	4

The second simulation test of the algorithms has been applied to probe abnormal behavior dataset which includes 4 different probe categories of abnormal behavior. Table 6.2 shows that the maximum accuracy is 99.92 % and the result show very low false positive errors and 0 false negative errors. The values of true positive and true negative are also demonstrated. The variation of accuracy due to the change of the features and α is shown in Figure 6.6. The true positive value is 1.0 and the true negative value is 0.9992, approaching to (1). These results indicate that there are fewer errors in the data classification.

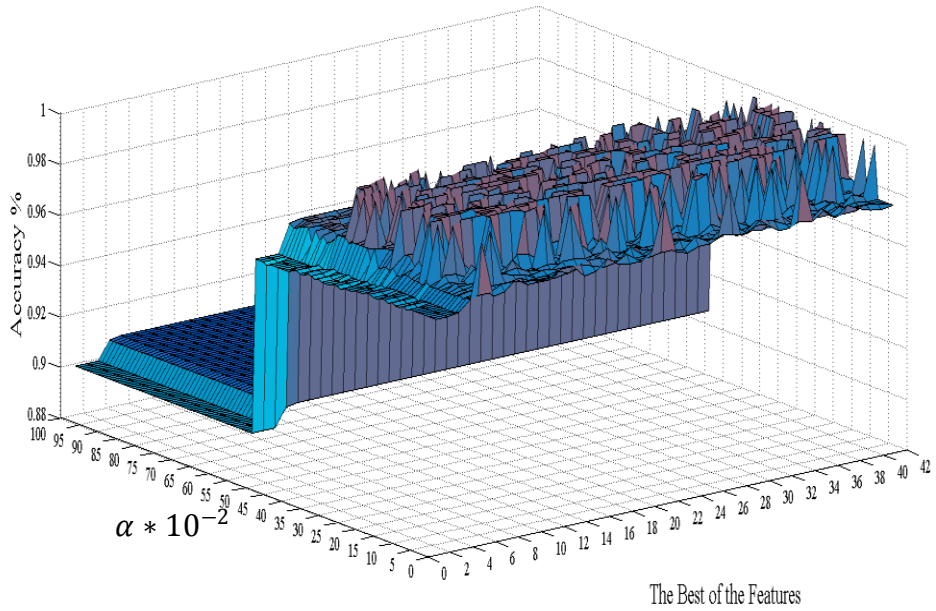


Figure 6.6: The Probe Simulation Test - accuracy values against the best features and α.

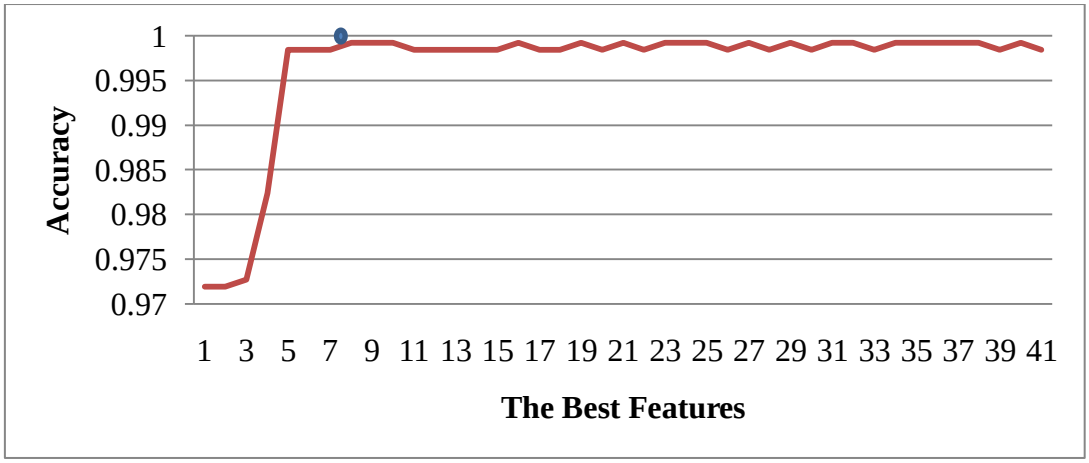


Figure 6.7: The Probe Simulation Test-the accuracy values with the best features.

For the probe simulation test, the change of the accuracy value is dependent on the variances of the features i.e. the value of the features and according to this the values of α. The maximum accuracy is reached at different features. The first feature that gives the fast and maximum accuracy value is feature number (8). In Figure 6.7, the change of accuracy with the best features is clearly shown.

From this result, the best feature that gives the maximum accuracy was estimated and applied in the second run. The variation of accuracy due to the change in the value of α at the best feature (8) of classification is given in Figure 6.8. Accuracy values vary from 99.92 to 90.71% at the best feature with different value of α.

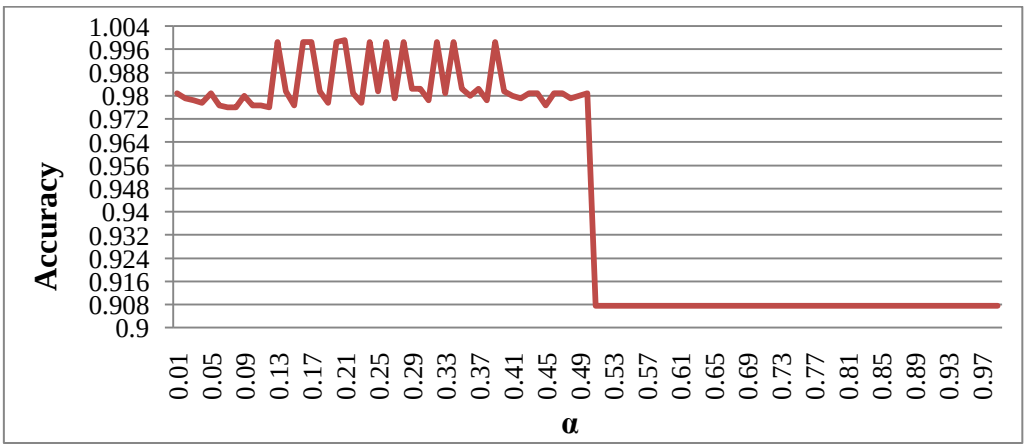


Figure 6.8: The Probe Simulation Test - change of accuracy at best feature with α.

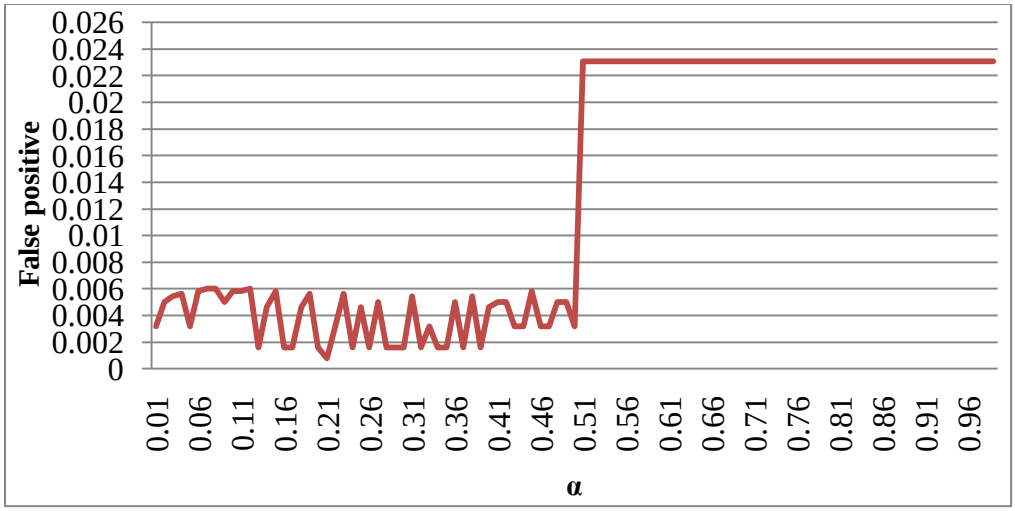


Figure 6.9: The Probe Simulation Test - the false positive error values against α.

The values of the false positive error at the best feature (8) vary from 0.08 to 2.3% at different value of αwhichis clearly shown in Figure 6.9. The values of the false negative error at the best feature (8) vary from 0 to 6.99 % at different value of α as is given in Figures 6.10.These results show the strength of the system against the different probe abnormal behavior.

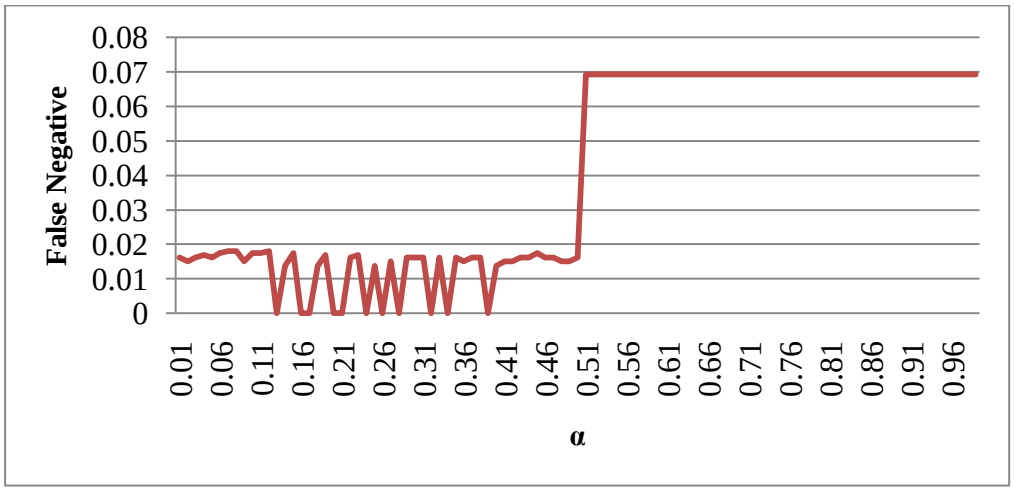


Figure 6.10: The Probe Simulation Test - the false negative errors value against α.

The third simulation test of the algorithms has been applied to unauthorized access from a remote machine R2L abnormal behavior dataset. For the R2L simulation test, the dataset is small compared to the previous one. This is due to the availability of R2L abnormal behavior records, but the ratio between the normal

behavior and abnormal behavior (5:1) is kept the same. The change of the accuracy value is dependent on the variances of the features i.e. the value of the features and according to this the values of α.

Table6.3: Results of Detection Accuracy Using R2L Dataset

Maximum Accuracy %	TP %	FP %	FN %	TN %	Identified Abnormal Behavior
99.60	99.6	0.0	0.4	100.0	4

The test results are shown in Table 6.3. From these results, the best feature that gives the maximum accuracy was estimated and applied in the second run; this is the value of feature number (3) and the maximum accuracy is 99.6%. In Figure 6.11, the change of accuracy at this value of feature and α is clearly shown.

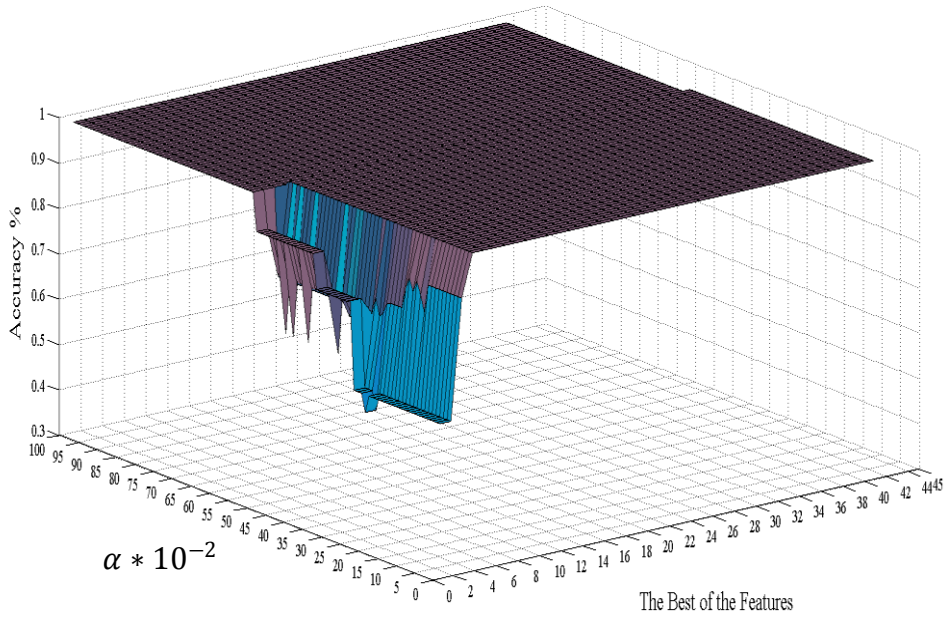


Figure 6.11: The R2LSimulation Test - accuracy values against the best features and α.

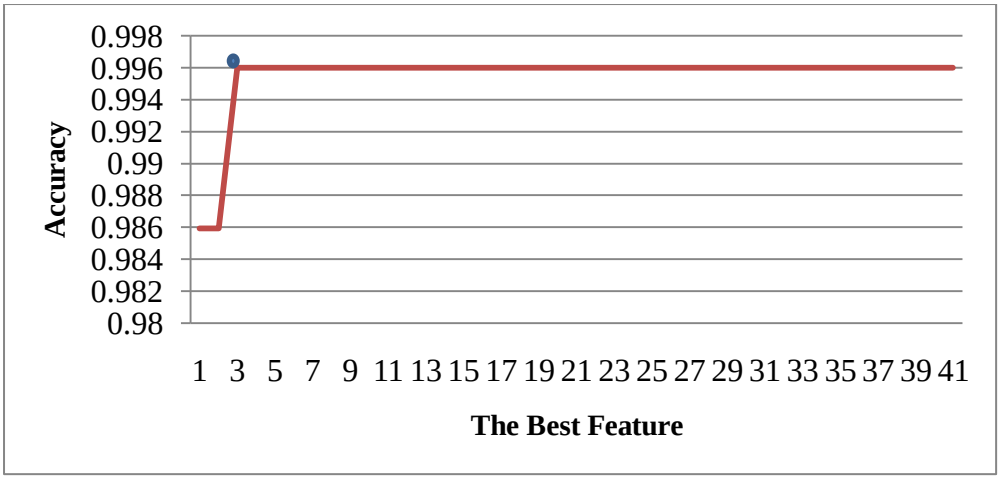


Figure 6.12: The R2L Simulation Test - the accuracy values with the best features.

In this test, 4 categories have been identified with 0 false positive errors at the best feature. High error rate in the false negative 0.4% is estimated which must be reduced in the future works for classifying R2L categories.

The variation of accuracy due to changing value of the best classification features is given in Figure 6.12. The maximum value is 99.6% and this is the value estimated at most features. The first and fast feature is (3). Figure 6.13 shows the change of accuracy at the point of the best feature with the values of α which vary from 98.6 to 99.6%.

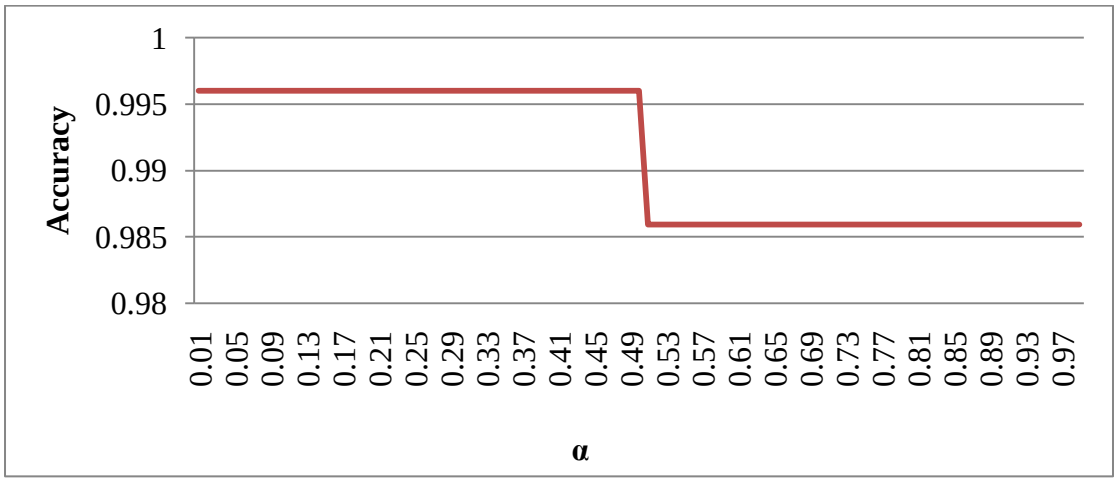


Figure 6.13: The R2L Simulation Test - change of accuracy at the best feature with α.

The values of the false positive error at the best feature (3) are vary from 0 to 1.3 % as is clearly shown in Figures 6.14, and this value could not be gained with any AIS system for intrusion detection and prevention system. Meanwhile, the value of the false negative error is slightly higher than the previous two simulation tests, but still showing some improvements compared to previous AIS systems.

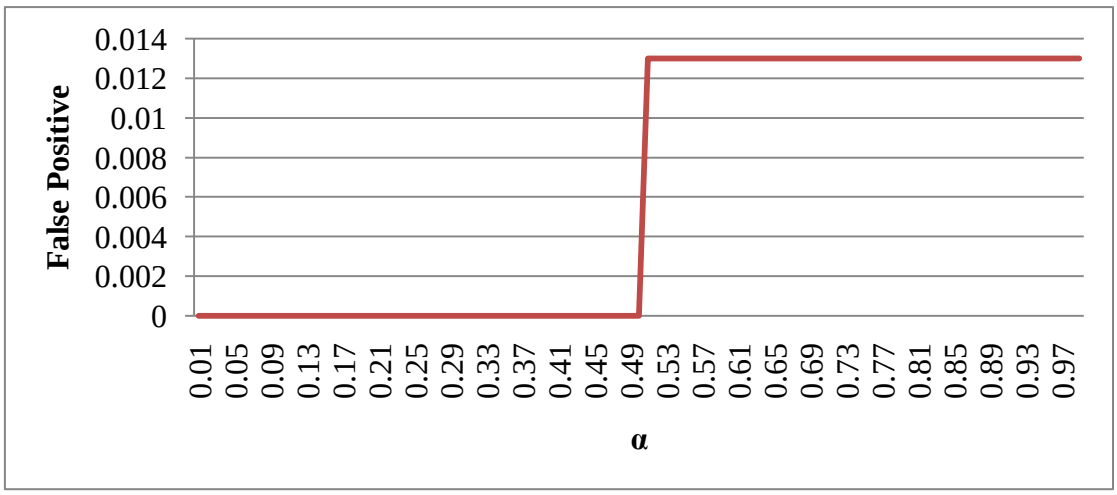


Figure 6.14: The R2L Simulation Test - the false positive error values against α.

Figure 6.15 shows the change of values false negative error with the change of α in the second run at the best feature (3)which are vary from 0.1 to 0.4%.

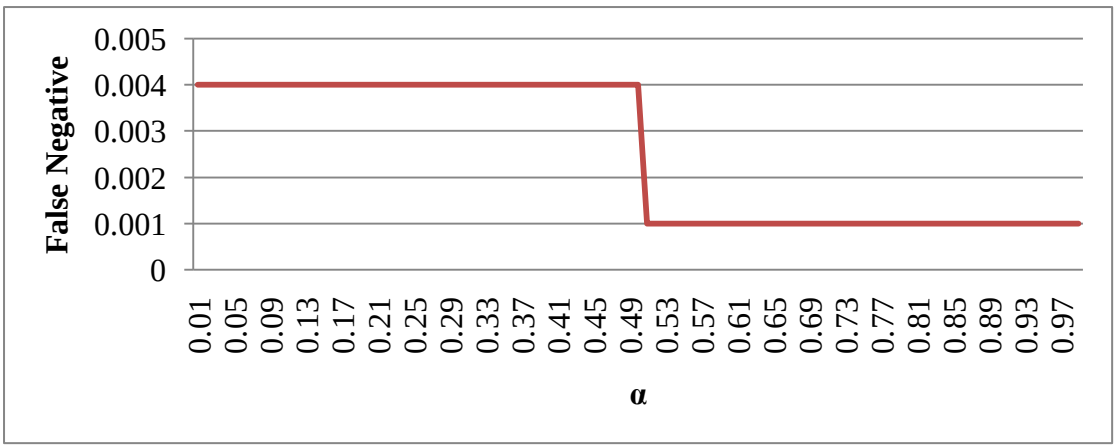


Figure 6.15: The R2L Simulation Test - the false negative error values against α.

The final simulation algorithm test for the KDD Cup classes dataset was applied to unauthorized access to local super user (root) privileges U2R abnormal behavior dataset.

Table6.4: Results of Detection Accuracy Using U2R Dataset

Maximum Accuracy %	TP %	FP %	FN %	TN %	Identified Abnormal behavior
100.00	100.0	0.00	0.00	100.00	4

The results obtained from the run at the best features are shown in Table 6.4. The maximum accuracy is 100% and the result shows 0 false positive errors and 0 false negative errors at the best features. In Figure 6.16, the change of accuracy at this value of feature and α is clearly shown.

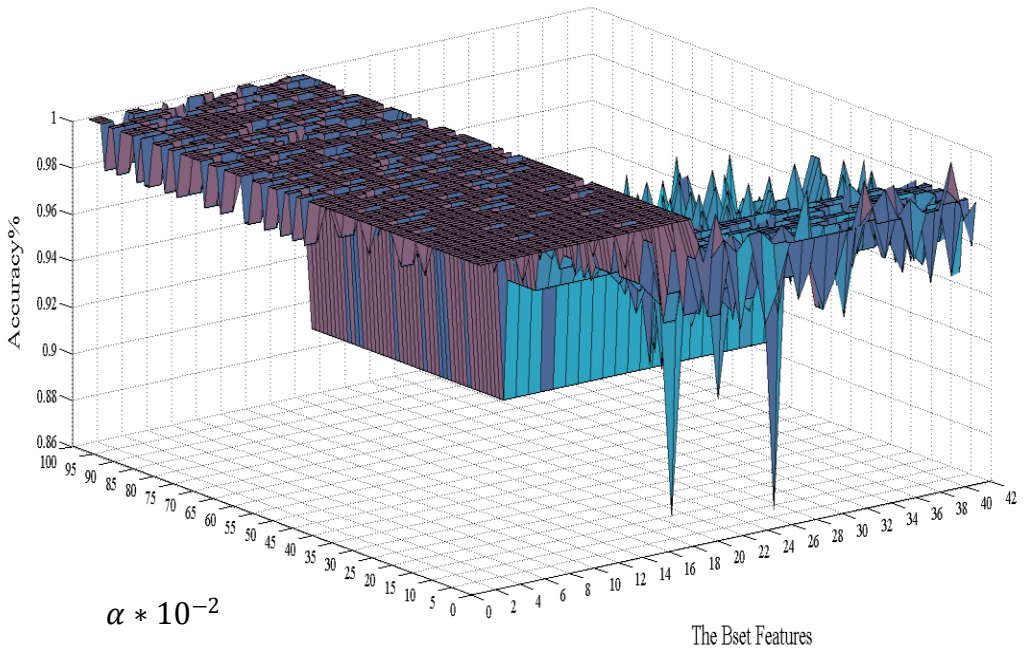


Figure 6.16: U2R Simulation Test - accuracy values against values of the best features and α.

Figure 6.17 shows the best features to gain the maximum accuracy. The value of 100% accuracy is obtained at most of the different features number. The values of

false positive and false negative rate are 0 at the maximum accuracy. These values are an indication of the strength of the algorithms in detecting U2R abnormal behavior categories which are 4 in this test.

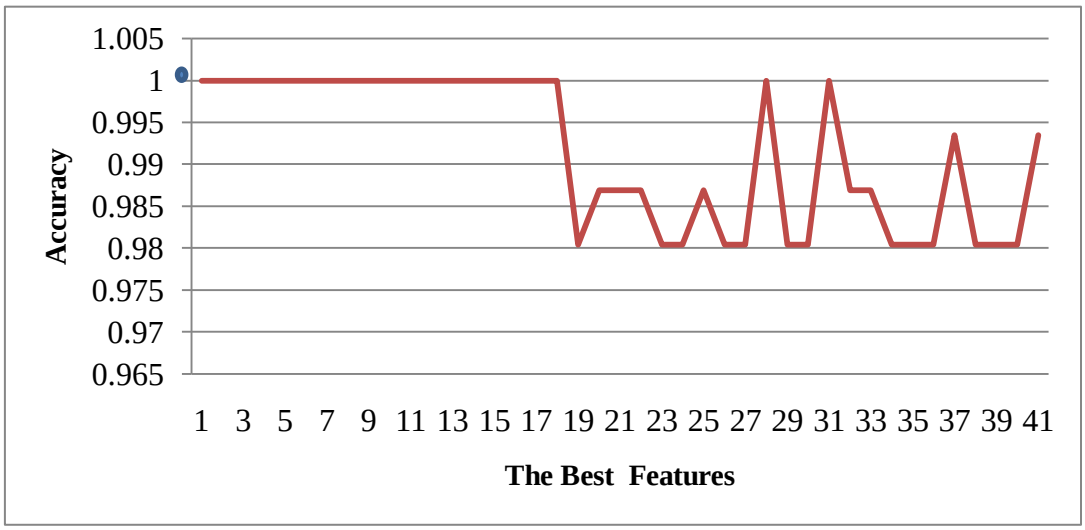


Figure 6.17: U2R Simulation Test -the accuracy values against values of the best feature.

In the second run, the change of accuracy at different values of α is demonstrated at Figure 6.18. The values of accuracy vary from 98.1 to 100%at the fast and the best feature number (1).

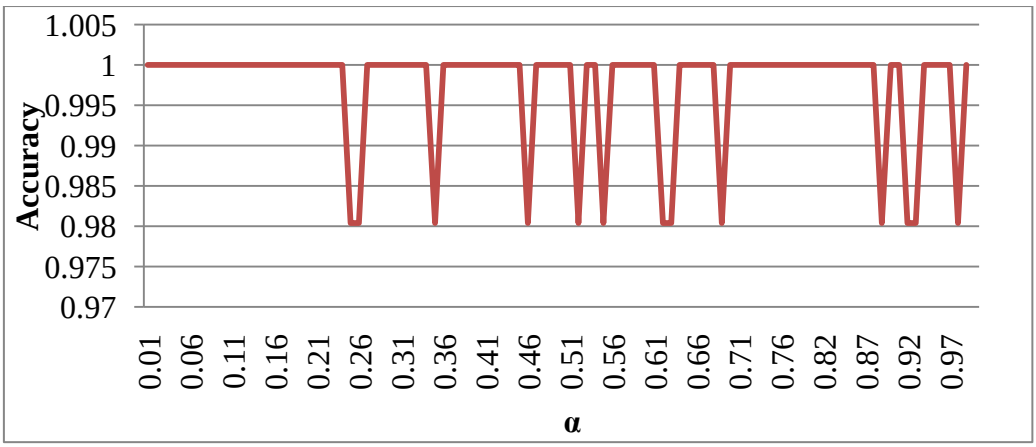


Figure 6.18: U2R Simulation Test - change in accuracy at the best feature against α.

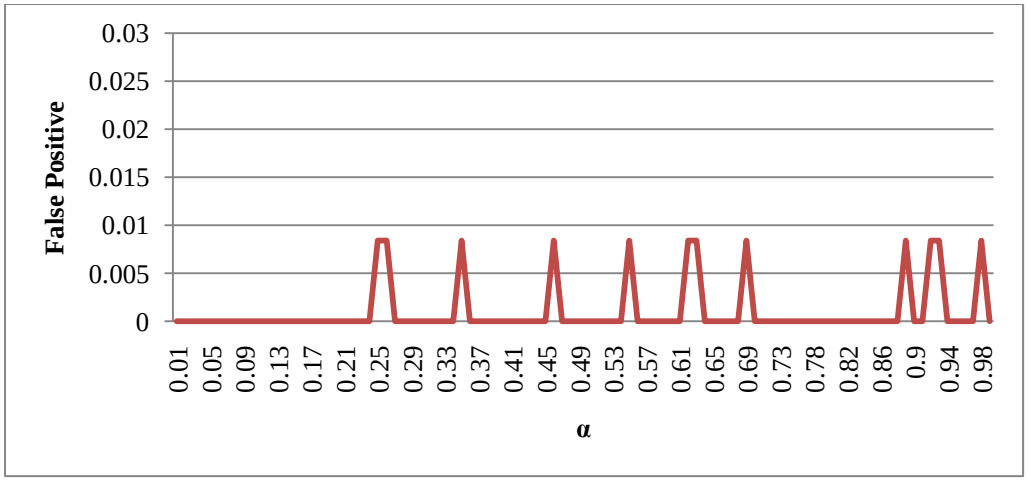


Figure 6.19: U2R Simulation Test - the false positive error values against α.

In Figure 6.19, the values of false positive errorvary from 0 to 0.79% at all values of αat the best feature; but the values false negative vary from 0 to 1.11% at different values of α as is demonstrated at Figure 6.20.

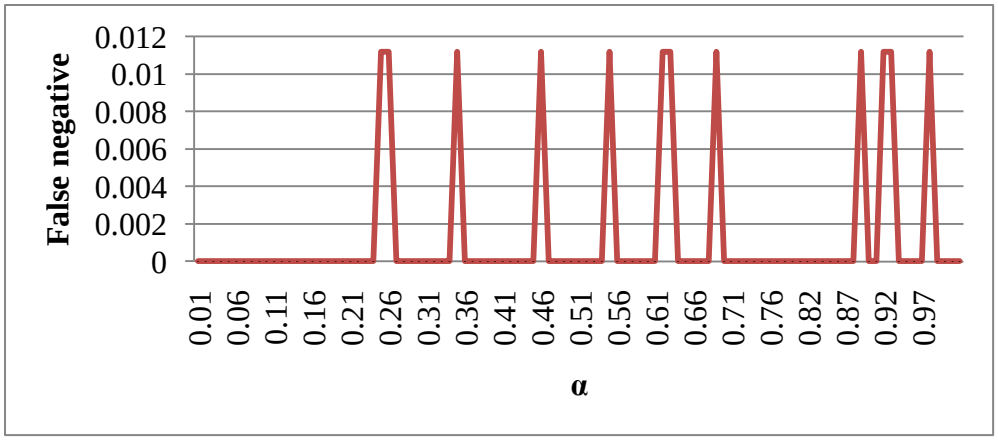


Figure 6.20: U2R Simulation Test - the false negative errors value against α.

From the obtained results of the four tests, the discussion of the comparative study will present in the next chapter.

6.7.2 Validation of IPS and SH Algorithms Using Process Behavior Dataset

The IPS and SH algorithms are validated using process behavior dataset. The main aim of this simulation is to validate that the system can be implemented at the host level i.e. host-based intrusion prevention system. This simulation process verifies that

the proposed system is a hybrid and can protect network system, as well as host system. As in the first simulation tests, the first run was implemented using the sendmail dataset in UNM dataset. The change of accuracy with different system call frequency is demonstrated in Figure 6.21. The results obtained are tabulated in Table 6.5. The maximum accuracy obtained is 98.51%. Compared to the accuracy of simulation results obtained using the KDD Cup dataset, the accuracy results obtained from this simulation are lower. However, this accuracy is still high compared to the accuracy of other systems tested using the process behavior dataset as we will see later in the comparative study in chapter 7. Although the slight decrease in accuracy is due to insufficient dataset records available for training and testing stages, nonetheless the results obtained are much better than the results of previous AIS system tested using the same dataset. Figure 6.21 shows the change of accuracy with α and the best system call frequencies.

Table 6.5: Results of Detection Accuracy Using Process Behavior Dataset

Maximum Accuracy %	TP %	FP %	FN %	TN %	Identified Abnormal behavior
98.51	98.51	0.00	1.49	100.00	2

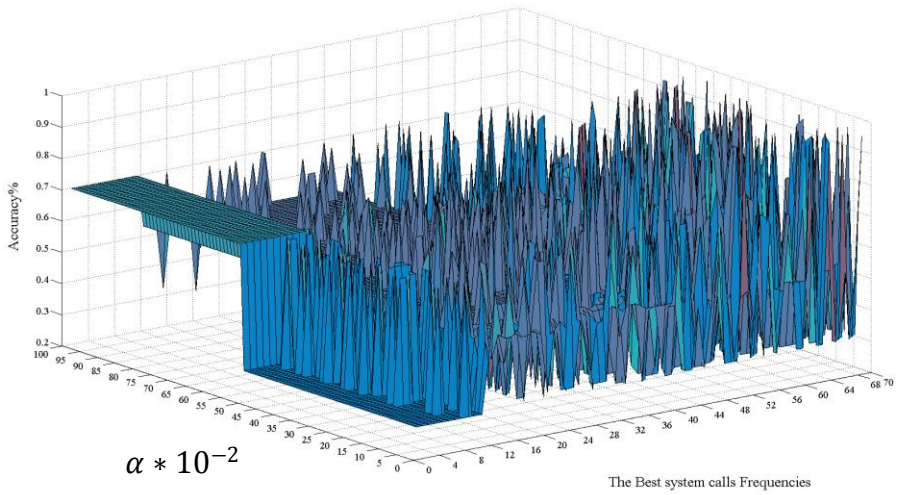


Figure 6.21: Process Behavior Simulation Test - accuracy values against values of system call frequencies and α.

Figure 6.22 shows the change of accuracy with the best system call frequencies.

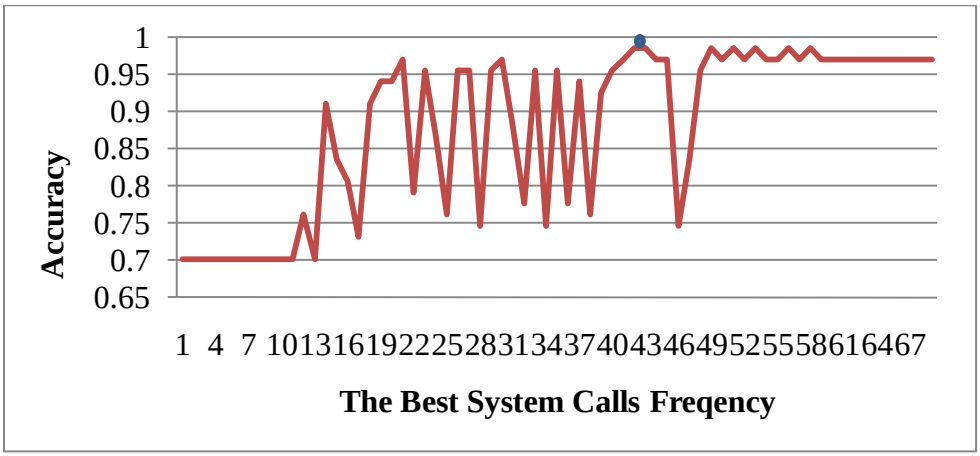


Figure 6.22: Process Behavior Simulation Test - accuracy against system call frequency.

A second simulation test for process behavior of sendmail was run at the best and fast system call number (42) frequency. The change in accuracy at different values of α at the best feature is illustrated in Figure 6.23. The values of the accuracy vary from 74.63 to 98.51% at the best system call number (42) frequency.

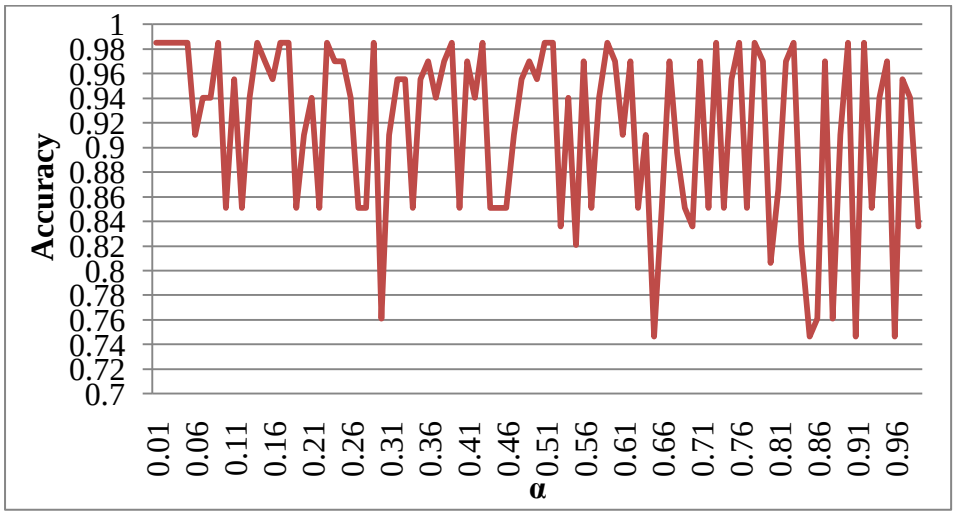


Figure 6.23: Process Behavior Simulation Test -detection accuracy against α.

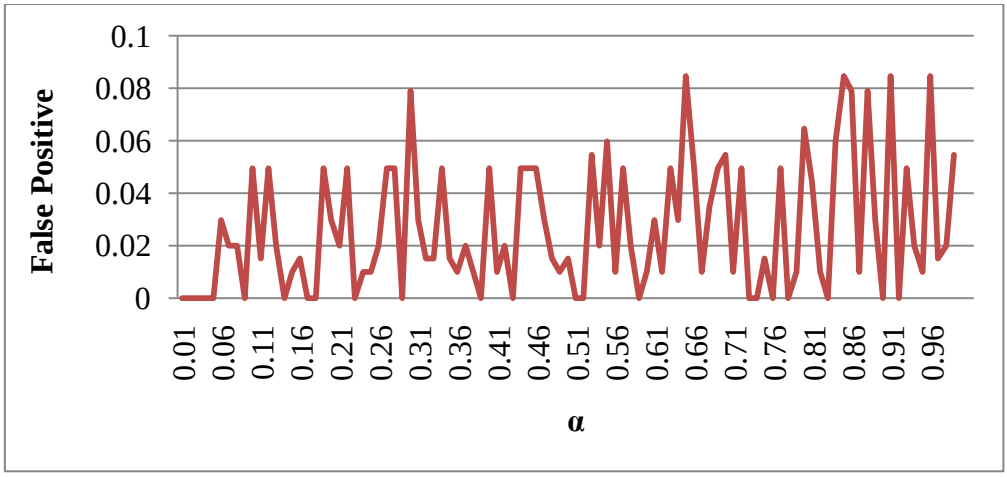


Figure 6.24: Process Behavior Simulation Test -the false positive error values against α.

Change in the values of the false positive error and false negative error with αat the best system call number (42) frequency are shown Figures 6.24 and 6.25 respectively. The values of false positive error vary from 0 to 8.47% while the values of false negative error vary from1.49 to 16.9%.

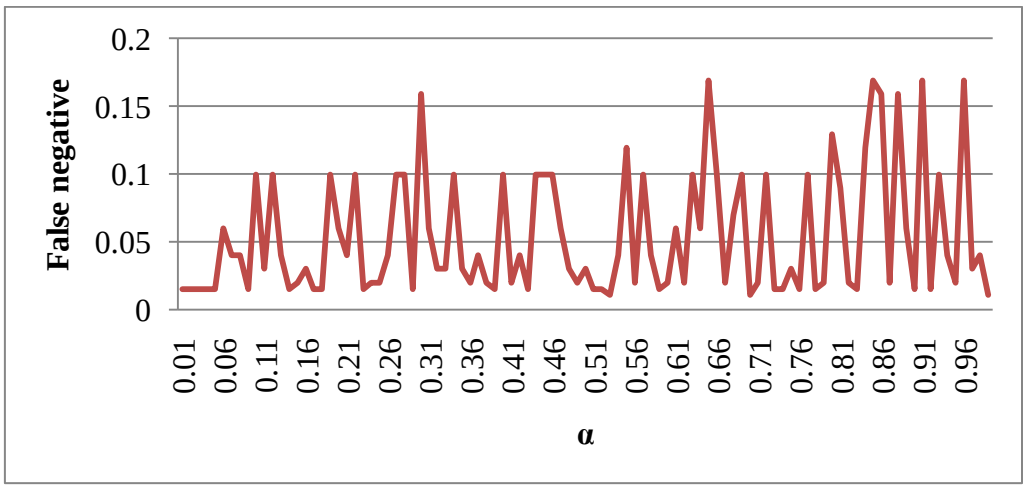


Figure 6.25: Process Behavior Simulation Test - the false negative error values against α.

The result obtained from the five different dataset class tests against the detection metrics are summarized in table 6.6

Table 6.6: Results of Detection Metrics for the Five Dataset Classes

Dataset class	Maximum Accuracy %	TP %	FP %	FN %	TN %	Identified Abnormal behavior
Denial of Services	99.79	99.86	0.07	0.14	99.93	6
Probe	99.92	100.00	0.08	0.00	99.92	4
R2L	99.60	99.6	0.0	0.4	100.0	4
U2R	100.00	100.0	0.00	0.00	100.00	4
Process Behavior	98.51	98.51	0.00	1.49	100.00	2

6.7.3 Simulation Results of Prevention and Analysis

Simulations of the IPS and SH algorithms produce results that can be used to measure the performance of prevention, analysis and adaptation tasks. Since this simulation is performed statically, the metrics for validating the prevention is limited. Only at this stage, the process of the classification time and positive predictive and negative predictive and the throughput values can be measured. The next chapter will discuss the new metrics for prevention predictability of the system and healing system reliability. For the prevention, analysis and adaption, the PP, NP, process mean time, throughput and Cl_{ER} values are calculated for each simulation test. The Cl_{ER} values depend on the error position, sub class representative and subclass number parameters. The results of all these metrics are detailed in Tables 6.7 and 6.8. The results indicate that prevention response will be taken with high predictive values when an intrusion occurs because the values of PP fall in the range between 99.62-100%. Meanwhile, the values of NP are also high, which means there will not be any response when there is no intrusion. The values of NP range from 99.3-100%. These results validate our algorithms at the prevention level since these values are equal or close to 100%.

Table 6.7: Prevention Metrics Results

Dataset category	PPV%	NPV%	Process Mean time sec	Throughput Process/sec
Denial of services	99.62	99. 97	0.000064	15625
Probe	99.70	100.00	0.000096	10358
R2L	100.00	99.90	0.00055	1824
U2R	100.00	100.00	0.00046	21714
Process behavior	100.00	99.30	0.000039	25631

The analysis and adaptation errors are very small compared to the amount of data processed, and ranged between 0.0-1.17%. In R2L test, the Cl_{ER} value is high, which is due to the very small size of available training dataset. The throughput and process time values are significant and show high performance of the system.

Table 6.8: Analysis and Adaptation Error Cl_{ER} Results

Dataset Class	Analysis and Adaptation Error
Denial of Services	0.446 %
Probe	0.00 %
R2L	1.17 %
U2R	0.00 %
Process behavior	0.00 %

6.8 Chapter Summary

This chapter explores the important and traditional metrics required to validate the simulation test of the system algorithms. The limitation and assumptions are clarified. The datasets used in the simulation test are identified and discussed. The mathematical equation and definition of the required metrics are explained in details.

Two standard network intrusion detection and prevention datasets have been used in the simulation test i.e. KDD Cup and UNM. The simulation results show that the system gives high detection accuracy with high performance in prevention, analysis and adaptation. The using of the diverse dataset in simulation tests validate that the system is hybrid i.e. can work as host-based or network-based IPS system. Meanwhile, the presented system can detect misuse and anomaly abnormal behavior with high accuracy and lower false errors.

The new metric for prevention response trust and how to keep tracking the intrusion prevention system with the Self-healing system will be addressed in the next chapter. In addition, the results obtained from the simulation tests will be compared with the results from other AIS systems for intrusion detection and prevention algorithms and other types of intrusion prevention systems.

CHAPTER 7
BENCHMARKING AND COMPARATIVE STUDIES

7.1 Chapter Overview

The new IPS and self-healing system algorithms need be more reliable than those from previous related researches and studies. In this chapter, the results achieved from the simulation tests in chapter 6 are discussed using new metrics for detection capabilities, predictability trust of the prevention responses and self-healing reliability. The comparative studies undertaken in this work are considered from two different views. The first view is a quantitative comparison among bio inspired IPS systems. System accuracy, false positive and false negative errors are the most traditional and important issues in quantitative comparison when developers and researchers introduce a new IPS algorithm. The second view is to compare the bio inspired IPS and SH design features with other AIS systems for intrusion detection and prevention. The combination of SH with IPS, and how it can be used to track a network system to ensure the security and survival of network systems for critical services are discussed. Finally, the advantages and disadvantages of deploying IPS strategies are discussed.

7.2 Capability of Intrusion Detection - New Benchmarking Metric

In [129], the authors provided an in-depth analysis of existing IPS metrics. In addition, they also provided a new information-theoretic analysis of IPS and proposed a new metric. They proposed how to examine the intrusion detection process from an information-theoretic point of view; they have less ambiguity about the input event data given the IPS output data classification. Thus, their new metric is *Capability of Intrusion Detection* C_{ID}. $\boldsymbol{C_{ID}}$ is the ratio of shared information between IPS input and output to the entropy of the input. $\boldsymbol{C_{ID}}$ has the most wanted features that:

1- It considers all the significant facets of detection capability naturally, i.e. true positive rate, false positive rate, false negative rate and positive predictive value.

2- It logically provides an essential measure of intrusion detection capability.

3- It is receptive to IPS operation parameters such as true positive rate and false positive rate, which can exhibit the effect of delicate changes of an intrusion detection and prevention system.

C_{ID} is proposed as a suitable performance measure to make the most of fine-tuning an IPS. The operation point obtained after fine-tuning is the best that can be achieved by IPS in terms of its essential capability to categorize input data. To measure and show the effect of C_{ID}, numerical examples and experiments of actual IPSs on various datasets are established using C_{ID}. The optimal operating point for IPS are chosen and objectively contrasted from IPSs.

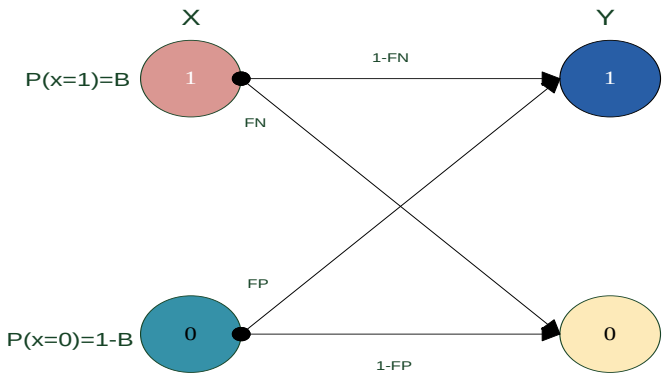


Figure 7.1: Abstract Model of IPS Input/Output at Detection Stage [129]

Generally, the IPS model takes the input as random variable X, where X = 1 represents an intrusion, and X = 0 represents normal traffic. Meanwhile, the output of an IPS is modeled as a random variable Y, where Y = 1 indicates an alert of an intrusion, and Y = 0 represents no alert from the IPS. The assumption here is: there is an IPS output result matching to each input. The exact encoding of X, Y is related to the unit of the input data stream, which is in fact related to unit of analysis. Most of the network security systems consider the unit of analysis as a packet. The abnormal behavior packets are predetermined as X = 1. The IPS inspects each packet to classify it as abnormal behavior Y = 1 or normal behavior Y = 0. From the point of view of

detection input/output, we can construct an abstract model of IPS input and output as shown in Figure 7.1.The abstract model shows that:

- Preceding probability of intrusion, the input event data is inspected by IPS: $P(X = 1)$is the base rate and is denoted by *B*.

- An intrusion event has a probability:

$P(Y = 0|X = 1)$, if considered normal behavior by the IPS. This is the false negative rate *FN* is denoted as β.

- A normal behavior has a probability:

$P(Y = 1|X = 0)$ of being misclassified as an intrusion. This is the false positive denote by δ.

To analyze the intrusion detection model using information theory, the declared model is practically helpful [129]. The metric of C_{ID} is based upon input/output model and information theory. First, we need to define some terms of the information theory and theory itself.

In self information theory, the entropy or uncertainty of random variable *X* is given by:

$$\mathrm{H(X)} = -\sum_{\mathrm{x}} \mathrm{p(x)log\ p(x)} \qquad (7.1)$$

The larger $\mathrm{H(X)}$ indicates a larger uncertain of X.

If $\mathrm{(X,Y)}$ is distributed together as $\mathrm{p(x,y)}$, the conditional entropy of $\mathrm{H(X|Y)}$is declared as:

$$\mathrm{H(X|Y)} = -\sum_{\mathrm{y}} \sum_{\mathrm{x}} \mathrm{p(x,y)log\ p(x|y)} \qquad (7.2)$$

$\mathrm{H(X|Y)} = 0$ if and only if the value of X is completely determined by the value of Y.

Conversely, $\mathrm{H(X|Y)} = \mathrm{H(X)}$ if and only if X and Y are completely independent.

Conditional entropy $\mathrm{H(X|Y)}$has the following property:

$$0 \leq H(X|Y) \leq H(X)$$

Consider two random variables X and Y with a joint probability mass function $p(x; y)$ and marginal probability mass functions $p(x)$ and $p(y)$. The mutual information $I(X; Y)$ is [135]:

$$I(X; Y) = \sum_x \sum_y p(x, y) \log \frac{p(x,y)}{p(x)p(y)} \quad (7.3)$$

Then the mutual information and entropy:

$$I(X; Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) \quad (7.4)$$

Guofei Gu et al. [129] defined Intrusion Detection Capability metrics C_{ID}, which is simply the ratio of mutual information between IPS input and output to the entropy of the input.

$$C_{ID} = \frac{I(X;Y)}{H(X)} \quad (7.5)$$

For a realistic low base rate, C_{ID} is more sensitive to changes in δ i.e. false positive rate than changes in β i.e. false negative rate.

In our study, we have considered both traditional and C_{ID} metrics in the evaluation of the proposed system. The parameters of the classification algorithms, which are used in the simulation tests and mentioned in chapter 6, are tuned to optimize the accuracy of classification between normal and abnormal behavior that automatically affect the metrics of the false positive, false negative, true positive and detection capabilities. Moreover, the window of detection is one of the aspects used as performance metrics.

The new metrics were used to measure the detection capability for all different simulation runs and the results are shown in Table 7.1. These metrics are sensitive and are affected by both the false positive rate and false negative rate. As C_{ID} is close to =1 for DoS, Probe, R2L and Process behavior tests, this shows that the system detection capability is perfect. The perfect result was obtained when the system was used to detect the U2R abnormal behavior. The other results, indicate that the system is perfect since the min value for detection capability is 0.975 which is also an

excellent result as the value of C_{ID} close to 1. The details of calculating this metrics are explained in Appendix C.

Table 7.1: Detection Capability Results

Test Dataset Class	The Detection Capability
Denial of service	0.990
Probe	0.993
R2L	0.991
U2R	1.00
Process Behavior	0.975

7.3 Prevention Response and Self-healing Event Tracking - New Benchmarking Metric

The main responsibility of IPS is to investigate whether to prevent the input data or to permit them into the network system; it is necessary for the system to have high predictability trust to be sure that the data can be prevented if they are 100% abnormal behavior. In the case that the prevention response is not trusted, the healing must be triggered. Using this philosophy, the IPS is constantly keeping track with the self-healing system. Figure 7.2 illustrates how the IPS keeps track with SH. The threshold values are an important value in this case. Two new methods to calculate the metric for prevention predictability trust coefficient, which depends on the nearest centroid in the abnormal behavior subclasses of the classes to the input data (e.g. data packet), are introduced in the classification algorithms of the proposed IPS system. This feature is mapped from the immune system when the DC reaches maturation affinity to bind the pathogen and to identify types of pathogens by T-helper through the antigen types then co-stimulate the T-killer.

The affinity binding is mapped to the nearest distance between specific features of a specific subclass belonging to the class of abnormal behavior i.e. bind to the suitable class features.

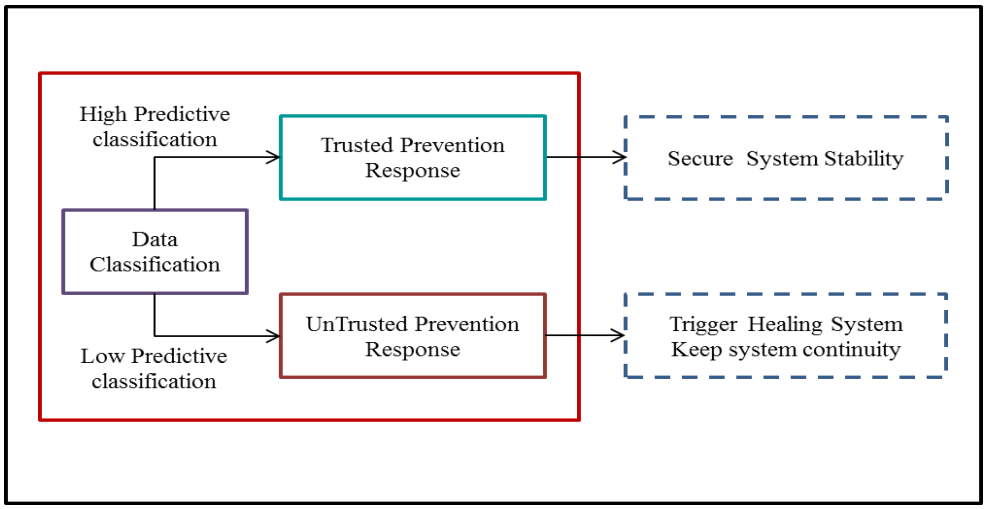


Figure 7.2: Scheme of Trusted Data Classification to Ensure System Stability and Continuity

The first method uses all the distances between the input data and the representatives of each class as exhibited in Figure 7.3. The value of the Prevention Predictability Coefficient (PPC) can be calculated from equation (7.6):

$$PPC = 1 - \left(\frac{\left(\frac{1}{\sum_{k=1}^{i} \frac{1}{d_i}} \right)}{d_{i+1}} \right) \qquad (7.6)$$

where;

$d_1 < d_2 < \cdots < d_i < d_{i+1}$,

$d_1, d_2, \ldots, d_i$ is the smallest distance from the same class,

d_{i+1} is the first smallest distance from the other class.

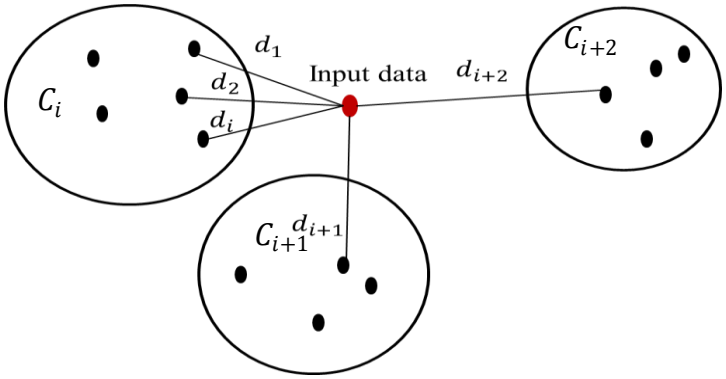


Figure 7.3: First Method to Calculate Prevention Predictability Trust Coefficient (PPC)

The second method depends only on the two nearest representative data to the input data as exhibited in Figure 7.4. The value of the Prevention Predictability Coefficient (PPC) can be calculated from equation 7.7:

$$\text{PPC} = 1 - \frac{d_1}{d_{i+1}} \tag{7.7}$$

where;

$$d_1 < d_{i+1}$$

d_1 is the smallest distance ,

d_{i+1} is the first smallest distance from different class other than d_1.

Both methods of calculating the Prevention Predictability Coefficient are used in the simulation.

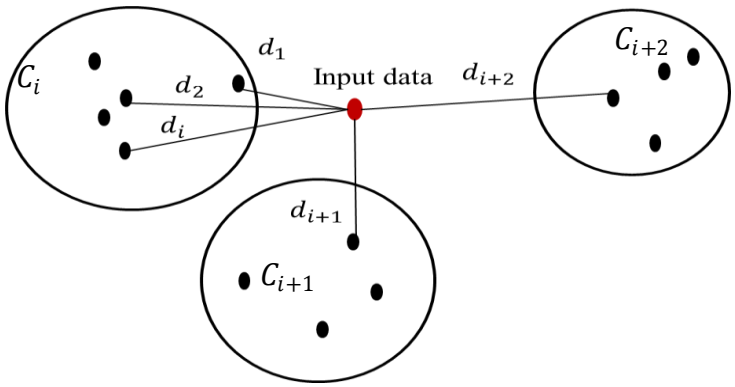


Figure 7.4: Second Method to Calculate Prevention Predictability Trust Coefficient (PPC)

7.3.1 Prevention Predictability Validation

The predictability trust is observed and calculated to estimate the threshold and affinity to get the confidence prevention response when an input data is classified as an intrusion. The value of threshold and predictably trust for each process is assessed by equations (7.6) and (7.7).

The Prevention Predictability trusts Coefficient and thresholds for four datasets of KDD Cup classes and process behavior dataset are demonstrated for each dataset test individually. There might be some error positions in classification i.e. data record that has been incorrectly classified. In all of the figures, these are marked with red points.

Figure 7.5 shows the Prevention Predictability trust Coefficient for each process data in the denial of services dataset test. In Figure 7.5, method 1 was used for PPC estimation. The simulation results show three error positions, two of them are false negative error and one is false positive error. The threshold of the trusted value is 0.8 i.e. the data processed above the threshold are mostly predictability trust values, which give confidence to the decision be it to prevent or to permit the data. For the data below 0.8, the self-healing system must to be triggered to make correction for classification if any misclassification is captured.

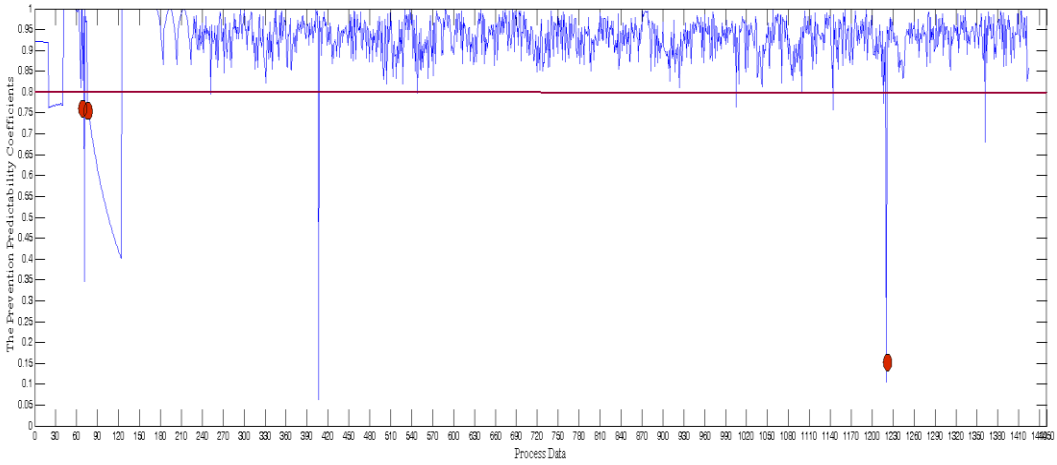


Figure 7.5 The Preventive Predictability Trust Coefficient (PPC) for the DoS data processed using method 1.

In Figure 7.6, method 2 was used for PPC estimation. The simulation results show three error positions, two of them are false negative error and one is false positive error. The threshold of the trusted value is 0.65 i.e. the data processed above the threshold are mostly predictability trust values. The numbers of data that fall below 0.8using method 2 are much more than the numbers of data using method 1 in the same range. This evaluation for the PPC confirms that method 1 gives a better estimation of the prevention response trustability.

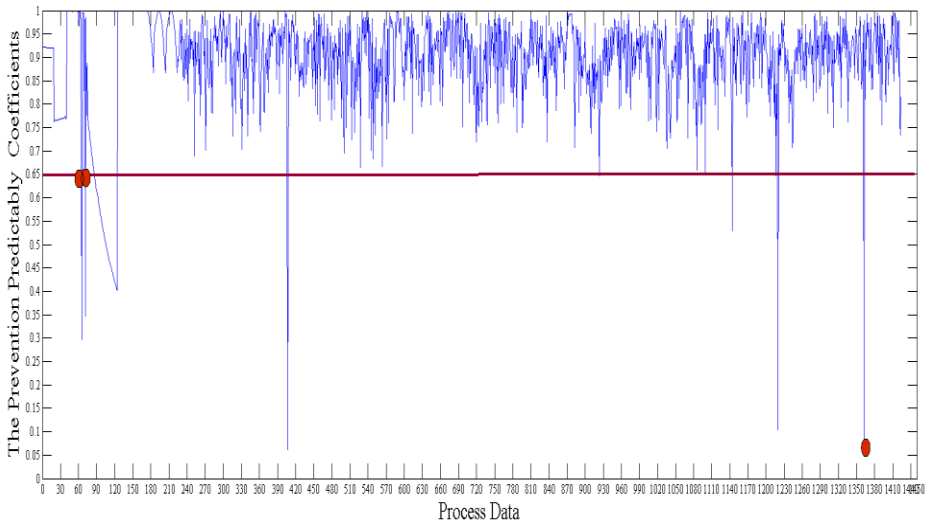


Figure 7.6 The Preventive Predictability Trust Coefficient (PPC) for the DoS data processed using method 2.

Figures 7.7 and 7.8 show the Prevention Predictability trust Coefficient for each process data in the probe dataset test. Figure 7.7 used method 1 for PPC estimation. The simulation results show there is only one error, which is false positive error. The threshold of the trusted value is 0.85 i.e. the data processed above the threshold are mostly predictability trust values. For Data below 0.85, the self-healing system must to be triggered to make correction for classification if any misclassification is captured.

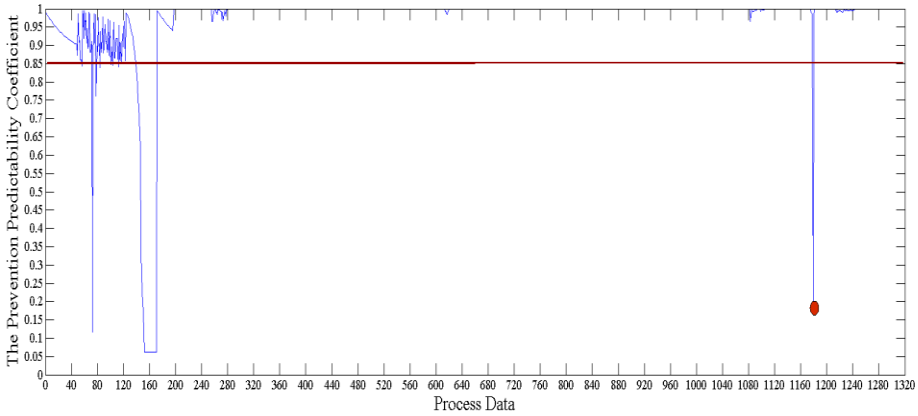


Figure 7.7 The Preventive Predictability Trust Coefficient (PPC) for the probe dataset processed using method 1.

Method 2 has been used to estimate the PPC for probe processed dataset in Figure 7.8. The simulation results show only one error, which is false positive error. The threshold of the trusted value is 0.70, indicating that the data processed above the threshold are mostly predictably trust values. For Data below 0.7, the self-healing system must to be triggered to make correction for classification if any misclassification is captured. The data that fall below 0.85 are much more according to method 2 than method 1in the same range. This evaluation of PPC of the probe dataset gives another confirmation that method 1 is better for estimation of the prevention response trustability.

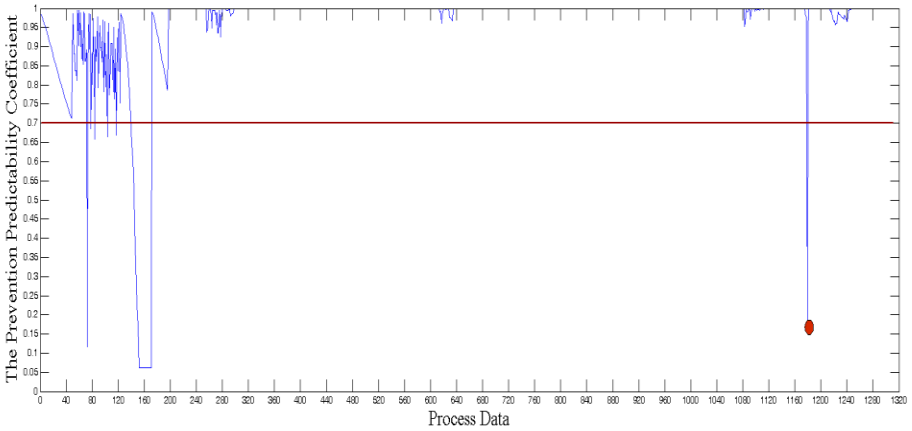


Figure 7.8 The Preventive Predictability Trust Coefficient (PPC) for the probe dataset processed using method 2.

Figures 7.9 and 7.10 show the Prevention Predictability trust Coefficient for each process data in the R2L dataset test. Method 1 has been used for PPC estimation in Figure 7.9. Two error positions are shown by the simulation results, one is false negative error and the other is false positive error. The threshold of the trusted values is 0.9 i.e. the data processed above the threshold are mostly predictability trust values. For Data below 0.9 the self-healing system must to be triggered to make correction for classification if any misclassification is captured.

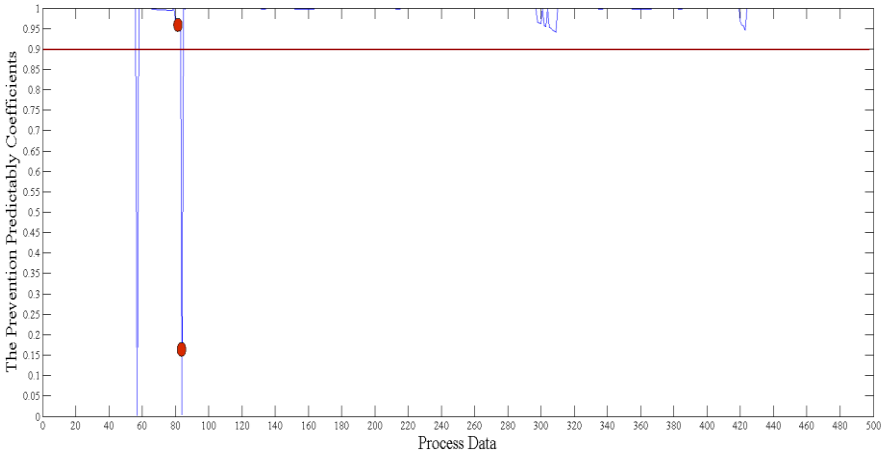


Figure 7.9 The Preventive Predictability trust Coefficient (PPC) for the R2L dataset processed using method 1.

Method 2 has been used for PPC estimation for R2L in Figure 7.10. Three error positions are shown by the simulation results, two are false negative error and one is false positive error. The threshold of the trusted value is 0.77. For Data below 0.77 the self-healing system must to be triggered to make correction for classification if any misclassification is captured. The data that fall below 0.9 are much more according to method 2 than method 1 in the same range. This PPC evaluation also confirms that in here, method 1 is more accurate to estimate the prevention response trustability.

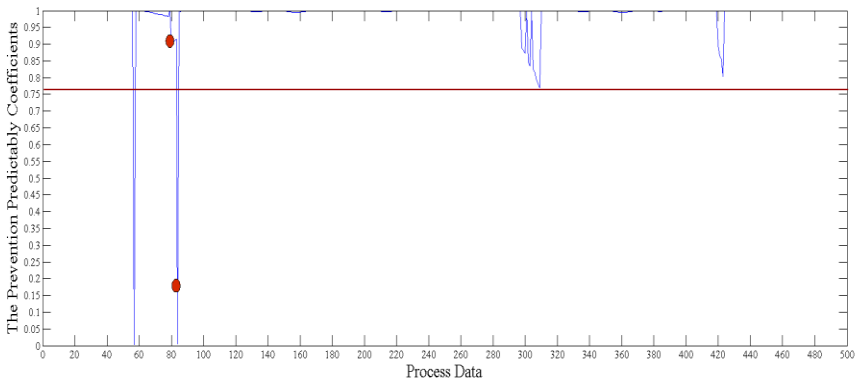


Figure 7.10The Preventive Predictability Trust Coefficient (PPC) for the R2L data processed using method 2.

Figures 7.11 and 7.12 shows the Preventive Predictability trust coefficients for each process data in U2R dataset test. For PPC estimation in Figure 7.11, method 1 has been used. The simulation results show that there is no error. The threshold of the trusted value is 1.0 i.e. all data classification are predictably trust values.

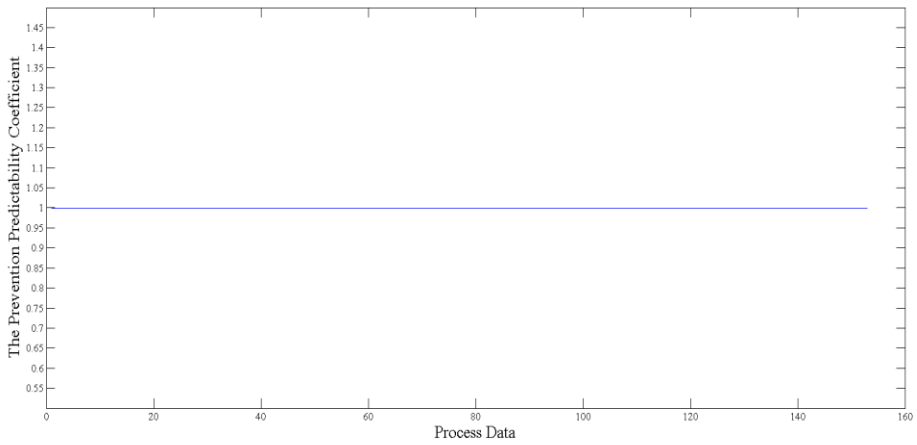


Figure 7.11: The Preventive Predictability Trust Coefficient (PPC) for theU2R dataset processed using method 1 and 2.

For Figure 7.12, method 2 has been used for PPC estimation and again the simulation results show that there is no error. The threshold of the trusted value is 1.0 i.e. the data processed are 100% predictability trust values. This evaluation of the PPC confirms that both methods give accurate estimation of the prevention response trustability.

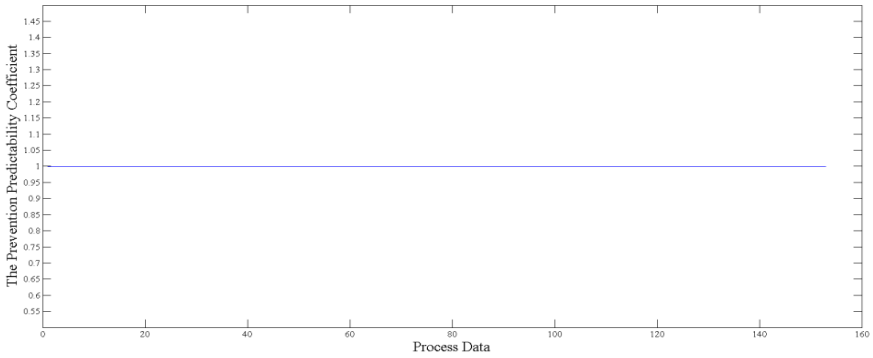


Figure 7.12: The Preventive Predictability Trust Coefficient (PPC) for the U2Rdataset processed using method 2.

The result of the Prevention Predictability trust Coefficient (PPC) is obtained from the process behavior dataset test. When PPC is estimated using the first method, the value of the threshold is 0.55 as shown in Figure 7.13. The processed data has a lower threshold compared to the other results. In the test, values of untrusted Prevention Predictability trust Coefficients are high when the second method was used. This untrusted result is due to two reasons:

1. The training data available are not sufficient.
2. The method of using the system calls sequences and frequencies as features to classify process behavior are not perfect to train agent for prevention, thus requiring some improvements.

Although the result obtained show some weaknesses in the prevention trust of process abnormal behavior, still the detection accuracy is better than previous algorithms as will be explained in section 7.5, which discusses the comparative studies. For Data below 0.55, the self-healing system must to be triggered to make correction for classification if any misclassification is captured.

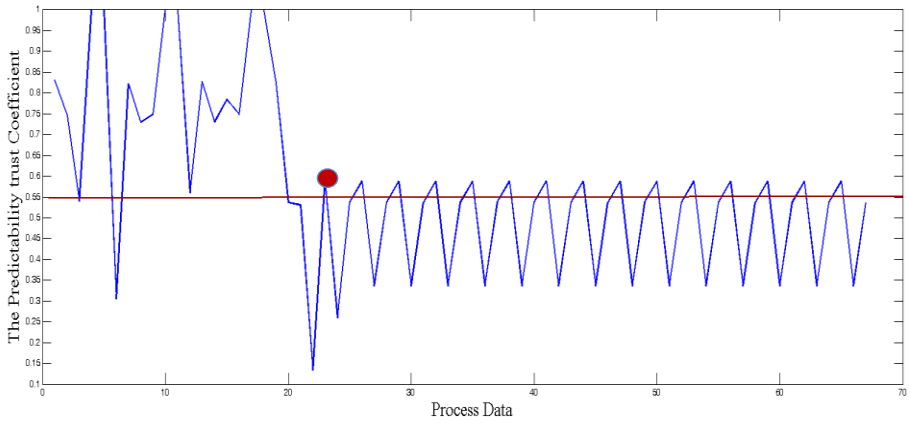


Figure 7.13 The Preventive Predictability Trust Coefficient (PPC) for the data processed using method 1.

When using method 2, the numbers of untrusted values of prevention predictability coefficient are higher than the one used in method 1;the threshold is 0.5 as shown in Figure 7.14. All process data above the threshold can be either prevented or permitted with trust. The data that fall below 0.55 are much more according to method 2 than method 1 in the same range. This PPC evaluation also confirms that, method 1 is more accurate to estimate the prevention response trustability.

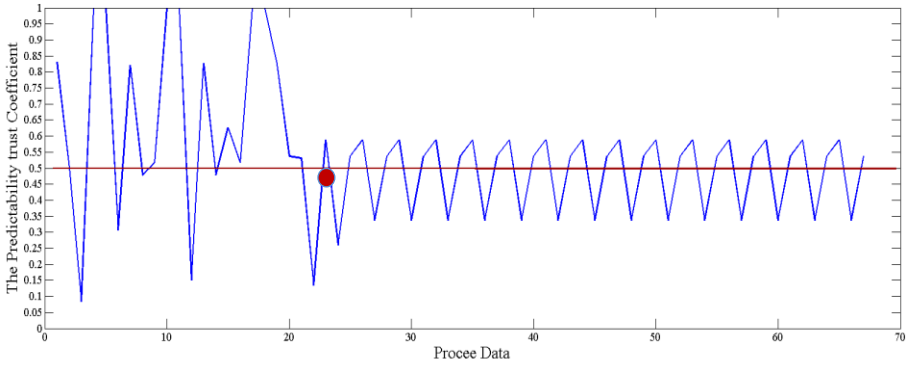


Figure 7.14 The Preventive Predictability Trust Coefficient (PPC) for the data processed using method 2.

7.3.2 Prevention Response Solutions

There is a diversity of techniques for responding to intrusion prevention. In deciding the appropriate prevention response to a detected intrusion, there are a number of criteria that must be considered to determine an appropriate response.

These criteria are: timing, type of attacker, type of attack, implications of the attack, degree of suspicion, and environmental constraint i.e. legal, ethical, institutional, and resources [136]. The developers of the IPS system must specify the technique needed to precede the prevention events. The prevention techniques mostly used by IPS developers are:

Disconnection of the source of attack from the network: For network-based attacks, disconnecting from the network is less draconian than shutting down the host, but it has the same effect. Network-based attacks can no longer affect the system, thus allowing the system at the training stage the appropriate time to respond to an attack.

Disabling the destination of attacking ports or services: If a single service or a port is subjected to an abnormal behavior, either the service or the port will be disabled. This will effectively stop the attack without affecting any of the other services offered by the system, here the system can use the feature of the port number to response activate the prevention response.

Block IP address: If the IP address of an attacking system can be identified from the features of the data packet, some network attacks can be neutralized by blocking, at a router, all traffics from that address. While this protection is often temporary if the attacker can change their IP address. The intelligent agent at the training stage can learn how to block the IP address in a significant time.

Termination of user session: If a user is involved in intrusive behavior, the user's session should be terminated and the user's account locked to prevent future damage.

7.4 Self–healing Evaluation Test

The self-healing system is triggered when the predictability trust values for prevention or permission of input data are below the threshold values i.e. untrusted to either prevent or permit. Since self-healing system has been the highlight during the recent years, up to date, few researchers have introduced evaluation metrics for self-healing performance. The mechanism of the self-healing system introduced in this research requires quantification of its efficacy and reason about trade-offs. In [66],

[67], [137], the researchers have discussed and introduced the evaluation of self-healing benchmark and reliable metrics.

To address this issue in our research, the problem of network system failure is simplified. The behavior of health system components are kept in a database and replicated three times as explained in chapter 5 section 5.6 in self-healing mechanisms. Reliability, availability and serviceability of the three copies of the database are used for the evaluation metrics. To present a metric, firstly failure injection to one component or more is needed; secondly, either the successful repair or failure of healing must be measured using mathematical modeling. Finally, the metric is derived from the point of views of reliability, availability and serviceability of the system.

Reliability: The injected failure in the original system component is replaced and confirmed by the two replicated databases and failure repair is successful.

Availability: The three databases are available and no failure in the databases themselves.

Serviceability: The continuity of the network system is maintained since the self-healing system is triggered within the essential time.

In [138], the metrics derived from Continuous Time Markov Chains (CTMCs) was introduced. These metrics are:

1. Limiting/steady-state availability.
2. Repair success rates (fault-coverage).
3. Repair times.
4. Yearly downtime.

Due to the limitation of this research, which is static evaluation of IPS and SH system, only the first three of these metrics were considered when the self-healing system simulation test was being established. The fourth metric was not considered because it needs real time measurement. Instead of taking downtime, randomization of rejecting the failure in the original database components is used periodically.

The algorithm of the self-healing system has four check states as illustrated in Figure 7.15.

From Figure 7.15:

O: Original database;

R1: First replication of the database;

R2: Second replication of the database;

C1: Check the steadiness of the system components;

C2: Check the availability of the system;

Rs: Repair time;

C3: Repair success.

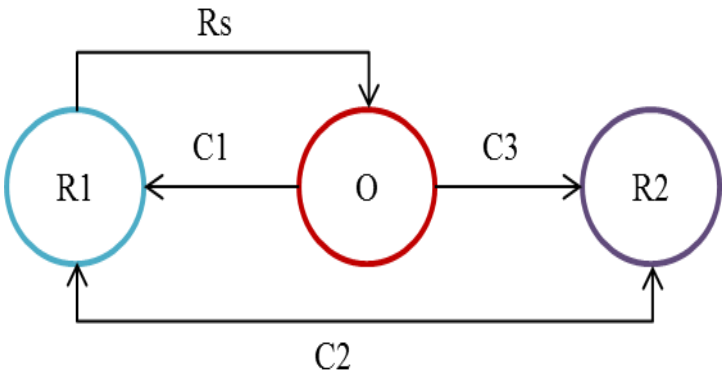


Figure 7.15: The States of Self-healing System Metrics Measurement

From these components of evaluation, the metrics is introduced and the simulation test is executed. The grid network database published in [139] is used and tuned to take the failure of the CPU average usage time, used memory, required time, waiting time and run time as measurable components. 2826 random records for the different processes are used in the test. A sample of this dataset is shown in Table D3.1 Appendix D. The test was analyzed to real time application by randomizing 100 changes in the original components in the dataset periodically. The total time for keeping the system steady and available, repair time, validation of the repair and repair success percentage are the benchmarking metrics for evaluation. The state availability time was taken after 100 random changes and repair. The results are tabulated in Table 7.2. The results show that the system has high efficiency based on the benchmarking metrics.

Table 7.2: Self-healing Simulation Test Results

Keep steady-state availability time (sec)	Repair time (sec/process)	Validation of Repair (sec/process)	Repair success (%)	Availability (%)
0.103574	0.000615	0.000239	100	100

This self-healing algorithm used very reliable static real dataset test. The self-healing system is expected to work fairly good in real time, as well as static work, since the analyzing method is mostly the same as in real time work. The system main features are lightweight software and autonomous. The system was not tested on the datasets tested in some other related works because the results obtained would not be comparable since these metrics are new. Moreover, some of the researchers, for example in [140], used the metrics for different types of tests while the current system is more efficient for system repair and availability. The original and replicated datasets must be updated periodically by the network administrator or by adding training features to register the system component update.

7.5 IPS and SH Integration Results

The aim of the integration between IPS and SH is to improve the performance and keep the continuity of the network system. From the results achieved, the accuracy of IPS is recalculated after the SH has been triggered. The keep tracking with SH system is used to correct the error positions of classification which increases the final accuracy of the presented system. The error positions below the threshold are corrected for each test. The improvement in accuracy is significant especially for process behavior. In table 7.3, a comparison between the accuracy before and after keep tracking with SH is detailed for each test.

Table 7.3: The IPS and SH Integration Results

Dataset	Accuracy before trigger SH	Accuracy after Trigger the SH
DoS	99.79%	99.86%
Probe	99.92%	100.00%
R2L	99.60%	99.80%
U2R	100.00%	100.00%
Process behavior	98.51%	100.00%

7.6 Comparative Studies

The comparative study for the IPS and SH system simulation results are constructed based upon three diverse criteria: firstly, between different current tests for validation, secondly between other AIS systems that have used the same standard datasets and finally, between features of the current system that are mapped from the immune system, and those of other artificial immune systems.

The results of the simulation tests are presented in Table 7.4. The important comparison parameters are detection capability (C_{ID}), Preventive Predictability trust coefficients (PPC), classification accuracy, and analysis and adaptation. The U2R test shows the highest results for predictability trust for prevention, detection capabilities and accuracy, which is 100%. The lowest PPC is obtained in process behavior for sendmail dataset and has a threshold of 0.55. The minimum value for analysis and adaptation error is achieved for U2R test, probe test and process behavior test, while this value is highest for R2L test which is 1.17%.

Table 7.4: Simulation Tests Results for IPS and SH System

Dataset	Denial of Services	Probe	R2L	U2R	Process Behavior
Detection Capabilities C_{ID},	0.990	0.993	0.991	1.00	0.975
Predictability Trust threshold value	0.80	0.85	0.90	1.00	0.55
Accuracy	99.79 %	99.92%	99.60%	100.0%	98.51%
Analysis and adaptation error	0.446%	0.00%	1.17 %	0.00%	0.00%

The second comparison is between three AIS systems and algorithms for intrusion detection; the first one is Multi-agent network intrusion active defense model based on immune (IMAAD) demonstrated by [82]. This system used multi immune agent for intrusion detection. This system was tested using the KDD Cup dataset. The second study was done by [141] where the adaptive and innate immune systems are integrated, and they used their own trace data for process behavior on FTP monitoring behavior of wuftpd. The last study was done by Julie.G [140]; she developed a dendritic cell algorithm (DCA) inspired from the innate immune system based on the danger theory concept. The DCA was tested using the same dataset as in [141].

The results are tabulated in Table 7.5 and Table 7.6 for comparison. Table 7.5 shows the comparison among 4 classes of KDD Cup datasets tests. The parameters of comparison are true positive error (TP), false positive error (FP) and number of abnormal behavior categories plus the normal category. The comparison is between current system and the system presented in [82]. Most of the bio inspired immune systems for intrusion detection suffers from the rate of false positive, so these parameters are chosen to show the improvement in the false positive rates. The results shows a significant improvement in detection accuracy and the false positive errors are reduced to 0.0 in R2L and U2R simulation tests. These results indicate that the combination between AIS and pattern recognition test is efficient.

Table 7.5: Comparison between IMAAD and BIIPSS

	Number of Identified Categories		**TP %**		**FP %**	
Simulation test	IMAAD [82]	BIIPSS	IMAAD [82]	BIIPSS	IMAAD [82]	BIIPSS
Denial of Services	5	7	97.2%	99.86%	2.8%	0.07%
Probe	4	5	96.5%	100.0%	3.5%	0.08%
R2L	3	5	95.2%	99.6%	4.8%	0.00%
U2R	4	5	94.5%	100.0%	5.5%	0.00%

Results of the current system are also compared with [140], [141], and the comparison is established in Table 7.6. The results of [140], [141] are chosen for comparison because both works used the concept of danger theory even though the simulation tests were performed for intrusion detection only. Moreover, the two related works validated their algorithm using FTP monitoring behavior of wuftpd session dataset, which is considered as one of the process behavior datasets created by Twycross [141]. The parameters of comparison are true positive error and rate of false positive. Both systems used in the comparison have better results than the early negative selection algorithm that used UNM dataset. The comparison recommends that the current system is better than the system presented in [140], [141], which also ensure that the current system is also better than the techniques used by negative selection. These results strongly suggest that the danger theory concept for intrusion detection and prevention system integrated with pattern recognition classification algorithm is better for detection accuracy, scalability and low positive errors than the earlier concept of negative selection algorithm.

Table 7.6 Comparison between AIS Algorithms and BIIPSS

AIS System	Julie[141]	Twycross [140]	BIIPSS
True positive	1.0	0.75	0.9851
False positive error	0.83	0.15	0.0

Since the current system integrates biological inspired system with machine learning and pattern recognition concept, it is necessary to validate the algorithms against some pattern recognition algorithms and AIS for intrusion detection system. Mahbod et al. [133] used specific arrangement of KDD cup dataset and implemented this arrangement on different intrusion detection dataset. Our system is simulated using the same arrangement for huge records of 21 categories of abnormal behaviors from all classes of attacks. The ratio between normal behaviors to abnormal behaviors is 1:1 i.e. abnormal behavior is not considered as outliers; and the accuracy attained is 98.87% for detection. Figure 7.16 shows the result obtained by [133] and the current system.

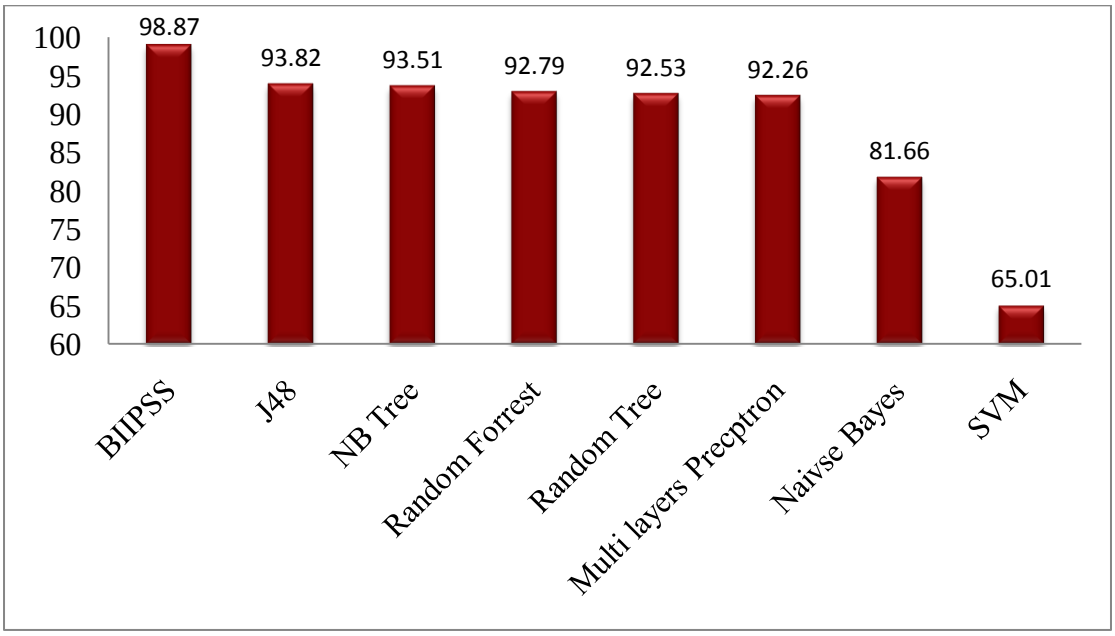


Figure 7.16: Comparison of Accuracy BIIPSS and Machine Learning IDPS Systems.

The last comparative study is between the current system and other AIS systems for IDPS. This comparison is established on the design features and mechanisms inspired from the immune system. This study is summarized in Table 7.7.

Table 7.7Comparison between Design Features and Mechanism of BIIPSS and other AISs

AIS	***DCA* Algorithm [140]**	***TLR* Algorithm [141]**	**Adaptive IPS approach [12]**	**BIIPSS Model**
Adaptive immune system		√	√	√
Innate immune system	√	√	√	√
Knowledge base	√		√	√
Training base		√		√
Prevention mechanism			√	√
Self-healing mechanism				√
Standard antigen database	√	√		√
Standard signal database	√	√		
Processing signal	√		√	
Multilayer system				√
Self adapting	√	√	√	√
Diverse		√	√	√
Distributed			√	√
Autonomy		√		√
Lightweight		√		√
Self-Organized	√	√		√

7.7The IPS and SH System Deployment

The deployment of IPS and SH system is important to ensure the security of network systems. While it is common practice to defend against abnormal behaviors by inspecting traffic at the data centers and corporate headquarters using firewall for example, it is also critical to distribute the network-level defense to stop malicious

traffic close to its entry point, either at the branch or telecommuter offices i.e. the critical services endpoint.

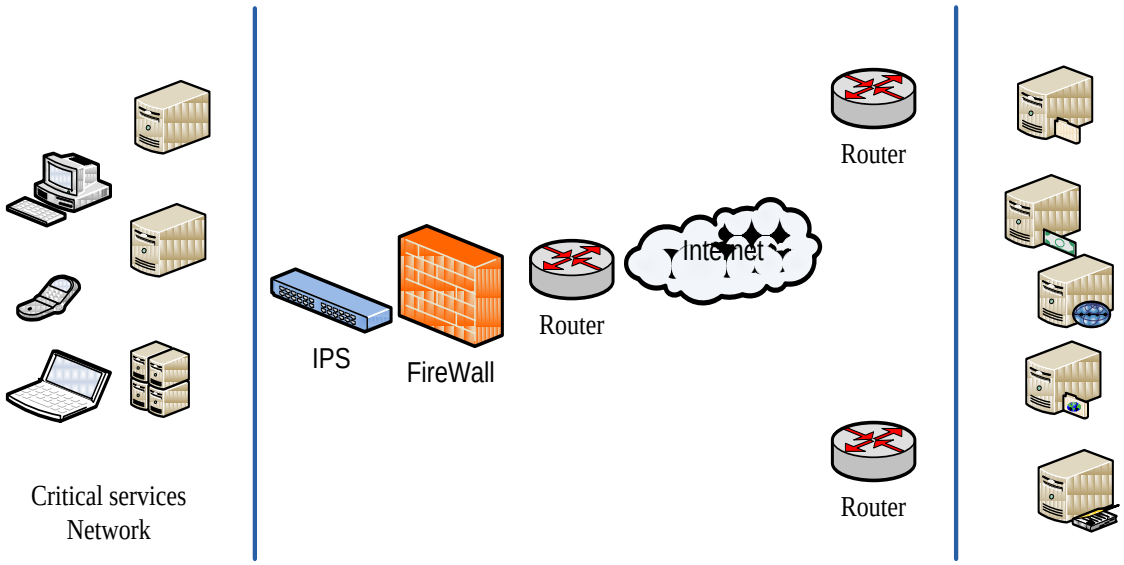


Figure 7.17: The IPS in front of the Firewall

Network engineering designers have different options to choose from for deploying the IPS and SH system in the critical services network system. The deployment is based on many criteria. Basically, IPS deployment depends on the nature of the critical services and the levels of security required by the services, and the vendor of the IPS system. Since the proposed system is a hybrid, the system also can reside in each host for host-based security purposes. Since The SH is integrated with IPS, It follows the same IPS deployment criteria.

However, the network engineering design issue has to be considered when IPS deployment is being established. Mainly four options for network engineering design and critical services network system vendor are available. The designer must tune the network design to optimize the security insurance for the network system.

The first option as shown in Figure 7.18 is to reside the IPS in front of the firewall. This option puts the priority advantages of the firewall function, which is to ensure that traffic policies are enforced i.e. examination rules are configured to allow TCP packets only from a certain source address; the firewall inspects that traffic stream.

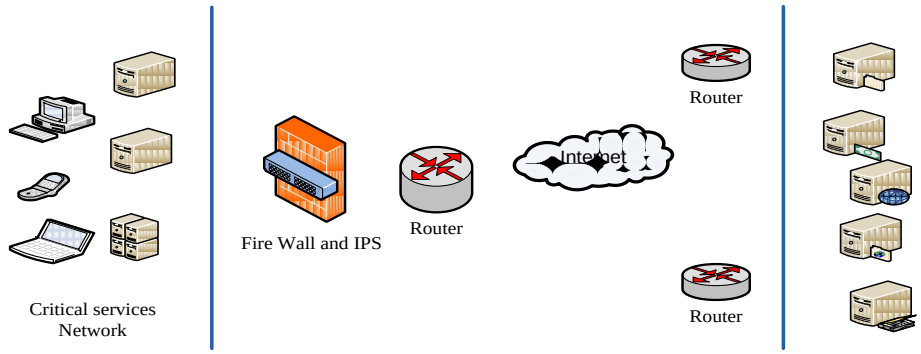


Figure 7.18: The IPS integrated with Firewall

Meanwhile for specific network security reason, the priority is to check the misuse and anomaly abnormal behavior simultaneously. Network engineering developers are suggesting another solution to solve this problem, either by integrating firewall with IPS or by setting the IPS behind the firewall as shown in Figures 7.19 and 7.20. These choices must be discussed between the designer and vendors because they may affect the latency of sending and receiving the data, bandwidth and other network engineering design parameters.

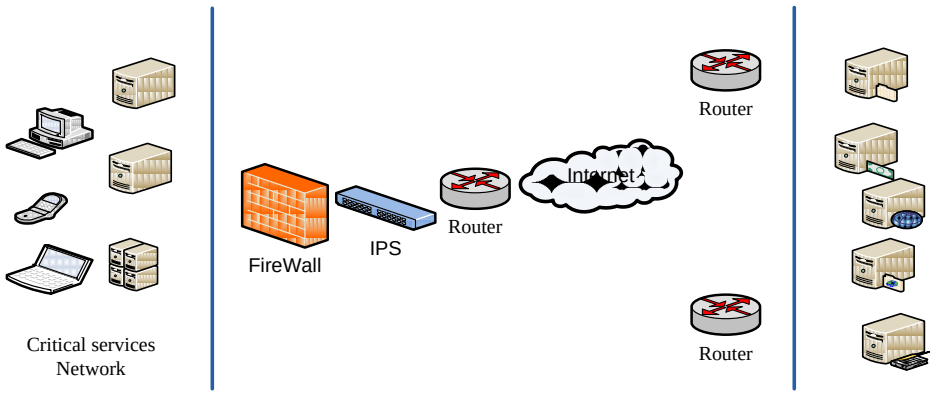


Figure 7.19: The IPS Behind the Firewall.

Other issues in network design are: memory efficient traffic scanning for attack signatures that will consume less memory on the router, capability to provide protection for larger number of common threats, and vulnerabilities which may also include the routing system and for accessing remote the data points. To deal with these issues, network engineering developers integrate the IPS with routing system as is clearly shown in Figure 7.20.

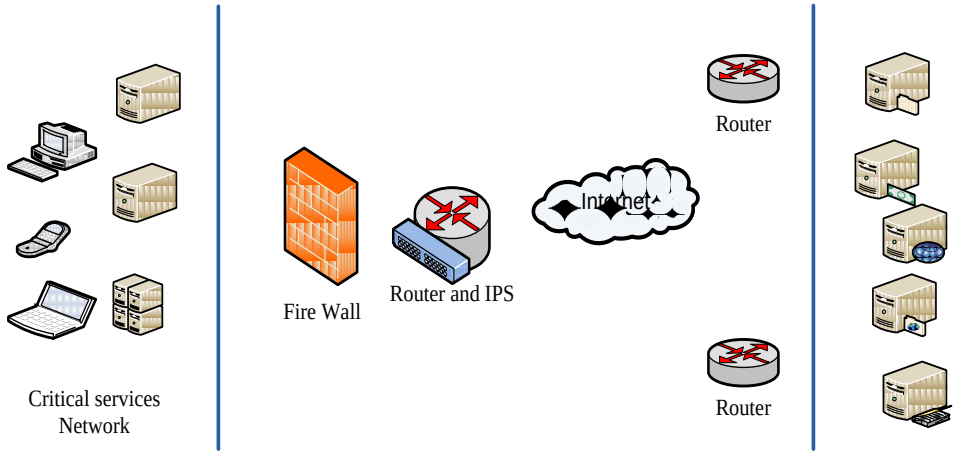


Figure 7.20: The IPS Integrated with Routing System

7.8 Chapter Summary

This chapter introduces new metrics for intrusion classification, Predictability trust for prevention, and self-healing reliability. The simulation tests for the proposed system have shown that the system has high performance for detection, prevention and healing mechanisms for all classes of abnormal behavior. The results also indicate a significant improvement in system accuracy compared to other results obtained by previous related works. The simulation test for process behavior shows significant improvements in reducing the false positive error to 0 while the true positive is value 98.51%. Comparative studies against other systems have also been done.

Using self-healing system measurements, the metrics for prevention predictability response and self-healing reliability are introduced. The simulation results show that the algorithms used in the self-healing simulation test are highly reliable. Many AIS systems that are based on the danger theory have been compared with the proposed system from the point of view of the design features inspired from HIS.

Finally, the deployments of IPS are discussed. The next chapter will present the conclusion, contribution and future work.

CHAPTER 8
CONCLUSIONS, CONTRIBUTIONS AND FUTURE WORK

8.1 Conclusions

This research introduced a new intrusion prevention system for critical services networks that are integrated with self-healing mechanism to keep the systems secure from any intrusion and heal any damages caused by the intrusion. The main objective of this research was to present an autonomous system that reduces the intervention of network administrator i.e. using autonomous intelligent agents for intrusion prevention system. At the same time, the system accuracy must be kept maximized by minimizing the false errors in detection and predictability of prevention responses.

The proposed system was inspired from human immune system based on danger theory. The second generation of artificial immune system inspired from the mechanism of innate immune system, which triggered when a danger signals captured by the innate immune cell mainly dendritic cell. Thus for the proposed system, the abstraction from the immune system is inspired from the integration of adaptive and innate mechanisms.

The classification of data used pattern recognition and machine learning algorithms. The results obtained showed that the combination between abstractions from immune system and machine learning algorithm improves the intrusion detection and prevention algorithm. The false errors were decreased notably, reaching zero errors in the U2R abnormal behavior class test. The use of machine learning satisfies the requirement of autonomous system.

To eliminate the defect caused by intrusion, the proposed system includes a self-healing system that keeps tracking damages caused by intruders when the predictability of the prevention is untrusted. In addition, the self-healing system has

the ability to investigate periodically the system components to preserve the system continuity and survivability. The combination between intrusion prevention and self-healing system has been successfully achieved.

8.2 Research Objectives and Achievement Evaluation

The research problem raised in this work was to face susceptible intrusions by intelligent and dynamic abnormal activities in the network system, particularly for critical services. The main objectives of this work were specified at the beginning of the thesis are:

- To develop an autonomous mechanism for intrusion prevention system that is effective for anomaly detection and prevention, based on artificial immune system and pattern recognition.

- To design a network security system that combines the intrusion prevention system with self-healing mechanism.

- To simulate the model for efficiency and robustness, and compare and contrast it with existing security models.

An intelligent and highly accurate autonomous intrusion prevention system that is capable of ensuring secure network systems for critical services has been developed. The objective has been further extended to include a new layer for robust continuity of the critical services by incorporating a self-healing mechanism to overcome any failures that may be caused by the intruder. The main design features of the IPS has been successfully abstracted from the human immune system. In the next section, the specific objectives and relevant achievements are discussed.

8.3 BIIPSS Model Specification and Design

A specification for the proposed IPS and SH, as an application system using multiagent system, was needed to clarify the agent's role, function and responsibilities as well as the states and transitions of each agent towards realizing the agent's aim. In this research, a specification language has been developed to specify

the roles, functions and responsibilities of the multiagent system. The specification language was needed to describe in details the implementation role, each of IPS and SH.

The IPS and SH specification language was constructed using set theory and Z-notation symbols. Both Z-notation and set theory were considered as logic design tools. The new specification language has allowed for a strong system description. To analyze the design of the IPS and SH model, Petri nets as an analysis and modeling tool were used to fulfill all the states and transitions of the multiagent system model. The Petri nets for IPS and SH agents have been successfully designed and verified mathematically for specific features of multiagent model such as free deadlock, liveness and boundedness. The use of Petri nets supports to develop a robust autonomous system. The first and main objective of the research, which was autonomous system, was accomplished by verifying the Petri nets model for the multiagent system.

8.4 BIIPSS Mathematical and Computational Model

The second objective of the research was to minimize the errors in intrusion detection, therefore more possible accurate prevention responses obtained. The derivations of the mathematical and computational models of the bio inspired IPS and SH has been undertaken using the features of k-NN means cluster and Gaussian mixture. The main algorithm for detection used a nonlinear classification methods based on the k-NN cluster, k-means and Gaussian mixture. The integration of bio inspired abstraction features of HIS and pattern recognition algorithms was efficient for detection of intrusion and identification of corresponding trust prevention response.

A highly accurate detection has been possible due to the nonlinearity features of the mathematical model of the classification derivation. The detection of intrusion and trustability of the prevention responses were tracked with the self-healing algorithm after the analysis and adaptation of the intrusion have been fully completed. The fully established mathematical and computational models of the IPS and SH multiagent

system, completed the development of the bio inspired intrusion prevention and self-healing system for critical services network.

8.5 BIIPSS Test Limitations and Validation

The limitations of this work were primarily due to the conditions of the simulation test:

- The test and simulation were implemented statically with specific real time standard datasets.

- Due to limited availability of resources, the tests were also used as the samples of random record. The ratio of abnormal behavior records to normal behavior records was specified by the researcher.

- Other effects such as network engineering design and network traffic noise have not been considered since this research has been focused mainly on the design of the IPS and SH algorithms as software development.

To validate the system developed in this research, the IPS and SH were simulated and tested using two different standard datasets. The simulation results showed that the autonomous agents were highly accurate and fast in detecting the 4 classes of network intrusion with 23 different categories of abnormal behavior. The first simulation test used KDD Cup dataset. Four different tests for 4 different classes of abnormal behaviors were run. The accuracy varies from 99.6-100%, which was considered as very high compared to the results obtained by IMAAD. The false positive percentage error was reduced to the range from 0.0 to 0.08. Meanwhile, the false negative error percentage was minimized to the range from 0.0 to 0.4.

The second test was specified for the sendmail dataset from UNM as process behavior intrusion detection and prevention test. The system developed in this research has shown significant increase in accuracy in process behavior classification for intrusion detection and prevention responses; 98.51 % accuracy, 0.0 for false positive error and 1.49% for false negative error. This percentage of false negative

error, which was considered relatively high, may be due to the small size of training and testing data.

The results were also validated according to the parameters used in the classification algorithm such as subclass number, best extraction feature for best classification accuracy, and values of the best clustering classes and representatives for each class. The prevention positive and negative predictabilities and the throughput of processing data were also validated, which showed noteworthy results. Finally, analysis and adaptation capabilities were measured, which showed high performance in identifying the class and categories of the abnormal behaviors.

The nonlinear classification algorithms have been verified as a new classification algorithm for diverse detection purposes such as cancer detection, image recognition and so on. This verification was obtained by simulation tests for cancer detection. The detail of this simulation test provided in Appendix F.

8.6 New Benchmarking Metrics

This research has introduced a new benchmarking metrics and has used new detection capabilities metric. The new metrics were measured using the results obtained in chapter 7. For an ideal system, the values of the detection capabilities must tend to 1. The IPS presented in this research has a detection capability ranging from 0.975-1.0. These results showed that the IPS system works ideally when detecting a class of abnormal behavior.

Moreover, a nonlinear classification algorithm for detection was used to define trustability of the Prevention Predictability, for which anew metrics for prevention was introduced. The new metrics measure the confidence of the system and how it was trusted to take the prevention response in the time of detection. In this research, two new benchmarking metrics were defined and compared, to show which was the best for calculating trust predictability for prevention. The values of trust predictability were used to determine the threshold that keeps track with the self-healing system and triggers the healing process. These new metrics can be used in future related research work.

Autonomic computing and particularly self-healing systems were new fields in software engineering. Specific metrics to measure the performance of self-healing system were still not available even though some evaluation metrics has been introduced by a few researchers. In this research, the principles of measuring self-healing system were simplified. The new metrics for reliable self-healing were used to measure and benchmark the self-healing system. The results obtained satisfy the requirements for keeping continuity of the system in static test. Tracking of the self-healing system with prevention predictability trust and threshold were accomplished with intelligent training. As an enhanced safety measure, the self-healing mechanism can also be triggered periodically according to the setting by the system designer, regardless of whether an intrusion is present or not.

8.7 Research Contributions

The main contributions from this research were mentioned as follows:

- Autonomous biological inspired intrusion prevention system.
- Highly accurate IPS system for detection of intrusion with minimum false errors, and high predictability of trusted prevention response.
- Nonlinear classification algorithms which can be used for diverse detection purposes rather than for intrusion detection solely.
- A self-healing system that keeps tracking the network with prevention predictability threshold.
- A new conceptual framework for developing AIS system.
- Specification language for IPS and SH systems for use in network security system.
- New classification metrics and benchmarking of intrusion detection capabilities and prevention predictability trust.
- New metrics for self-healing systems.

- The presented system was a hybrid which can be used as network-based, as well as host-based intrusion prevention system.

8.8 Future work

For future work, researchers need to have a deep knowledge about the natural mechanism as rich metaphor for more system inspiration. This work has shown that the integration of HIS inspired model, pattern recognition and machine learning was able to bring significant improvement to the security of critical services network system. By following the same methodology, this combination can be extended to develop other security systems to gain a more reliable and secure network systems.

The biological inspired system needs to be validated as a real time system to overcome any deficiencies that may arise when the system is implemented in real time. The accuracy of the process behavior showed lower value, this test needs a more reliable dataset and of sufficient size. The implementation and simulation test must be performed using other datasets to solve problems in other domains.

In some tests, the false negative values were significantly higher than expected. Thus, the detection algorithms must be improved to keep both false positive and false negative rates minimized to 0. The nonlinear classification algorithm must be modified to find the Global minimum of variance instead of getting the local minimum variance; this is one of the problems of k-means cluster algorithm. Even though the use of the adaptation algorithm has shown excellent results, but a more detailed specification of the features of recognition and registration of anomaly intrusion is needed to allow for more diagnoses ability and particular damaged component identification rather than diagnosing all the system components for fault and test.

For faster and accurate classification, selection of the features extraction i.e. reducing number of features can be implemented by using the following metric:

$$Var_{total} = \min_{i,j} \left\{ \sum_{i \neq j} \frac{\left(m_i - m_j\right)^2}{var_i} \right\}, \quad (8.1)$$

where; var_i is variance of the class *i*.

In spite of the successful new integration between the IPS and self-healing system, a reliability feature can be added to keep the self-healing system itself healthy, in the case the self-healing system components themselves are the target of the intelligent intruder. Finally, the performance of the self-healing system in real time needs to be measured and compared with other previous self-healing systems.

REFERENCES

[1] S.M. Garrett, “How do we evaluate artificial immune systems?” Evolutionary computation, vol. 13, Jan. 2005, pp. 145-77.

[2] F. Sun,” Artificial Immune Danger Theory Based Model for Network Security Evaluation”, journal of Networks, Vol 6, No 2, 2010, PP 255-262.

[3] J. Kim, P.J. Bentley, U. Aickelin, J. Greensmith, G. Tedesco, and J. Twycross, “Immune system approaches to intrusion detection – a review,” Natural Computing, vol. 6, Jan. 2007, pp. 413-466.

[4] J. Twycross and U. Aickelin, “Biological Inspiration for Artificial Immune Systems,”Proc of the 6th International Conference on Artificial Immune Systems (ICARIS 2007), vol. 4628, 2007, pp. 300-311.

[5] S. Forrest, S.A. Hofmeyr, and A. Somayaji, “Computer immunology,” Communications of the ACM, vol. 40, 1997, pp. 88-96.

[6] S. Forrest, A.S. Perelson, L. Allen, and R. Cherukuri, “Self-nonself discrimination in a computer,” Research in Security and Privacy, proc 1994 IEEE Computer Society Symp on, IEEE, 1994,pp. 202–212.

[7] B. Kim, “Evaluating negative selection in an artificial immune system for network intrusion detection,” Dept CIS, Univ College of London,U.K. GECCO39; vol. 2, 2001, pp. 225-228.

[8] S. Hofmeyr, S. Forrest, and A. Somayaji, “Intrusion detection using sequences of system calls,” Journal of Computer Security, vol. 2, 1998, pp. 225-228.

[9] F.A. González and D. Dasgupta, “Anomaly detection using real-valued negative selection,” Genetic Programming and Evolvable Machines, vol. 4, 2003, pp. 383-403.

[10] P. Matzinger, “The danger model: a renewed sense of self,” Science,New York, N.Y. USA, vol. 296, Apr. 2002, pp. 301-305.

[11] U. Aickelin and S. Cayzer, “The Danger Theory and Its Application to AIS,” Proc. of the 1st International Conf. on Artificial Immune Systems (ICARIS2002), 2002, pp. 141-148.

[12] A. Krizhanovsky and A. Marasanov, “An Approach for Adaptive Intrusion Prevention Based on The Danger Theory,” The 2nd Int Conf. on Availability, Reliability and Security (ARES’07), Apr. 2007, pp. 1135-1142.

[13] F. Sun, M. Kong, and J. Wang, “An immune danger theory inspired model for network security threat awareness,” Proc. Of the 2010 2nd Int Conf on Multimedia and Information Technology, vol. 2, 2010 pp.93-95.

[14] F. Sun, X. Han, J. Wang, “An immune danger theory inspired model for network security monitoring,” Proc. Of the 2010 Int Conf Challenges in Environmental Science and Computer Engineering (CESC2010), vol. 2, 2010 pp. 33-35.

[15] J. Greensmith, U. Aickelin, and S. Cayzer, “Introducing dendritic cells as a new immune-inspired algorithm for anomaly detection,” 4th International Conf on Artificial Immune Systems, August 2005, pp. 153–167.

[16] G. Tedesco, J. Twycross, and U. Aickelin, “Integrating Innate and Adaptive Immunity for Intrusion Detection,” Lecture Notes in Computer Science including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 4163 LNCS, 2010, p. 10.

[17] T. Stibor, R. Oates, G. Kendall, and J.M. Garibaldi, “Geometrical insights into the dendritic cell algorithm,” Proc. of the 11th Annual Conf. on Genetic and evolutionary computation (GECCO ’09), 2009, p. 1275.

[18] Y. Guo and C. Wang, “Autonomous decentralized network security system,” Proc. 2005 IEEE Networking, Sensing and Control, 2005, pp. 279-282.

[19] J.O. Kephart and D.M. Chess, “The Vision of Autonomic Computing,” Computer, vol. 36, 2003, pp. 41-50.

[20] R.A. Kemmerer and G. Vigna, “Intrusion detection: a brief history and overview,” Reliable Software Group, CS Dept, Univ of California Santa Barbara ,CA,Rep,Computer,IEEE, vol. 35, Aug. 2002, pp. supl. 27-supl. 30.

[21] E. Schultz and E. Ray, “The future of intrusion prevention,” Computer Fraud & Security, SciVerse, vol.2007, Aug. 2007, pp. 11-13.

[22] Federal Beureau Investigation, FBI, http://www.fbi.gov/,online-2010

[23] I.V. Paputungan and A. Abdullah, “Modelling a Survivable System through Critical Service Recovery Process,” snd UKSIM European Symp. on Computer Modeling and Simulation, Sep. 2008, pp. 441-446.

[24] E. Biermann, “A comparison of Intrusion Detection systems,” Computers & Security, vol. 20, Dec. 2001, pp. 676-683.[1] S.M. Garrett, “How do we evaluate artificial immune systems?,” Evolutionary Computation, vol. 13, 2005, p. 145–177.

[25] A. Fuchsberger, “Intrusion Detection Systems and Intrusion Prevention Systems,” Information Security Group, Royal Holloway, Univ of London. U.K, Information Security Tech. Rep,Elsevier,vol. 10, 2005, pp. 134-139.

[26] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly detection: A survey,” ACM Computing Surveys, vol. 41, 2009, pp. 15-73.

[27] K. Scarfone and P. Mell, “Guide to Intrusion Detection and Prevention Systems (IDPS)” the National Institute of Standards and Technology, Commerce Dept,Nist Special Publication Washington USA, online 2007, available at:http://csrc.nist.gov/publications/nistpubs/800-94/SP800-94.pdf.

[28] D. Yeung and Y. Ding, “Host-based intrusion detection using dynamic and static behavioral models,” Pattern Recognition, vol. 36, 2003, pp. 229-243.

[29] H. Debar, M. Dacier, and A. Wespi, “A Revised Taxonomy for Intrusion Detection Systems,” Ann des Telecommunications, vol. 55, 1999, pp. 361-378.

[30] D.E. Denning, “An Intrusion Detection Model,” Proc. of the IEEE Symp. on Research in Security and Privacy, IEEE Computer Society Press, 1986.

[31] A. Patcha and J. Park, “An overview of anomaly detection techniques: Existing solutions and latest technological trends,” Computer Networks, vol. 51, 2007, pp. 3448-3470.

[32] C.P. Pfleeger, Shari L.P “Security in Computing”, New Jersy, USA, Prentice Hall, 3rd Ed,2003.

[33] G.Stoneburner, “Underlying Technical Models for Information Technology Security”,The National Institute of Standards and Technology, Commerce Dept, Nist Special Publication, NISTWashington USA,” 2001.pp 1-28

[34] N. Athanasiades, R. Abler, J. Levine, H. Owen, and G. Riley, “Intrusion detection testing and benchmarking methodologies,” Proc. 1st IEEE International Workshop on Information Assurance, (IWIAS), 2003, pp. 63-72.

[35] J. McHugh, A. Christie, and J. Allen, “Defending yourself: the role of intrusion detection systems,” IEEE Software, vol. 17, 2000, pp. 42-51.

[36] S. Northcutt and J. Novak, “Network Intrusion Detection: .An Analyst Handbook,” 3rd edition, New Riders, USA, 2003.

[37] C. Endrof, E. Schultz, J. Mellander,”Intrusion detection and prevention,” McGraw Hill, Hacking Exposed, 2004.

[38] A. Seleznyov, “An anomaly intrusion detection system based on intelligent user recognition,” Univ Jyväskylä, Pekka Olsbo, Marja-Leena Tynkkynen Rep, 2002.

[39] R. Sekar, T.F. Bowen, and M.E. Segal, “On Preventing Intrusions by Process Behavior Monitoring,” Proc. of the Workshop on Intrusion Detection and Network Monitoring Santa Clara CA USA, USENIX, 1999, pp. 29-40.

[40] J. Kim and P. Bentley, “The Human Immune System and Network Intrusion Detection,” Proc. of the 7th European Conf. on Intelligent Techniques and Soft Computing (EUFIT99), 1999.

[41] W. Lee and S.J. Stolfo, “A framework for constructing features and models for intrusion detection systems,” ACM Transactions on Information and System Security, vol. 3, 2000, pp. 227-261.

[42] S.A. Hofmeyr and S. Forrest, “Immunity by design: An artificial immune system,” Proc. of the Genetic and Evolutionary Computation Conf., Citeseer, 1999, pp. 1289–1296.

[43] A. Somayaji, S. Hofmeyr, and S. Forrest, “Principles of a computer immune system,” Proc. of the 1997 workshop on New security paradigms (NSPW ’97), 1997, pp. 75-82.

[44] C. Janeway, P. Travers, and M. Walport, “Immunobiology: The Immune System in Health and Disease,” Garland Publishing, 2005.

[45] H.I. Works, “Understanding the Immune System - How It Works.”National Institute of Allergy and Infectious Diseases San Francisco USA Rep,No. 07-5423, 2007.

[46] M. Brossard and S.K. Wikel, “Tick immunobiology,” Parasitology, vol. 129, Oct. 2004, pp. S161-S176.

[47] A. Panda, A. Arjona, E. Sapey, F. Bai, E. Fikrig, R.R. Montgomery, J.M. Lord, and A.C. Shaw, “Human innate immunosenescence: causes and consequences for immunity in old age,” Trends in immunology, vol. 30, Jul. 2009, pp. 325-33.

[48] S. Marino, S. Pawar, C.L. Fuller, T.A. Reinhart, J.L. Flynn, D.E. Kirschner, Dendritic Cell Trafficking and Antigen Presentation in the Human Immune

Response to Mycobacterium Tuberculosis , Journal of Immunology, 2004, pp. 494-506

[49] S. Cayzer, “On the Effects of Idiotypic Interactions for Recommendation Communities in Artificial Immune Systems,” Proc of the 1st Int Conf on Artificial Immune Systems (ICARIS-2002), UK, 2002, pp. 154-160.

[50] J.O. Kephart, G.B. Sorkin, W.C. Arnold, D.M. Chess, G.J. Tesauro, and H. Integrity, “Biologically Inspired Defenses Against Computer Viruses,” IJCAI'95 Proc of the 14th int joint conference on Artificial intelligence, vol.1, 1995, pp. 985-996.

[51] T. Pradeu and E.D. Carosella, “The Self Model and the Conception of Biological Identity in Immunology,” Biology & Philosophy, vol. 21, Mar. 2006, pp. 235-252.

[52] S. A Hofmeyr and S. Forrest, “Architecture for an artificial immune system,” Evolutionary computation, vol. 8, Jan. 2000, pp. 443-73.

[53] S. Forrest, S. Hofmeyr, A Somayaji, and T. a Longstaff, “A sense of self for Unix processes,”Proc. of IEEE Symp. on Security and Privacy, 1996, pp. 120-128.

[54] D. Dasgupta and F. González, “An Immunity-Based Technique to Characterize Intrusions in Computer Networks,” IEEE Transactions on Evolutionary Computation, vol. 6, 2002, pp. 1081-1088.

[55] P. D' haeseleer, S. Forrest, and P. Helman, “A Distributed Approach to Anomaly Detection,” ACM Transactions on Information System Security, 1997, 30 pages

[56] U. Aickelin and J. Greensmith, “Sensing danger: Innate immunology for intrusion detection,” Information Security Technical Report, vol. 12, 2007, pp. 218-227.

[57] P. Matzinger, “Essay 1: The Danger model in its historical context,” Scandinavian Journal of Immunology, vol. 54, 2001, pp. 4-9.

[58] P. Matzinger, “An innate sense of danger,” Ann of The New York Academy of Sciences, vol. 961, 2002, pp. 341-342.

[59] H. Psaier and S. Dustdar,” A survey on self-healing systems: approaches and systems” , Journal of computing Springer-Verlag, 2010, PP 43-73.

[60] H. Psaier, F. Skopik, D. Schall, S.Dustdar,” Behavior monitoring in self-healing service-oriented systems”, Proc of International Computer Software and Applications Conf, 2010, Pages 357-366

[61] A.D. Keromytis, “Characterizing self-healing software systems,” Proc. of the 4th International Conf. on Mathematical Models and Architectures for Computer Networks Security (MMMACNS), St. Petersburg Russia, Citeseer, 2007, pp. 22–33.

[62] M. Jiang, J. Zhang, D. Raymer, and J. Strassner, “A Modeling Framework for Self-Healing Software Systems,” Motorola Network Infrastructure Research Lab, Autonomics Research, Illinois USA, Rep, 2007.

[63] S. George, D. Evans, and L. Davidson, “A biologically inspired programming model for self-healing systems,” Communications, 2002, pp. 102-104.

[64] J. A.Chaudhry and M.Imran, "Autonomic Fault Identification for Ubiquitous Self Healing Systems," Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing, 2010, pp.370-374.

[65] G. Yoo, J. Park, and E. Lee, “Hybrid Inference Architecture and Model for Self-healing System,” Lecture Notes in Computer Science, Springerlink, vol 4238 2006, pp. 566 - 569.

[66] J. Park, “Self-healing Mechanism for Reliable Computing,” Int. Journal of Multimedia and Ubiquitous Engineering, vol. 3, 2008, pp. 75-86.

[67] E. Grishikashvilipereira, R. Pereira, and A. Talebbendiab, “Performance evaluation for self-healing distributed services and fault detection mechanisms,” Journal of Computer and System Sciences, vol. 72, 2006, pp. 1172-1182.

[68] J. Wong, “Intelligent mobile agents in large distributed autonomous cooperative systems,” Journal of Systems and Software, vol. 47, Jul. 1999, pp. 75-87.

[69] M. Swimmer, “Using the danger model of immune systems for distributed defense in modern data networks,” Computer Networks, vol. 51, Apr. 2007, pp. 1315-1333.

[70] B. Barrishi,” Modeling the Artificial Immune System to the Human Immune System with the Use of Agents,” MSc. Thesis, Univ Oklahoma State, Graduate College, Oklahoma USA , 2004.

[71] A. Boukerche, R.B. Machado, K. Juca, J. Sobral, and M.S.M.A. Notare, “An agent based and biological inspired real-time intrusion detection and security model for computer network operations,” Computer Communications, vol. 30, 2007, pp. 2649-2660.

[72] T. Sproull and J. Lockwood, “Distributed Instrusion Prevention in Active and Extensible Networks,” Network Applied Research Laboratory Department of Computer Science and Engineering, Washington University in Saint Louis USA, 2005.

[73] R.L. Fanelli, “A hybrid model for immune inspired network intrusion detection,” Lecture Notes in Computer Science including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 5132 LNCS, 2008, pp. 107-118.

[74] J. Yang, X. Liu, T. Li, G. Liang, and S. Liu, “Distributed agents model for intrusion detection based on AIS,”Knowledge-Based Systems, vol. 22, Mar. 2009, pp. 115-119.

[75] R.C. Cardoso and M. Freire, “SAPA: Software Agents for Prevention and Auditing of Security Faults in Networked Systems,” ALecture Notes in Computer Science, Springerlink, Volume 3391, 2005, pp. 80-88.

[76] K.M. Begnum and M. Burgess, “A scaled, immunological approach to anomaly countermeasures: combining pH with cfengine,” IFIP/IEEE Eighth Int Symp. on Integrated Network Management, 2003, pp. 31-42.

[77] S. Powers and J. He, “A hybrid artificial immune system and Self Organising Map for network intrusion detection,” Information Sciences, vol. 178, Aug. 2008, pp. 3024-3042.

[78] S. Sarafijanovic, “An Artificial Immune System for Misbehavior Detection in Mobile Ad-Hoc Networks with Virtual Thymus, Clustering, Danger Signal, and Memory Detectors,” International Conf. on Artificial Immune Systems (ICARIS), vol. 2004, 2004, p. 342.

[79] S. Sarafijanović and J.-Y. Le Boudec, “An artificial immune system approach with secondary response for misbehavior detection in mobile ad hoc networks,”IEEE Transactions on Neural Networks, vol. 16, 2005, pp. 1076-1087.

[80] J. Greensmith, J. Feyereisl, and U. Aickelin, “The DCA: SOMe comparison,” Evolutionary Intelligence, vol. 1, May. 2008, pp. 85-112.

[81] J. Greensmith and U. Aickelin, “Dendritic Cells for SYN Scan Detection,” Proc. of GECCO 07 Genetic and Evolutionary Computation Conf., ACM Portal, 2007, pp. 49-56.

[82] S. Liu, T. Li, D. Wang, X. Hu, and C. Xu, “Multi-agent network intrusion active defense model based on immune theory,” Wuhan University Journal of Natural Sciences, vol. 12, Jan. 2007, pp. 167-171.

[83] S. Stepney, R. Smith, J. Timmis, and A. Tyrrell, “Towards a Conceptual Framework for Artificial Immune Systems,” Third International Conf. on Artificial Immune Systems, 2004, pp. 53-64.

[84] M. Neal, S. Stepney, R.E. Smith, and J. Timmis, “Conceptual frameworks for artificial immune systems,” International Journal of Unconventional Computing, vol. 1, 2005, pp. 315-338.

[85] L.N.D. Castro and J.I. Timmis,"Artificial immune systems as a novel soft computing paradigm,"Soft Computing - A Fusion of Foundations, Methodologies and Applications, vol. 7, Aug. 2003, pp. 526-544.

[86] A. Igbal," Danger Theory metaphor in Artificial Immune System for System call Data", Ph.D dissertation, Dept. of CIS, Universiti Teknologi Malaysia, 2006.

[87] J. Twycross and U. Aickelin. "Libtissue Implementing Innate Immunity". In Congr on Evolutionary Computation, CEC, 2006, pp. 499–506.

[88] A. Panda, A. Arjona, E. Sapey, F. Bai, E. Fikrig, R.R. Montgomery, J.M. Lord, and A.C. Shaw, "Human innate immunose nescence: causes and consequences for immunity in old age," Trends in immunology, vol. 30, Jul. 2009, pp. 325-33.

[89] P. Horn, "Autonomic Computing: IBM's perspective on the State of Information Technology" .http://www.research.ibm.com/autonomic/, Oct 2001.

[90] M. Parashar and S. Hariri, "Autonomic Computing: An Overview," Unconventional Programming Paradigms, Lecture Notes in Computer Science, Springer Verlag, vol. 3566, 2005, pp. 257-269.

[91] R. Nakkeeran, T. Aruldoss Albert and R.Ezumalai,"Agent Based Efficient Anomaly Intrusion Detection System in Ad hoc networks", IACSIT International Journal of Engineering and Technology ,Vol. 2, No.1, February, 2010, pages 1-5.

[92] M. Schumacher, "Objective coordination in multi-agent system engineering: Design and Implementation," Springer-Verlag Inc.vol. 2039, 2001, pp. 137-149

[93] N. Hiroyuki, M. Fumio," Design and Implementation of Security System Based on Immune System", Springer-Verlag Berlin Heidelberg, ISSS , LNCS 2609, 2003, pp. 234–248.

[94] T. Murata, “Petri nets: Properties, analysis and applications,” Proc. of the IEEE, vol. 77, 1989, pp. 541-580.

[95] J.R. Celaya, A.A. Desrochers, and R.J. Graves, “Modeling and analysis of multi-agent systems using petri nets,” 2007 IEEE Int Conf. on Systems Man and Cybernetics, vol. 4, 2007, pp. 1439-1444.

[96] Q. Zhou and X. Guo, “The learning mechanism design with Petri nets for adaptive agent,” 2008 International Conf. on Service Systems and Service Management, Jun. 2008, pp. 1-5.

[97] FIPA, the standards organization for agents and multi-agent systems, http://www.fipa.org , IEEE , eleventh standards committee on 8 June 2005.

[98] O. Robert, Graham. K, Jonathan. M.G. “Limitation of frequency Analysis for Dendritic Cell Population Modelling,”ICARIS 2008, Springer Verlag Berlin Heidelberg, LNCS 5132,2008, pp. 328-339.

[99] A. Banerjee, V.Chandola, A.Lazarevic, V.Kumar, and J.Srivastava, "Data Mining for Anomaly Detection", Presented at ECML PKDD Conf, Antwerp, Belgium Expo, September 2008.

[100] T.P. Ninag, Steinbach M., and Kumar, V. “Introduction to Data Mining,” Addison-Wesley, 2005.

[101] Duda, R. O., Hart, P. E., and Stork, D. G. “Pattern Classification,” 2nd Edition, Wiley, New York.2000.

[102] H. Dutta, C. Giannella, K. Borne, and H. Kargupta, “Distributed top-k outlier detection in astronomy catalogs using the DEMAC system,” Proc. of 7th SIAM International Conf. on Data Mining, 2007.

[103] C. De Stefano, C. Sansone, and M. Vento, “To reject or not to reject: that is the question-an answer in case of neural classifiers,” IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews), vol. 30, 2000, pp. 84-94.

[104] D. Barbara, N. Wu, and S. Jajodia, "Detecting Novel Network Intrusions using Bayes Estimators," First SIAM Conf on Data Mining, Citeseer, 2001, pp. 1-17.

[105] D. Barbará, J. Couto, S. Jajodia, and N. Wu, "ADAM: A test bed for exploring the use of data mining in intrusion detection," ACM SIGMOD Record, vol. 30, 2001, pp. 15–24.

[106] L .M. Ibrahim," Anomaly Network Intrusion Detection System based on distribution Time-delay Neural Metwork(DTDNN)", Journal of Engineering Science and Technology, Vol. 5, No. 4 ,2010. pp 457 - 471

[107] V. Roth, "Outlier Detection with One-class Kernel Fisher Discriminants," Advances in Neural Information Processing Systems 17, MIT Press, 2005, pp. 1169-1176.

[108] V. Roth, "Kernel fisher discriminants for outlier detection," Neural Computation, vol. 18, 2006, pp. 942-960.

[109] M.F. Augusteijn and B.A. Folkert, "Neural network classification and novelty detection," International Journal of Remote Sensing, vol. 23, 2002, pp. 2891-2902.

[110] S. Jakubek and T. Strasser, "Fault-diagnosis using neural networks with ellipsoidal basis functions," Proc. of the 2002 American Control Conf. (IEEE Cat.No.CH37301), 2002, pp. 3846-3851.

[111] A. Valdes and K. Skinner, "Adaptive, Model-based Monitoring for Cyber Attack Detection,"Springer-Verlag Berlin Heidelberg, 2000, pp. 80-92.

[112] A.A. Sebyala, T. Olukemi, and L. Sacks, "Active Platform Security through Intrusion Detection Using Naïve Bayesian Network for Anomaly Detection," Univ college London. England, UK, 2002,pp. 1-5.

[113] A. Bronstein, J. Das, M. Duro, R. Friedrich, G. Kleyner, M. Mueller, S. Singhal, and I. Cohen, "Using Bayesian Networks for Detecting Anomalies in Self-Aware Services,"HP Laboratories, Palo Alto Rep, October 2001.

[114] E. Eskin, “Anomaly Detection over Noisy Data using Learned Probability Distributions,” Proc. of the 25th Int. Conf. on Machine learning, Morgan Kaufmann, San Francisco, CA, 2000, pp. 255-262.

[115] A. Lazarevic, L. Ertoz, A. Ozgur, V. Kumar, and J. Srivastava, “A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection 2 - Evaluation of Intrusion Detection Systems,”Proc of SIAM the third intl conf in data mining, vol 3,2003, pp. 25-36.

[116] K.A. Heller, K.M. Svore, A.D. Keromytis, N. York, and S.J. Stolfo, “One Class Support Vector Machines for Detecting Anomalous Windows Registry Accesses,”Dept. C.S. Univ.New York.USA, 2003.

[117] D. Barbara, Li, Y., Couto, J., Lin, J.-L., and Jajodia, S,” Bootstrapping a data mining intrusion detection system, ”In Proc. of the 2003 ACM Symp. on Applied Computing. ACM Press, 2003, pp.421-425.

[118] K. Chan, M.V. Mahoney, and M.H. Arshad, “A Machine Learning Approach to Anomaly Detection,” Tech. Rep. CS-003 06, Department of Computer Science, Florida Institute of Technology, Melbourne FL 32901, 2003, pp. 1-13.

[119] M. Otey, S. Parthasarathy, Ghoting, G. Li, S. Narravula, and D. Panda, “Towards NIC-based intrusion detection,” Proc. of the ninth ACM SIGKDD Int. Conf. on Knowledge discovery and data mining (KDD ’03), 2003, p. 723.

[120] G. Tandon and P.K. Chan, “Weighting versus pruning in rule validation for detecting network and host anomalies,” Proc. of the 13th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD ’07), 2007, p. 697.

[121] W. Lee, S.J. Stolfo, and K.U.I.W. Mok, “Adaptive Intrusion Detection: A Data Mining Approach,” Artificial Intelligence Review, 2001, pp. 533-567.

[122] M. Qin and K. Hwang, “Frequent Episode Rules for Intrusive Anomaly Detection with Internet Data mining,”Proc. of the Third IEEE International Symp. on Network Computing and Applications, 2004, pp. 161-168.

[123] S. Brahim-belhouari, A. Bermak, S. Member, M. Shi, P.C.H. Chan, and S. Member, “Fast and Robust Gas Identification System Using an Integrated Gas Sensor Technology and Gaussian Mixture Models,” Sensors. Peterborough, NH, vol. 5, 2005, pp. 1433-1444.

[124] X. Jia, S. Member, and J.A. Richards, “Fast k-NN Classification Using the Cluster-Space Approach,” IEEE GEOSCIENCE AND REMOTE SENSING LETTERS, vol. 2, no. 2, APRIL 2005, vol. 2, 2005, pp. 225-228.

[125] B.VDasarstly, Ed.,”Nearest Neighbor (NN) Norms: NN Pattern classification ”, AC:IEEE Computer Society press, 1990.

[126] D.M. Titterington, A.F.M. Smith, and U.E. Mako,”Statistical analysis of finite mixture distriburions,” John Wiley, NewYork, 1985.

[127] D. Barbara, J. Couto, S. Jajodia, L. Popyack, and N. Wu, “ADAM: Detecting intrusions by data mining,” IEEE Workshop on Information Assurance and Security, 2001, pp.11-16.

[128] W. Lee, and S.J. Stolfo, “A Framework for constructing features and models for intrusion detection systems,” ACM Transactions on Information and System Security, 2000, vol.3, no. 4, pp. 227-261.

[129] G. Gu, P. Fogla, D. Dagon, W. Lee, and B. Skorić, “Measuring intrusion detection capability: an information-theoretic approach,” Proc. of the ACM Symp. on Information Computer and Communications Security (ASIACCS 06), ACM, 2006,pp. 90-101.

[130] KDD CUP 99 DataSetAvailable online (2010) at http://www.sigkdd.org/kddcup/index.php,1994.

[131] University of New Mexico syscall datasets. Available online (2010) at: http //www.cs.unm.edu/~immsec/data/, 1998.

[132] R.P. Lippmann, D.J. Fried, I. Graf, J.W. Haines, K.R. Kendall, D. McClung, D. Weber, S.E. Webster, D. Wyschogrod, R.K. Cunningham, and M.A. Zissman, “Evaluating intrusion detection systems: The 1998 DARPA off-line

intrusion detection evaluation," Proc. DARPA Information Survivability Conf. and Exposition (DISCEX00), vol. 2, 2000, pp. 12-26.

[133] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," 2009 IEEE Symp. on Computational Intelligence for Security and Defense Applications, Jul. 2009, pp. 1-6.

[134] H.G. Kayacik, A.N. Zincir-Heywood, and M.I. Heywood, "Selecting Features for Intrusion Detection: A Feature Relevance Analysis on KDD 99 Intrusion Detection Datasets," NSERC and CFI. NIMS Laboratory,http://www.cs.dal.ca/projectx/., 2005, pp. 3-8.

[135] T. Cover and J. Thomas,"Elements of Information Theory," John Wiley, 1991.

[136] N.B. Anuar, S.M.Furnel, M.Papadaki, N.L. Clarke,"Response Mechansims For Intrusion Response Systems," Center of Security, Communication and Networks Research(CSCAN), Univ Plymouth, Plymouth ,U.K, 2010.

[137] A.B. Brown and C. Redlin, "Measuring the Effectiveness of Self-Healing Autonomic Systems," Snd Int. Conf. on Autonomic Computing (ICAC05), 2005, pp. 328-329.

[138] R. Griffith, R. Virmani, and G. Kaiser, "RAS-Models : A Building Block for Self-Healing Benchmarks," Programming Systems Lab (PSL), Dept. C.S. Columbia Univ.New York.USA,2007, pp. 1-23.

[139] Hui Li, Michael Muskulus, and LexWolters, "Modeling Job Arrivals in a Data-Intensive Grid," in Job Scheduling Strategies for Parallel Processing, EitanFrachtenberg and UweSchwiegelshohn, (ed.), Springer Verlag, Lect. Notes Comput. Sci. vol. 4376, 2006, pp. 210-231.

[140] J. Greensmith, "The Dendritic Cell Algorithm," PhD dissertation, Dept of CS, Nottingham Univ Nottingham U.K, 2007.

[141] J.P. Twycross," Integrated Innate and Adaptive Artificial Immune System Applied to Process Anomaly Detection," PhD dissertation, Dept. of CS, Nottingham Univ,Nottingham,U.K , 2006.

[142] Z. He , X. Deng,” Discovring cluster-based local outliers” Pattern Recognition letters,2003,1651-1660,.

[143] I.T.Nabney,”Algorithm for Pattern Reginition”london,UK:SpringerVerlag, 2003.

LIST OF PUBLICATIONS

1. M. Elsadig, Abdullah.A, "Biological inspired approach in parallel immunology system for network security," *Information Technology. ITSim 2008. Int Symp Informayion Technology, IEEE*, vol.1, Agust 2008, pp.1-6

2. M. Elsadig, Abdullah.A," Hybrid Biological Intrusion Prevention and Self-healing System for Network Security, Proc of National Postgraduate Conference NPC UTP, Malaysia, 2009.

3. M. Elsadig, Abdullah.A,"Biological Inspired Intrusion Prevention and SelfhealingSystem for Network Security Based on Danger Theory,"Computer Science Letter ,ISSR Journals,2009, vol 1,pp.1-17

4. M. Elsadig, Abdullah.A," Biological Intrusion Prevention and Selfhealing model for Network Security,"Proc of 2nd Int Conf on Future Networks, Sanya,China,IEEE,Jan 2010,pp.337-342.

5. M.Elsadig, Abdullah. A.; Samir, B.B. , "Intrusion Prevention and self-healing algorithms inspired by danger theory," The 2nd Int Conf on Computer and Automation Engineering (ICCAE),IEEE, 2010 , vol.5, 26-28 Feb. 2010 pp. 843-846.

6. M.Elsadig, Abdullah. A.; Samir, B.B.,"Immune Multi Agent System for Intrusion Prevention and Self Healing System Implement a Non-Linear Classification," Information Technology.Proc ITSim 2010. Int Symp Informayion Technology, IEEE, 15-17 June 2010

7. M.Elsadig, A.; Samir, B.B.,‘’ A novel Non Linear-Classification Algorithm for Anomaly Detection,” Proc of World Academy of Science , Engineering and Technology Conf, France, 28-30 June,2010.

8. M.Elsadig, A.; Samir, B.B.,Abdullah. A.” Immune Multiagent System for Network Intrusion Detection Use Nonlinear Classification Algorithm”, International Journal of computer and Applications foundation of Computer Science, New York, USA.Nov.2010.

APPENDICES

Appendix A

A.1 Design Features

- *Autonomy:* The IPS must not require outside management or maintenance i.e. it autonomously classifies and stops abnormal events.

- *Self-Repair*: It repairs itself by replacing damaged cells. It will be progressively more important for computers to handle most security problems automatically.

- *Distributed*: A distributed IPS can support robustness, configurability, extendibility and scalability. It is robust since the failure of one local intrusion detection and prevention process does not cripple the overall IPS. It is also easy to organize distributed IPSs when each intrusion detection and prevention processes can be simply tailored for the local requirements of a specific host. The addition of new intrusion detection process running on different operating systems does not require modification of existing processes, and hence it is extensible. It can also scale better, since the high volume of audit data is distributed amongst many local hosts and is analyzed by those hosts.

- *Self-Organized*: A self-organizing IPS provides adaptability and global analysis. Without external management or maintenance, a self-organizing IPS automatically detects intrusion signatures, which are previously unknown and/or distributed, and eliminates and/or repairs compromised components. Such a system is highly adaptive because there is no need for manual updates of its intrusion signatures as network environments change. Global analysis emerges from the interactions among a large number of varied intrusion detection processes.

- *Lightweight*: A lightweight IPS supports efficiency and dynamic features. Furthermore lightweight IPS does not impose a large overhead on a system or

place a heavy burden on CPU and I/O. It places minimal work on each component of the IPS. The primary functions of hosts and networks are not adversely affected by the monitoring. It also dynamically covers intrusion and non-intrusion pattern spaces at any given time rather than maintaining entire intrusion and non-intrusion patterns.

- *Multilayered*: A multilayered IPS increases robustness. The failure of one layer defense does not necessarily allow an entire system to be compromised. While a distributed IPS allocates intrusion detection processes across several hosts, a multi-layered IPS places different levels of sensors at one monitoring place.

- *Adaptability:* The IPS needs to learn to detect new abnormal activities, and retains the ability to recognize previously seen abnormal activities through learned knowledge based. A computer immune system should be similarly adaptable as the immune system, both learning to recognize new intrusions and remembering the signatures of previous attacks.

- *Diverse*: A diverse IPS provides robustness. A variety of different intrusion prevention processes spread across hosts will slow an attack that has successfully compromised one or more hosts. This is because an understanding of the intrusion process at one site provides limited or no information on intrusion processes at other sites.

- *Disposable*: A disposable IPS increases robustness, extendibility, and configurability. A disposable IPS does not depend on any single component. Any component can be easily and automatically replaced with other components.

A.2 Comparison Study

Table A.1: Comparisons of Intrusion Prevention Techniques

	Monitor	Components	Security Capabilities
Network-based	Network traffic for network segment	Sensors, management server	• Information gathering: identify hosts,.Oss, application and network characteristics. • Logging capabilities: time stamp, session ID, alert type, rating source and destination, protocols, payload data. • Detection capabilities: layer attacks, tuning and customization of port scan and alert setting. • Prevention capabilities: passive only, inline only, both.
Host-based	Characteristics of host and event	Detection software agents deployed to critical hosts	• Logging capabilities: data confirm the validity of alert, time stamp, alert type, rating payload data, event type IP, port, file names, user ID • Detection capabilities: code analysis, buffer over flow, application and library list, traffic filter. • Prevention capabilities: prevent code from being executed, stop incoming network traffic, file system monitoring. • Other capabilities: removable media restriction, process status monitoring, audiovisual device monitoring.
Network Behavior Analysis	Examine network traffic identify threat	Sensors, consoles, management server	• Information gathering capabilities: IP, OS. Provide services in IP,TCP and UDP port uses. Host communication services. • Logging capabilities: time stamp, alert type, rating source and destination, protocols, payload data, UDP port or ICMP types and code. • Detection capabilities: Denial of service attack, scanning worm, unexpected application services, policy violation. • Prevention capabilities: passive only, inline only, both.

Table A.2: Comparisons of Intrusion Prevention Methodologies

	Effectiveness	Method	Limitation
Signature-Based prevention	• Detecting and preventing known threats	• Compare the current incoming activities to the list of signature using string comparison or signature database.	• Little understanding of many network or application protocols. • Cannot track and understand the state of complex communications. • Cannot detect attacks that comprise multiple events if none of the events contains a clear indication of an attack.
Anomaly-Based prevention	• Very effective at detecting previously unknown threats	• Has profile that represents normal behavior of user, hosts, network connection or application. • Monitors the characteristics of typical activity over a period of time. • Then uses statitical method to compare the characteristics of current activity to thresholds related to the profile. •	• Inadvertently including malicious activity as part of a profile is a common problem. • Building profiles can be very challenging in some cases to make them accurate. • Produces many false positives because of benign activity that deviates significantly from profiles.
Stateful Protocol Analysis	• Capable in understanding and tracking the state of the network, transport and application protocols	• Relies on vendor developed universal profiles that specify how particular protocols should and should not be used. • Identify analysis can identify unexpected sequences of command. • Analyze protocols that perform authentication. • Usually include reasonableness checks for individual commands. • Uses protocol based on standard from software vendors and standard bodies.	• Very resource-intensive because of the complexity of analysis and overhead involved in performing state tracking for many simultaneous sessions. • Cannot detect attacks that do not violate the characteristics of generally acceptable protocol behavior . • Might conflict with the way the protocol is implemented.

Appendix B

Features Extraction Calculations

$$P(error) = \sum_{i=1}^{2} P(x \in class_i).P(error / x \in class_i)$$

and is assumed $P(x \in class_i) = \frac{1}{2}$ then ,

$$P(error/x \in class_1) = P(x > m_T)/x \in class_1)$$
$$= P\left[\frac{x - m_1}{\sigma_1} > \frac{m_T - m_1}{\sigma_1}\right] = P\left[N(0.1) > \frac{m_T - m_1}{\sigma_1}\right]$$
$$= P\left[N(0.1) < \frac{m_1 - m_T}{\sigma_1}\right] = \emptyset\left[\frac{m_1 - m_T}{\sigma_1}\right]$$
$$= 1 - \emptyset\left[\frac{m_T - m_1}{\sigma_1}\right]$$

$$P(error/x \in class_2) = P(x < m_T/x \in class_2)$$
$$= P\left[N(0.1) < \frac{m_T - m_2}{\sigma_1}\right]$$

$$= 1 - \emptyset\left[\frac{m_2 - m_T}{\sigma_2}\right]$$

$$P(error/_x) = \frac{1}{2}\left(1 - \emptyset\left(\frac{m_T - m_1}{\sigma_1}\right)\right) + \frac{1}{2}\left(1 - \emptyset\left(\frac{m_2 - m_T}{\sigma_2}\right)\right)$$

$$P(error/_x) = 1 - \frac{\left[\emptyset\left[\frac{m_T - m_1}{\sigma_1}\right] + \emptyset\left[\frac{m_2 - m_T}{\sigma_2}\right]\right]}{2}$$

If we take :

$\left[\frac{m_T - m_2}{\sigma_2}\right] > 1$ *and* $\left[\frac{m_1 - m_T}{\sigma_1}\right] > 1$ then;

$$P(error) < 1 - \emptyset(1) = 1 - 0.84$$

$\emptyset(1)$*=0.84* *// from cumulative standardized normal distribution*

$$P(error) \leq 15\%$$

Appendix C

Metrics Calculation

$$C_{ID} = \frac{I(x,y)}{H(x)} = \frac{H(x) - H(x/y)}{H(x)}$$

$$0 \leq C_{ID} \leq 1$$

$$H(X) = \sum_{i=0}^{1} P_i + logP_i = P_i(x=1)\log\left[\frac{1}{P(x=1)}\right] + P_i(x=0)\log\left[\frac{1}{P(x=0)}\right]$$

$$P(x=1) = \frac{n}{n+ab}$$

$$P(x=0) = \frac{ab}{n+ab}$$

$$H(H/Y) = \sum_{y}\sum_{x} P(x,y)logP(x/y)$$

$$= -[P(0,0)logP(0/0) + P(0,1)logP(0/1) + P(1,0)\log(1/0) + P(1,1)logP(1/1)]$$

$$P(x=0/y=0) = \frac{P(x=0,y=0)}{P(y=0)} = \frac{P(x=0)P(y=0/x=0)}{P(y=0)}$$

$$= \frac{P(x=0)(1-\delta)}{P(x=0)(1-\delta) + P(x=1)\beta}$$

$$P(x=1/y=0) = 1 - P(x=0/y=0)$$

$$P(x=0/y=1) = \frac{P(x=0,y=1)}{P(y=1)} = \frac{P(x=0)P(y=1/x=0)}{P(y=1)}$$

$$= \frac{P(x=0)\delta}{P(x=1)(1-\beta) + P(x=0)\delta}$$

$$P(x=1/y=1) = 1 - P(x=0/y=1)$$

P(0,0)=P(x=0)(1-δ)

P(1,1)=P(x=1)(1-β)

P(0,1)=P(x=0)δ

P(1,0)=P(x=1)β

Positive predictive value (PPV): The probability of a chance that an intrusion I, is present when an IPS outputs an alarm and response, A.

$$PP = P(I|A)$$

$$=\frac{P(I)TP}{P(A)TP+P(\bar{I})FP}$$

Negative predictive value (NPV): The probability of a chance that there is no intrusion $\rightarrow I$, when an IPS does not output an alarm and response, A.

$$NP = P(\rightarrow I| \rightarrow A)$$

$$= \frac{P(\bar{I})(1-FP)}{FN + P(\bar{I})(1-FP)}$$

Appendix D

D.1 Description of KDD 99 intrusion detection features

Table D.1.1 the description of the features and data types from [132]

Feature	Description	Data type
1.duration.	Duration of the connection	Continuous.
2.protocol type	Connection protocol (e.g. tcp, udp)	Discrete.
3. service	Destination service (e.g. telnet, ftp)	Discrete.
4. flag	Status flag of the connection	Discrete.
5. source bytes	Bytes sent from source to destination	Continuous.
6. destination bytes	ytes sent from destination to source	Continuous.
7. land	1 if connection is from/to the same host/port; 0 otherwise	Discrete.
8.wrong fragment	number of wrong fragments	Continuous.
9. urgent	number of urgent packets	Continuous.
10. hot	number of "hot" indicators	Continuous.
11. failed logins	number of failed logins	Continuous.
12. logged in	1 if successfully logged in; 0 otherwise	Discrete.
13 #compromised	number of "compromised" conditions	Continuous.
14. root shell	1 if root shell is obtained; 0 otherwise	Continuous.
15.su attempted	1 if "su root" command attempted; 0 otherwise	Continuous.
16. # root	number of "root" accesses	Continuous.
17. # file creations	number of file creation operations	Continuous.
18. # shells	number of shell prompts	Continuous.
19. # access files	number of operations on access control files	Continuous.
20. # outbound cmds	number of outbound commands in an ftp session	Continuous.
21. is hot login	1 if the login belongs to the "hot" list; 0 otherwise	Discrete.
22. is guest login	1 if the login is a "guest" login; 0 otherwise	Discrete.
23. Count	number of connections to the same host as the current connection in the past two seconds	Continuous.
24. srv count	number of connections to the same service as the current connection in the past two sec.	Continuous.

Feature	Description	Data type
25. serror rate	% of connections that have “SYN" errors	Continuous.
26. srv serror rate	% of connections that have “SYN" errors	Continuous.
27. rerror rate	% of connections that have “REJ" errors	Continuous.
28. srv rerror rate	% of connections that have “REJ" errors	Continuous.
29. same srv rate	% of connections to the same service	Continuous.
30. diff srv rate	% of connections to different services	Continuous.
31. srv diff host rate	% of connections to different hosts	Continuous.
32. dst host count	count of connections having the same destination host	Continuous.
33. dst host srv count	count of connections having the same destination host and using the same service	Continuous.
34. dst host same srv rate	% of connections having the same destination host and using the same service	Continuous.
35. dst host diff srv rate	% of different services on the current host	Continuous.
36. dst host same src port rate	% of connections to the current host having the same src port	Continuous.
37. dst host srv diff host rate	% of connections to the same service coming from different hosts	Continuous.
38. dst host serror rate	% of connections to the current host that have an S0 error	Continuous.
39. dst host srv serror rate	% of connections to the current host and specified service that have an S0 error	Continuous.
40. dst host rerror rate	% of connections to the current host that have an RST error	Continuous.
41. dst host srv rerror rate	% of connections to the current host and specified service that have an RST error	Continuous.

D.2 UNM dataset sample of Sendmail data set for two different sequences [131]

Table D.2.1: Sequence 1 Normal system calls frequencies

System call Number sequence 1	The first system call	Process ID frequencies for the			
		3794	1387	1387	1387
2	1	1	1	1	1
66	2	26	20	20	21
66	3	8	2	2	3
4	4	29	29	29	29
138	5	98	98	98	98
66					

Table D2.2: Sequence 1 intrusion system calls frequencies

System call Number sequence 1	The first system call	System calls frequencies			
		Process ID			
		204	243	936	954
2	1	1	1	1	1
66	2	11	11	11	11
66	3	0	0	0	0
4	4	13	13	13	13
138	5	272	272	272	272
66					

UNM dataset sample of Sendmail data set for two different sequences

Table D2.3: Sequence 2 system call frequencies

System call Number sequence 1	The first system call	System call frequencies			
		Process ID			
		1492	1575	1796	1863
105	1	1	1	1	1
104	2	38	92	25	25
104	3	23	39	16	16
106	4	22	22	21	21
105	5	44	44	44	44
108					
112					

Table D2.4: Sequence intrusion system calls frequencies dataset

System call Number sequence 1	The first system call	System call frequencies			
		Process ID			
		939	257	957	223
105	1	3	2	0	2
104	2	40	29	33	28
104	3	11	10	6	8
106	4	22	22	19	18
105	5	44	44	37	37
108					
112					

D.3 Grid Dataset

Table D3.1: Grid dataset used for Self-healing system test [139]

WaitTime	RunTime	NProcs	AverageCPUTimeUsed	UsedMemory	ReqNProcs	ReqTime:
1	1851	1	9	59868	1	31980
227	1852	1	8	57144	1	31980
226	1852	1	8	59924	1	31980
1	1851	1	8	59920	1	31980
1	1850	1	9	57164	1	31980
1	1850	1	8	57180	1	31980
290	1852	1	8	57228	1	31980
290	1853	1	8	59892	1	31980
290	1852	1	8	57160	1	31980
292	1850	1	8	59860	1	31980
292	1850	1	8	59884	1	31980
67	1850	1	10	59844	1	31980
67	1851	1	8	59900	1	31980
68	1851	1	8	59744	1	31980
1	94872	1	94777	152676	1	259200
1	95505	1	95396	152684	1	259200
2	95104	1	95008	152708	1	259200
1	94965	1	94869	152716	1	259200
1	1850	1	9	57152	1	31980
1	1860	1	9	57192	1	31980
1	94762	1	94674	152648	1	259200
2	68	1	25	88760	1	900
1	85	1	27	88500	1	900
2	1853	1	8	59908	1	31980
2	1853	1	9	57164	1	31980
2	1853	1	8	57136	1	31980

Wait Time	Run Time	N Procs	Average CPU TimeUsed	Used Memory	Req NProcs	Req Time:
1	1851	1	8	59916	1	31980
0	1850	1	8	57124	1	31980
1	1851	1	9	59868	1	31980
227	1852	1	8	57144	1	31980
226	1852	1	8	59924	1	31980
1	1851	1	8	59920	1	31980
1	1850	1	9	57164	1	31980
1	1850	1	8	57180	1	31980
290	1852	1	8	57228	1	31980
290	1853	1	8	59892	1	31980
290	1852	1	8	57160	1	31980
292	1850	1	8	57148	1	31980
292	1850	1	8	59860	1	31980
292	1850	1	8	59884	1	31980
67	1850	1	10	59844	1	31980
67	1851	1	8	59900	1	31980
68	1851	1	8	59744	1	31980
1	94872	1	94777	152676	1	259200
1	95505	1	95396	152684	1	259200
2	95104	1	95008	152708	1	259200
1	94965	1	94869	152716	1	259200
1	1850	1	9	57152	1	31980
1	1860	1	9	57192	1	31980
1	94762	1	94674	152648	1	259200
2	68	1	25	88760	1	900
1	85	1	27	88500	1	900
2	1853	1	8	59908	1	31980
2	1853	1	9	57164	1	31980
2	1853	1	8	57136	1	31980
0	1852	1	8	59892	1	31980

Appendix E

Source Code

```
% Best Feature Approximation Extraction
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Size of the data for each class
class_Z=[80 19 50 100 100 100 2500];
fid=fopen('DOS.txt','r'); read the data file
dim=41 ;   % Dimension of the features
hole_data=fscanf(fid,'%f');
hole_data=reshape(hole_data,dim,sum(class_Z));
hole_data=hole_data';
size(hole_data);
data_all=hole_data;

% just the first time 0.5 but after you need to change it
accordingly to important_features
% calculate the variance
i=1;
meann(i,:)=mean(data_all(1:class_Z(1),:));
varr(i,:)=var(data_all(1:class_Z(1),:));
for i=1:class_number
    meann(i,:)=mean(data_all((1+sum(class_Z(1:i-
1))):sum(class_Z(1:i)),:));
    varr(i,:)=var(data_all((1+sum(class_Z(1:i-
1))):sum(class_Z(1:i)),:));
end
meann_size=size(meann);
vect_remove=0;
vect_removeS=0;
% extracts the Best features
for j=1:dim
var_mod(j)=0;
min_var(j)=100^100;
for k=1:class_number
ifmin_var(j)>=(meann(k,j)-
mean(meann(:,j)))^2/(0.0000001+varr(k,j))
       min_var(j)=(meann(k,j)-
mean(meann(:,j)))^2/(0.0000001+varr(k,j));
end
    var_mod(j)=var_mod(j)+(meann(k,j)-
mean(meann(:,j)))^2/(0.0000001+varr(k,j));
end
var_mod(j)=var_mod(j)/15;
ifvar_mod(j)<thshold
vect_remove=[vect_remove j];
end
end
```

```
 % sort the important features
important_features=sort(var_mod,'descend');
vect_remove(1)=[];
data_all1=data_all;
data_all1(:,vect_remove)=[];
% the new data it will be data_all1
hole_data=data_all1;
// End extraction feature part
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
// Agents testPart
k=1;
control=0;

size_subclass=zeros(class_number,max(class_size));

%%%%%%111 class_size become max class_size%%%%%%%%%%%%%%%
data_t=hole_data(1:class_size(1),:);
for tr=1:(class_number-1)
data_t=[data_t;hole_data(1+sum(class_size_t(1:tr)):class_
size(tr+1)+sum(class_size_t(1:tr)),:)];   %%%%%%111     50
have to be change
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

data_control=hole_data(class_size(1)+1:class_size_t(1),:)
;
for tr=1:(class_number-1)

data_control=[data_control;hole_data(1+sum(class_size_t(1
:tr))+class_size(tr+1):class_size_t(tr+1)+sum(class_size_
t(1:tr)),:)];
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

dim=size(data_t,2);
incream=1;
check_part=1000;
while(check_part>0 &&incream<2)

% check that the coefficient is never negative

var1=sum(sum((data_t-
ones(sum(class_size),1)*mean(data_t)).^2));

%%%%%%111  class_zize have to be change
var2=0.5*var1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
forcn=1:class_number
control=0;
    x=data_t(                          sum(class_size(1:cn-
1))+1:sum(class_size(1:cn)),:);
var1=sum(sum((x-ones(class_size(cn),1)*mean(x)).^2));
var_class_i(cn)=var1;
var1=2*var1;
for k=1:(class_size(cn)-1)

% there are two options you can use the function
kMeansCluster or
% the function kMeansCluster_near_to_near_modified as
following:
%[y,b,c]=kMeansCluster_near_to_near_modified(x,k,1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% callingkMeansCluster

     [y,b,c]=kMeansCluster(x,k,1);

size_subclass(cn,:)=zeros(1,max(class_size));
          [k1, k2]=size(c);
for n=1:k1
              [Rf,Cf,Vo]=find(b==n);
size_subclass(cn,n)=sum(b==n);
            contro_var(n)=sum(sum((x(find(b==n),:)-
ones(sum(Vo),1)*c(n,:)).^2));
end
var2=var1;
var1=sum(contro_var(1:k));
var_list(k)=var1;
variation_var_list(cn,k)=abs(((var1-
var2)/max(var2,var1)));
if abs((var1-var2)/max(var2,var1))<coefficient(cn)
control=control+1;
end
if abs((var1-var2)/max(var2,var1))>= coefficient(cn)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% choose the treshold
control=0;
end
if (control >= 1 || var1==0)  % better to take control>=2
cn;
k;
break;
end
end
subclass_number(cn)=k;
var_class_f(cn)=var1;
```

```
ifcn==1
diction_class=[y,cn*ones(class_size(cn),1)];
repre_class=c;
else
diction_class=[diction_class;y,cn*ones(class_size(cn),1)]
;
repre_class=[repre_class;c];
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[class_assig,occur,erreur_position,time_spend_NN,trust_co
efficient,trust_coefficient2]=classify_control_data(repre
_class,subclass_number,data_t,class_size,size_subclass,1)
;
class_aff=ones(1,class_size(1));
for p=2:class_number
class_aff=[class_aff, p*ones(1,class_size(p))];
end
  [r c v]=find(class_assig~=class_aff);
if sum(v)~=0
erreur_location=unique(class_aff(c));
coefficient(erreur_location)=coefficient_reduction*coeffi
cient(erreur_location);
end
check_part=sum(v);
incream=incream+1;
end

//End test Part
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
// distance Matrix
function d=DistMatrix(A,B)
% The distance matrix is distance between points in A as
rows
% and points in B as columns.
% example: Spacing= dist(A,A)
% Headway = dist(A,B), with hA ~= hB or hA=hB
%          A=[1 2 3; 4 5 6; 2 4 6; 1 2 3]; B=[4 5 1; 6 2
0]
%  dist(A,B)= [ 4.69    5.83; 5.00 7.00; 5.48 7.48; 4.69
5.83]
%
% dist(B,A)= [ 4.69   5.00     5.48    4.69;
%          5.83   7.00     7.48    5.83]
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 [hA,wA]=size(A);
[hB,wB]=size(B);
```

```
IfwA ~= wB,  error(' second dimension of A and B must be
the same');
end
for k=1:wA
C{k}= repmat(A(:,k),1,hB);
D{k}= repmat(B(:,k),1,hA);
end
S=zeros(hA,hB);
for k=1:wA
S=S+(C{k}-D{k}').^2;
end
 d=sqrt(S);
// distance Matrix

//Data Control Part Analysis and Adaptation

function     [class_assig,     accurracy,erreur_position,
time_spend_NN,
trust_coefficient,trust_coefficient2]=classify_control_da
ta(repre_class,subclass_number,data_control,class_size_C,
size_subclass,n);

m1=sum(subclass_number);
[ertclass_number]=size(subclass_number);
[control_number dim]=size(data_control);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
forind=1:control_number
if n==1
erreur=sum(((repre_class-
ones(m1,1)*data_control(ind,:)).^2)');
            [valposi1]=min(erreur);
posi=posi1;
for l=1:class_number
ifposi-subclass_number(l)<=0
class_assig(ind)=l;
break
end
posi=posi-subclass_number(l);
end

[value_minposi_min]=sort(erreur);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
forind_t=2:m1
posi_min2=posi_min(ind_t);
forl3=1:class_number
ifposi_min2-subclass_number(l3)<=0
secd_posi=l3;
break
end
posi_min2=posi_min2-subclass_number(l3);
```

```
end
ifsecd_posi~=class_assig(ind)
break
end
end

trust_coefficient(ind)=erreur(posi1)/erreur(posi_min(ind_
t));

xp=posi1-sum(subclass_number(1:l-1));
N_subclass_min=xp;

xp=posi_min(ind_t)-sum(subclass_number(1:(l3-1)));
N_subclass_min2=xp;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

                [value_minposi_min]=sort(erreur);
class_out=ones(1,subclass_number(1));
Class_size=size_subclass(1,1:subclass_number(1));
forhh=2:class_number
class_out=[class_outhh*ones(1,subclass_number(hh))];
Class_size=[Class_sizesize_subclass(hh,1:subclass_number(
hh)) ];
end
class_out=class_out(posi_min);

          % the new metric
Class_size=Class_size(posi_min);
metric_num=1/value_min(1);

forhh=2:sum(subclass_number)
ifclass_out(hh)~=class_out(1)
second_class=hh;
break;
end
metric_num=metric_num+(1/value_min(hh));
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

trust_coefficient2(ind)=(1/metric_num)/value_min(second_c
lass);
end
if n==2
erreur=sum(                ((repre_class-
ones(m1,1)*data_control(ind,:)).^2)./
(max(0.1*abs(repre_class),
0.1*abs(ones(m1,1)*data_control(ind,:))).^2) )'  );
        [valposi]=min(erreur);
for l=1:class_number
```

```
ifposi-subclass_number(l)<=0
class_assig(ind)=l;
break
end
posi=posi-subclass_number(l);
end

end

%stop time
End

%average of stop time
occur=0;


for u=1:class_number
    occur=occur+sum(   class_assig(1+sum(class_size_C(1:u-
1)): sum(class_size_C(1:u)))==u  );
end
accurracy=occur/(sum(class_size_C));

class_aff1=ones(1,class_size_C(1));
for p=2:class_number
class_aff1=[class_aff1, p*ones(1,class_size_C(p))];
end

[r1c1v1]=find(class_assig~=class_aff1);
erreur_position=c1;
time_spend_NN=sum(subclass_number)/(sum(class_size_C));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%trainingpart%%%%%%%%%%%%%%%
% Max_Ac=zeros(1000,100);
Max_Accu=-1*ones(size([0.01:0.01:0.999],2),10);
thshold=0;
class_number=2;
accurrency_graph=zeros(1000,100);
paper_Approximation_detail_var;

forll=2:size(important_features,2)
%1:size(important_features,2)       THIS   BECOME       for
ll=size(important_features,2):-1:1
ll
thshold=important_features(ll);
paper_Approximation_detail_var;

accurrency=0.5;
Max_Accu_va=-1;
accu=0;
```

```
coo=0;
h1=1;
for co=0.01:0.01:0.9999
coo=coo+1;
accurrency_graph(coo,ll)=0;

forcor=0.01:0.5:0.999
h1=1;
coefficient=co*ones(class_number,1);
coefficient_reduction=cor;
while(accurrency<1.0 &h1<8)% change it to 20 or bigger if
you can
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ifaccurrency>=Max_Accu_va
Max_Accu_va=accurrency;
Max_Accu(coo,ll)=accurrency;
accurrency
optimal_assig=class_assig;
subclass_number;
trust_coefficient2_max=trust_coefficient2;
trust_coefficient_max=trust_coefficient;
erreur_position_max=erreur_position;
end

h1=h1+1;
accu=accu+accurrency;

ifaccurrency>=Max_Accu(coo,ll)
Max_Accu(coo,ll)=accurrency;
accurrency_graph(coo,ll)=accurrency;
end
end
end
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
End of training part


%%%%%%%%%%%%%%%Analysis and Adaptation Part%%%%%%%%%%%%%%%
load-matprope.mat
%class_test=[50 75 50 25 1050];
fid=fopen('Readdata.txt','r');
dim=41;
testdata1=fscanf(fid,'%f');
fclose(fid);
testdata1=reshape(testdata1,dim,length(testdata1)/41);
testdata1=testdata1';
% a=dataset(testdata1);
```

```
fid1=fopen('readTestdat.txt','r');
dim=41;
testdata2=fscanf(fid1,'%f');
fclose(fid1);
testdata2=reshape(testdata2,dim,length(testdata2)/41);
testdata2=testdata2';
testdata2_orignal=testdata2;
testdata2(:,vect_remove)=[];
% b=dataset(testdata2);
class_vect=-1;
ANAmatrix=[];
count=0;
cc=0;
for j=1:length(testdata2)
ANAmatrix 1=testdata2(j,:);

[class_assig,      trust_coefficient,trust_coefficient2]=
classify_control_data1(repre_class,subclass_number,ANAmat
rix1,1,size_subclass,1);
class_vect=[class_vectclass_assig];
end
class_vect(1)=[]
i=1;
%start search
indicator_ANAmatrix=zeros(1,size(ANAmatrix,1));
for i=1:size(ANAmatrix,1)
for n=1:size(testdata1,1)
if                                                    sum(
testdata2_orignal(i,:)==testdata1(n,:))~=size(ANAmatrix,2
) &&class_vect(i)~=5

indicator_ANAmatrix(i)=1;
end
end
end
testdata1_LV=testdata1;
for i=1:size(ANAmatrix,1)
if indicator_ANAmatrix(i)==1
testdata1_LV=[testdata1_LV;testdata2_orignal(i,:)];
testdata2_orignal(i,:);
end
end
dataset1=testdata1_LV;

%%%%%%%%%%%%%%%%%%%%End and Adaptation %%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%Self-healing system%%%%%%%%%%%%%%%%%%%%%
```

```
%% replication of data
component_records=[2826];
fid=fopen('data    withoutlabel.txt','r');    %    system
component s
dim=7;
component_data1=fscanf(fid,'%f');
%hole_data=reshape(hole_data, sum(class_size_t),dim);
component_data1=reshape(component_data1,dim,sum(component
_records));
component_data1=component_data1';
component_data2=component_data1;
component_data3=component_data1;
%% changes in data
tic
i=0;
for i=1:1:100
x = floor(rand()*7);
if x < 1
    x = 1;
end
y = floor(rand()*2826);
if y < 1
    y = 1;
end
component_data1(y,x)
% corrupt the records
component_data1(y,x)     =     component_data1(y,x)     +
floor(rand()*10);
component_data1(y,x)
% check for corruption and repair it
flag = 1;
while(flag)
% repeat
tic;
if sum(sum(component_data1 - component_data2)) ~=0
for l = 1:7
for k = 1:2826
ifcomponent_data1(y,x) ~= component_data2(y,x)
ifcomponent_data3(y,x) == component_data2(y,x)
component_data1(y,x) = component_data2(y,x);

end
end
end
end
end
toc;
    m=1
% validation of repair
% tic;
```

```
for y = 1:7
for x = 1:2826
ifcomponent_data1(x,y) == component_data3(x,y)
flag = 0;
end
end
end
%toc;
    m=2
end
end
%%%%%%%%End Self-healing system%%%%%%%%%%%%%%%%%%%%%%
```

Appendix F

Cancer Detection Simulation Test Results

Cancer Dataset

Cancer dataset used in the simulation test has two classes: malignant and benign. This data set consists of 9 real-valued features F1,…,F9 computed for each cell nucleus. 569 instances were recorded for two types of diagnosis, M for malignant and B for benign. The class distribution of 682 instances indicated 444 benign and 238 malignant, and there is no missing values. The features were computed from a digitized image of a fine needle aspirate (FNA) of a breast mass.

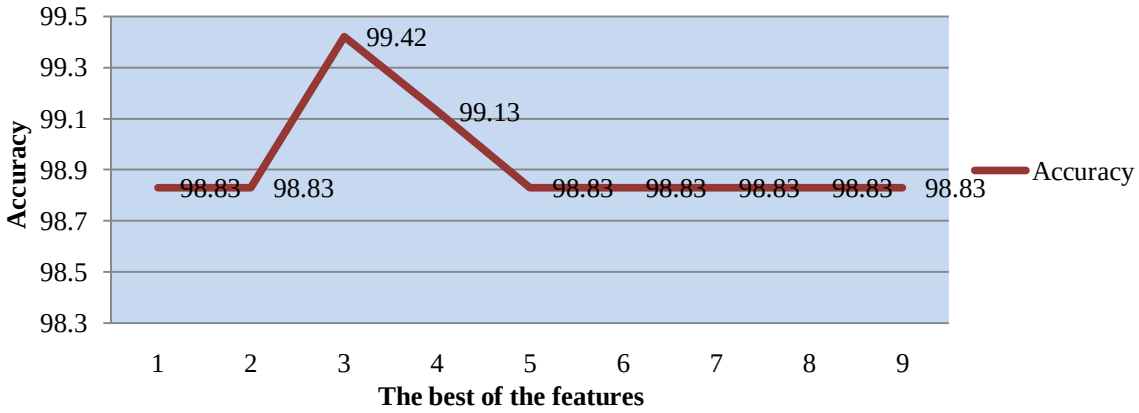


Figure F.1 :Accuracy changing with best of the features 1,...,9

The result is compared with other recent results and the comparison is as shown in Table F.1. The maximum accuracy is achieved with by using features with the most variance that affects the differences between the two classes i.e. the benign and malignant. These features are feature1, feature 2 and feature 3 as shown in Figure F.1. The accuracy is investigated at this point i.e. F=1, F=2 and F=3, α=0.92 and the change in accuracy according to the value of α is shown Figure F.2.

Table F.1: Comparison between in breast cancer results

Ref	FR	Accuracy
[142]	5.63%	94.37%
[143]	2.93%	97.07%
Current	0.58%	99.42%

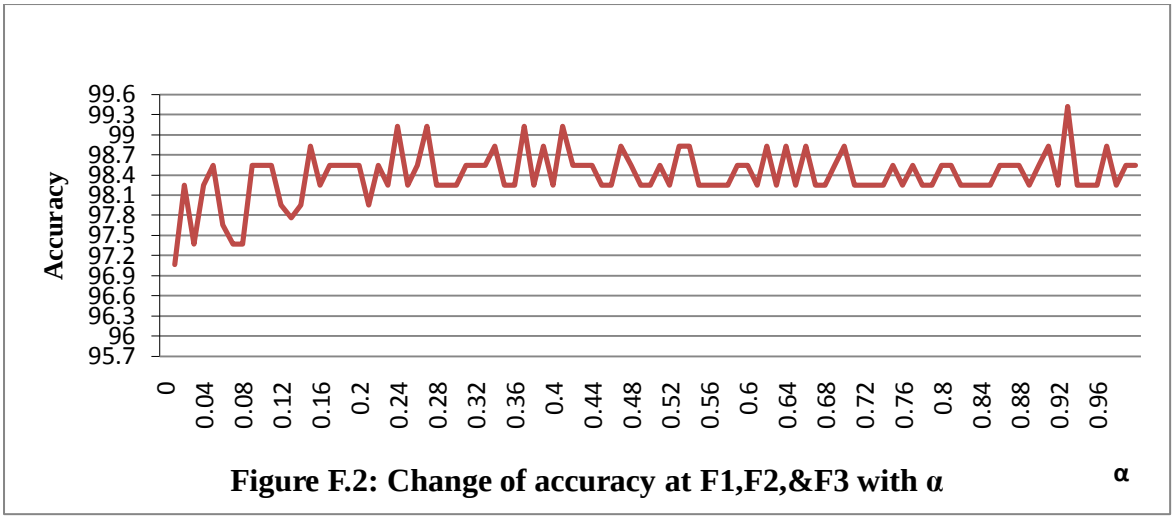


Figure F.2: Change of accuracy at F1,F2,&F3 with α