Intrusion Detection and Prevention: Immunologically Inspired Approaches

Devid Pipa

Technical Report

RHUL-MA-2008-01

15 January 2008

Department of Mathematics

Roal Holloway, University of London

Egham, Surrey TW20 0EX, England

http://www.rhul.ac.uk/mathematics/techreports

Intrusion Detection and Prevention:
Immunologically Inspired Approaches

Devid Pipa

Supervisor: Alexander W. Dent

Submitted as part of the requirements for the award of the MSc in Information Security at Royal Holloway, University of London.

I declare that this assignment is all my own work and that I have acknowledged all quotations from the published or unpublished works of other people. I declare that I have also read the statements on plagiarism in Section 1 of the Regulations Governing Examination and Assessment Offences and in accordance with it I submit this project report as my own work.

Signature:

Date: 05 – October -2007

Dedication

To my beloved parents who helped me get this far and were always there for me. They believed in me even when I did not believe in myself and never let me down in any way

Acknowledgments

The author gracefully acknowledges the help of Professor Genc Sylcebe of the Universit of Tirana and the Mother Teresa Hospital, for the very helpful and patient explanations and insights into immunology.

A special thanks also goes to Dr. Alex Dent for his very kind help, endless patience and understanding.

Also a special thanks to Professor Peter Wild and Dr. Zbigniew "Chez" Ciechanowicz for their very good will and helpfulness.

A special thanks goes to fellow student Igor Yuklyanyuk for some of his good insights.

Royal Holloway University of London
Information Security Group

Intrusion Detection and Prevention: Immunologically Inspired Approaches

By

Devid Pipa

September 2007

Table of Contents

Introduction 7
Motivation from Immunology 9
Aims & Expected Outcomes 9
A general overview 10
Chapter 1 12
Human Immunology 12
1.1 An Overview 14
1.2 The Layers 15
1.3 Innate and Adaptive Immunity 16
1.3.1 The Innate Immune System 17
1.3.2 The Adaptive Immune System 18
1.4 Adaptive Immunity in Detail 21
1.4.1 Pathogenic Substances 21
1.4.2 Lymphocytes 22
1.4.3 The Recognition Process 22
1.4.4 Specificity of Lymphocytes 24
1.4.5 Self/Non-Self Discrimination & Negative Selection 24
1.5 Adaptation and Learning 25
1.5.1 B Lymphocytes 26
1.5.2 Autoimmune Responses 28
1.5.3 T Lymphocytes 28
1.6 A Summary 29
Chapter 2 31
Issues, Principles & Guidelines 31
2.1 Information Security Issues 33
2.2 Design Principles of Computer Security 34
2.3 Principles for Computer Immune Systems 35
2.4 Detection & Prevention 38
2.5 The Security Policy of the Immune System 39
2.6 Self & Non-Self In a Computer 40
2.7 Misuse or Anomaly Detection 41
2.8 Issues to be Addressed 42
Chapter 3 44
Current Research & Literature 44
3.1 Architecture Proposals 46
3.1.2 Limitations 49
3.2 The Methods and Algorithms 51
3.2.1 Defining Self 53
3.2.1.2 Analyzing with Respect to Knowledge of Self 55
3.2.1.3 Loss of Information 56
3.3 Anomaly Detection Method 57

3.3.1 Some Analytical Points 62
3.3.1.1 Splitting the Data Stream 62
3.3.1.2 A Summery of the Splitting Process 68
3.3.1.3 Detector Set Size 69
3.3.1.4 Detector Set Size – A quick summarization 73
3.3.1.5 The Existence of Holes 74
3.3.1.6 The Existence of Holes – A Summary 77
3.4 Negative Selection Methods 79
3.4.3 Linear Time Algorithm 81
3.4.4 Greedy Algorithm 82
3.4.5 Counting the Holes Method 83
3.5 A General Summary 84
Chapter 4 85
Evaluation, Analysis & Ideas 85
4.1 System Calls for a Sense of Self 87
4.2 The Architectures 88
4.2.1 The Architectures – Some conclusions 89
4.3.1 Splitting the Data 90
4.3.2 Detector Set Size & Holes 91
4.3.3 Hypermutation 91
4.4 Dual Authentication 92
4.5 General Conclusions and Opinions 93
Bibliography 95

Introduction

In the area of information security, one of the main issues to be dealt with, is intrusion detection. As a term, this refers to the process of the detection of an anomaly that would cause undesired or unwanted effects. These anomalies are generated by malicious users, through the use of malicious software, such as viruses, worms, etc, or through the direct exploitation of holes in the system. Holes, are a result of design and implementation oversights in software. These holes are then exploited by malicious users. Intrusion detection systems have the purpose of covering such holes with the appropriate protection. A large selection of intrusion detection software is available nowadays, including firewalls, intrusion prevention systems (IPS), virus scanning software etc. The systems have sets of functions that provide some level of security that is missing in the software they are protecting. These pieces of protective software are placed as an extra layer on top of the base software, such as the operating system, to provide an extra layer of security.

The functions performed by these types of software include the following are to ensure that the three issues of information security, namely **confidentiality**, **integrity**, **availability**, are addressed, and the appropriate protection level is placed in order to meet the requirements in terms of the three above issues. Also the two

not so popular issues of **accountability** and **correctness** need to be addressed.

Some of the functions performed by these systems are:

- Examine and monitor system and user activity
- Examine configuration files
- Monitor overall integrity of the system
- Track security policy violations

The main concern of these systems is the enforcement of the security policy, however as stated in the [18] and [16], the security policy is often incomplete due to oversights caused by the complexity of nowadays systems. It is not possible to consider all eventualities. The problem with these traditional methods is that they are based on a static set of rules and can only offer the protection that is stated in each specific rule. These rules are, each, highly specific to one kind of attack. If an attack is performed in a novel way for which a rule does not exist, the system will not be able to offer the right protection. Due to this, a more dynamic method for the protection of the systems we rely so much on is needed.

Motivation from Immunology

According to [13], computer security can be viewed as a process of discrimination between authorized actions, legitimate users, etc, and intrusions such as viruses, trojans, etc. The immune system of the human body has been performing such an action for a much longer time and it is very likely that it has developed a set of techniques and mechanisms that are, in comparison, a great deal better than the ones used in the current computer security systems. And it certainly has, as in the opposite case, the human race would be extinguished by now.

The immune system of the human body is a collection of mechanisms and techniques that offer an overall defense for the organism in a both distributed and localized manner. These are specific and non specific mechanisms. The specific ones, offer a level of defense against one single type of threat, whereas the non specific ones have a more wide range. This is much like the defense mechanism in the information security world such as specific ones, through virus signatures and non specific ones such as firewalls and encryption mechanisms. The specific ones, are a good way of defense towards known and previously encountered attacks, for which a signature as been developed. These however have a difficulty in keeping up with the dynamically changing attacks. The non specific ones, do offer a good level of general efense, however they are static. They form a preventive barrier in the prospect of intrusion and are not able to detect a currently ongoing intrusion.

The immune system offers levels of defense for the organism that are very dynamic. They prevent known intrusions and are also able to dynamically adapt themselves in order to detect ongoing ones. This latter concept is the one of interest to this study.

The idea of applying immunological principles to the systems of computer security was introduced in 1994 by Jeffrey Kephart in the design for an immune system for computers and networks [15].

Aims & Expected Outcomes

The aim of this study is to primarily gain a good knowledge of the human immune system and the techniques it uses for the detection and prevention of intrusions in the organism. The next step is to make a research into the current advancements of technology towards the embedding of these techniques in the systems of

intrusion detection for computers. Also a number of concepts, acting as guidelines, for the design of such systems will be looked into.

This study is an attempt towards the individuation of the progress of research towards the goal of embedding human immune system principles in the are of intrusion detection. The intent is to gain a good knowledge of both areas and derive a set of conclusions ad evaluations for this.

Once the appropriate research has been conducted, the study will present an evaluation for the number of concepts that have been embedded form the immune system into the intrusion detection methods and techniques designed.

The study will also present a set of conclusions in an evaluative manner for the problems that might be encountered by these systems in the prospect of intrusion detection. In this, an attempt will be made to present a number of personal ideas.

A general overview

I would like to present a general overview of the contents of this study so that the reader can gain an idea of what there is to follow in the next chapters.

The initial step, as previously mentioned, will be to take a deep look into the human immune system. A set of different concepts will be introduced here in the context of how the immune system of the human body provides the defense mechanisms at cellular level. The concepts of innate and adaptive immunity will be looked into. The latter of the two is the one of interest to the field, therefore details will be provided for the explanation of its techniques and methods. Here the reader will be introduced to the concepts of self/non-self discrimination and negative selection. These are the concepts that are of importance to the current research.

Further in the study, we will deal with gaining a knowledge of the issues of information security and what these mean. Also, the reader will be presented with a collection of guidelines to be kept in mind in the design of these systems. The next step will be to individuate the concepts and issues that need not be looked into through the literature research.

The next chapter will deal with presenting a series of methods and techniques that have been designed by

researchers in order to analogize these concepts on self/non-self discrimination and negative selection and adapt them towards addressing the issues of information security. In some cases, a detailed description will be given for these methods, however the details of these methods do not have a major bearing on this study as we are only trying to evaluate the general idea. The details will be extracted from a selection of sources and presented more for an illustrative purpose.

The final chapter will deal with providing a set of conclusions derived form the research conducted in the previous part of the study analyzing the extent to which the concepts have been embedded from human immunity to computer security.

Chapter 1

Human Immunology

The study cellular and molecular immunology is one the the newest topics of the biological and medical fields. This science deals with the study of the **immune system of the human species**. The following chapter will deal with getting a good overview and understanding of the immunological system of our organisms.

With the discovery of the different roles of molecules, cells and organs in the years 1955 to 1960, immunology went from being simply a phenomenological science to forming a major medical discipline of its own [1].

The obvious question at this point would be; **Why should the world of information security concern itself with immunology**? The answer to that is quite simple and straight forward. Our computer systems are reaching complexity of magnificent proportions. Programs are constantly increasing in size and operational

complexity. This has led to higher unpredictability in their day to day tasks and also unreliability in the context of their security. Although we are led to believe that almost all software undergoes very thorough testing phases in the search for bugs in the code and vulnerabilities, experience has taught us this is often not enough.

Because of all of the above considered factors we need defense mechanisms to ensure the **confidentiality**, **integrity** and **availability** of all these systems and the data contained in them.

There are many different solutions to enforce such security. We all know about **firewalls** and **intrusion detection and/or prevention systems**, **anti virus software** etc. However, as much as they are capable of detecting old or better, **known** attacks they are very vulnerable to new or better, **novel** attacks. These systems rely on the knowledge of known signatures for the detection and prevention of intrusions. This makes them vulnerable to attacks for which a specific signature does not exist.

At this stage it becomes quite obvious to try and learn from someone or something that has been fighting intrusions for much longer than we have. The **natural immune system** is a complex system that has been detecting and fighting intrusions in order to protect our bodies since the start of mankind. This system is very good at fighting known intrusions and it also has the capability of adapting itself to detect and neutralize new ones. Clearly a system with so much experience could teach us a thing or two about detecting and combating intrusions.

1.1 An Overview

In medicine. and all related areas, the term **immunity** refers to protection from disease or infection. The term **immune system** refers to that set of **molecules, cells** and **organs** which perform the needed functions to produce such immunity. The collective and coordinated response of this system upon the detection of a threat is called an **immune response**.

Our organism is under constant threat from harmful substances and foreign agents. In the medical world these are called **pathogenic substances** (also known simply as pathogens). Pathogens are infectious agents that cause a deterioration of the state of well being of their host. The duty of the immune system is to recognize these pathogenic substances and neutralize them while trying to cause the **least damage** to the body [1][2].

At this stage it is important to put some emphasis on the above point. **Neutralization of the pathogenic substance while causing the least damage**. The immune system performs a very indispensable set of tasks in order to keep the organism it belongs to, **healthy**. However, it has to be noted that there is a very fine line drawn here. The immune system has the potential to fully destroy such organism. This can occur in two different ways. Upon the introduction of a pathogen in the system, an **under reaction** by part of the immune system will allow the pathogen to overpower the organism that it penetrates. This will lead to the deterioration of this organism and eventually to its death [1][2].

There is also the concept of **overreaction**. This is the case when the immune system reaction towards an intrusion is of much larger magnitude then actually needed. This eventuality would also lead to deterioration of the individual's health and if continued in time could also lead to total neutralization of the organism itself. In one simple word, death [1][2].

The immune system is a highly complex defense mechanism. It is a self maintaining and self monitoring set of organs which is able to respond to any challenge. It is also self regulating in order for its response to a challenge not to be harmful to the organism's health [3].

This system involves a set of organs and cells which work in very sophisticated ways to create the best possible response towards the detection and elimination of a threat. There is a number of concepts and ideas to be deduced from the workings of such system that can be very beneficial to the world of information security.

The architecture of this system is a multi-layered one. There are multiple defense mechanism working at different levels in the body, each providing indispensable defense. The aim of this complex set of defense

mechanisms is to maintain **homeostasis**. This word is derived form ancient Greek and is formed of the following two words:

Homoios – same, resemblant, alike
Stasis – to stand, a posture, a state

In immunology, this term, refers to the organism's property to maintain a coherent state of tranquility and stability. This state is very relative and is very variable even when the organism is considered to be in a healthy state. The reason for this is the constant threat that any body is under. This threat being a large amount of pathogenic substances which attack our system at any given time. However the immune system tries to maintain this state, to the best of it's possibilities, within certain boundaries.

1.2 The Layers

As mentioned above, there are a number of mechanism that form the immune system. Each of these perform a set of tasks in order for the detection and neutralization of pathogenic substances. These mechanism perform these task at different levels within the body. Some of them act as an actual physical barrier. Other mechanisms perform functions to create a physiological barrier for pathogenic substances. These are factors such as acidity or high body temperature, which create an inhospitable environment for pathogenic substances. Under these living conditions, such substances face a much higher level of difficulty towards their evolution. However, if these substances menage to perpetrate these types of defenses, cellular immunology comes into play. These are the defenses that form the innate and adaptive immune system. This type of defense is the one that we are most interested in, and for the purpose of this study, in particular, the adaptive immunity.

Breaking down the levels of immunity present in this system, we would have the following [1]:

- **Physical** – E.g. Skin. This organ plays a very important role. It acts as shield towards any pathogenic substances trying to intrude into our system.

- **Physiological** – E.g. Temperature, Sweat Glands. These create inhospitable living conditions by lowering the pH level and creating a higher acidity on the surface. These conditions are very

unfavorable towards the evolution of any pathogenic substances.

- **Cellular** – If one or more pathogenic substances perpetrate the previous two levels of defense, cellular immunological mechanism come into play. These act at the very cellular level through a process known as **intra-cellular** killing. The immune system has a large number and variety of cells to its disposal to serve this purpose.

1.3 Innate and Adaptive Immunity

Immunologists like to differ between two kinds of immunity. Innate and Acquired immunity. There is a big difference between the two, in context of the different types of defense they offer against pathogenic substances [2] [3].

a) **Innate Immunity** – This is simply the immunity that we are born with. It is not conditioned by a previous contact with a pathogenic substance. It is composed of a complexity of factors which create the first barrier from a foreign intrusion. In the evolution of a species this is the first immunity to be created. In humans this immunity is created in the fetus during pregnancy. This includes the anatomic physical barrier that we know as the skin.

b) **Adaptive Immunity** – This is the immunity that is created by the organism itself through contact with a pathogenic substance. It is also known as **Acquired** or **Gained Immunity**. The agents of this type of immunity are high specific. This means that a set of agents is only built to respond to a certain type of pathogenic substance.

Innate immunity can be considered as a traditional intrusion detection method. It is light-weight and has a broad spectrum, however there is no capability there for the detection of novel attacks. Therefore, this study will base itself around the principles behind the adaptive immune system.

1.3.1 The Innate Immune System

This part of the chapter will only provide a short description of the innate immune system. The redear is asked to refer to [1] [2] [3] for more detail.

As stated above the innate immune system is the set of immunological knowledge that our body is born with. In this is included all the immunological resistance formed through the evolution of the species and passed on generation after generation to the present date. This type of immunity is ready and functional from the moment that we are born. It s not in any way conditioned by a prior contact with a pathogenic substance. The limitation of this system is that it is non specific to any pathogenic substance, therefore it has a relatively narrow gamma of action towards the different types of intrusions. This is only effective against already known pathogens.

Immunology includes in this system the physical and physiological barriers mentioned in the previous section. However, once a pathogenic substance menages to perpetrate these barriers, cellular mechanism come into action. Here is where the process of intra-cellular killing comes into action [1].

This type of immunity is very similar between individuals given that all the information contained is purely passed on from individual to individual through generations. It can only defend against pathogenic substances previously encountered by the individual's predecessors.

Also, this type of immunity is constant throughout the lifetime of an individual. It has no ability to adapt towards combating novel pathogens.

Given the low specificity of innate immunity and it's stated inability to individualize novel pathogenic substances, it is not feasible to continue investigation towards it. We are interested in the individuation of novel attacks. Protection from previously known attacks is easily implementable in nowadays intrusions detection and prevention systems. For the purpose of recognition and defense from novel attacks, we need to look at the procedures and mechanism of the Adaptive Immunity present within our bodies.

1.3.2 The Adaptive Immune System

This part of the research is extracted from [12] [1] [2] [3], and to some extent, the initial chapters of [10] and

[11] have been consulted.
As stated above, this type of immunity, is the one that is able to detect and protect the organism from novel attacks. As opposed the the innate immunity, adaptive immunity has highly specific responses. There are two main properties that make the difference between this type of immunity and the innate one [1]:

- Gained immunity is high specific towards one type of pathogenic substance as opposed to the innate immunity which does not have such property.

- The evolution of gained immunity towards a specific type of pathogenic substance requires a prior contact with the pathogenic substance.

Upon the event of a pathogenic substance entering the organism there are a number of steps taken towards the recognition and neutralization of the pathogenic substance and the evolution of the immunity towards that type of substance.

The first step is the recognition of the existence of a threat. So, once a novel pathogenic substance has perpetrated into the organism, there is a procedure called **recognition process**.

Once the recognition process is conducted successfully, and a threat from an unknown pathogenic substance is individuated, the immune system takes action towards fighting this threat.
The first encounter with an unknown type of pathogen is known as the **primary immune response**. This response takes action and eliminates the threat.
The next step taken by the immunological system is a **learning process** that is performed after the pathogenic substance has been cleared. In this process the immune system will **memorize** a fraction of the cell that successfully recognized the pathogenic substance. This is called the **evolution of immunity**. Through this process, the immune system trains itself towards the recognition of novel pathogens, so in future re occurrences of a threat by the same or a similar type of pathogenic substance, the response will be much faster and effective.
There is also something known as a **secondary immune response**. This is the response made by the immunological memory of the system towards a pathogenic substance. This response is much more efficient, as the immune system already has the necessary information about how to fight this substance and the duration in time from the moment of the intrusion till its full elimination is much shorter and homeostasis is maintained much more optimally than in the case of the primary immune response.

So, a conceptual summarization of these procedures would be; Upon the detection of a malicious intrusion in the body the system reacts with an immune response. There are two types of immune responses [10] [11]:

- **Primary immune response** – The body deploys a series of agents to combat the intruder. This leads

to the neutralization of the threat. During this phase, the deployed agents must adapt in a certain way to meet some requirements in order to succeed in the neutralization of the threat. Once the pathogenic substance posing such threat is eliminated, a fraction of the recognizing agent is saves in the immunological memory for use in case of future occurrences.

- **Secondary immune response** – This is the response created by the immune system in the eventuality of a threat by a pathogenic substance that was previously combated or a similar one. Less time and energy has to be spent in the recognition and adaptation towards meeting the requirements for fighting this substance, therefore the response if quicker and the immune system requires less time to deploy more, already adapted agents, for the purpose of the neutralization of this intrusion.

The following graph, Figure 1.1, shows a graphical representation of the primary and secondary immune responses, visualizing the difference in lymphocyte number and time. The antibody level represents the number of lymphocytes allocated towards combating the intrusion.

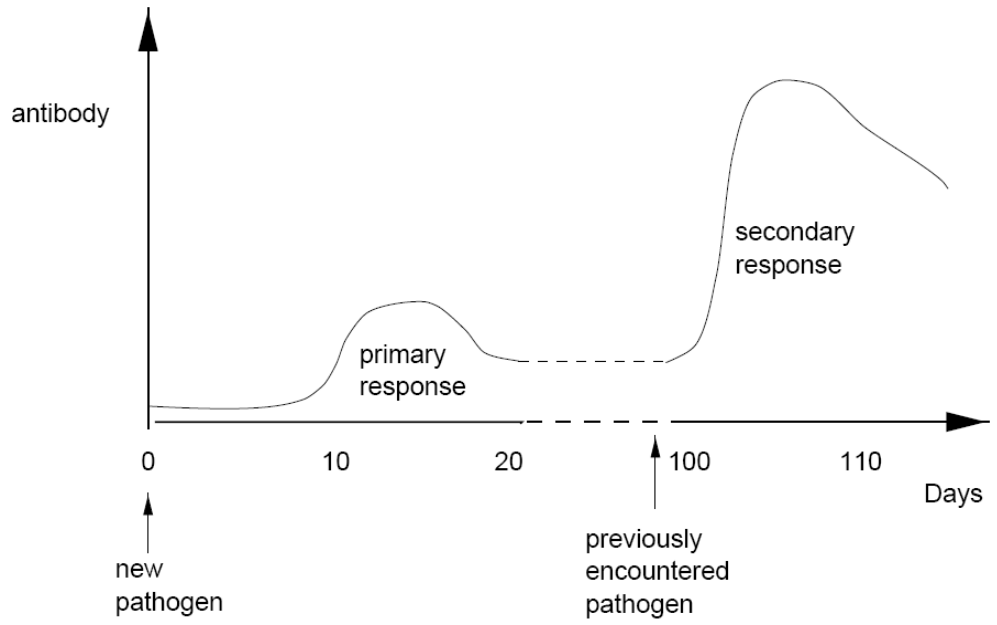


Figure 1.1 Graphical representation of the primary and secondary immune responses in the human immune system. Extracted from [10].

Even the adaptive immune system however is not absolute. By no means can it be called perfect. But, relatively to other types of defense mechanism, it is the best defense mechanism that we know. It does not provide an absolute defense against harmful substances. There are many pathogens which this system fails to

combat successfully. However, the main property of this system that is of most interest to this study is its ability to detect an intrusion from a previously unknown pathogenic substance.

The adaptive immune system uses a set of cells which act as agents for this system and detect pathogenic substances based on recognition of certain protein structures [12]. Once the protein structure is recognized these cells bind to the pathogenic agent and eliminate it.

There is a very important notation to be made at this point. All cells in the body have a protein structure. It is the duty of the immune system to create some agent cells, called lymphocytes, on the basis that they should not bind to protein structures of cells which are part of our organism. This process is better known as **discrimination between self and non-self**. This is a very important concept which has a major role in this study and shall be looked at in more detail in the next section [12].

These agent cells, in the world of immunology, are known as **lymphocytes**. These form part of the white blood cells. Being part of the blood stream, is a very important attribute for lymphocytes. This allows them to circulate in the body and makes them able to reach any organ or cell in any system of our organism.
Lymphocytes are very diverse and have a high specificity to pathogenic agents and substances. They act as detectors within the body and identify what is classified and non-self. The large number of these detectors and also the distributed architecture of this level of the system creates an efficient mechanism for the detection and elimination of pathogenic substances or microorganisms.

Lymphocytes also play a role in the neutralization of pathogenic substances with the aid of other microorganisms of the body called macrophages. This however is not of interest to this study and will not be covered. We will only deal with the detection process.

The next sections of this chapter present a much better and more detailed view of the concepts and mechanisms used by the immune system for the successful protection of the organism.

1.4 Adaptive Immunity in Detail

At this point the reader will recall from the previous section the attributes and general processes used by the adaptive immune system in the successful detection and elimination of pathogenic substances. At this point I would like to go into some more detail about the procedures involved in this detection. I will explain in some more detail some of the components that form this immunity and take part in the detection process. Also I will pay some attention towards pathogenic substances and what properties of them make them recognizable. Furthermore, there will be some investigation into the phenomenon of the self and non-self discrimination.

1.4.1 Pathogenic Substances

Extracted form [1] and [12]
A pathogenic substance is a general term referring to all cellular based microorganisms that if introduced in our organism would be considered as harmful, non-self substances. The introduction of such substances into our organisms would cause the triggering of an immunological response. Like all cells, the cells of a pathogenic substances present certain protein structures. In the case of pathogenic substances these are called antigenic protein structures, or simply, for convenience **antigens** or **epitopes**. These protein structures are found on the surface of the pathogenic cell.
The cells of the adaptive immune system are built in such a way to be able to recognize a pathogenic substance form any other type of substance through the patterns in these antigenic protein structures.

The reason for the name “antigenic protein structure” is the fact that these structures go against the genetic code, and are not similar to the protein structure of self cells of the system.

These pathogenic substances are detected by a set of cells that form part of the white blood cells and act as agents for the immune system. These cells are called lymphocytes. Let us take a look at these in the next section.

1.4.2 Lymphocytes

Extracted from [12].

As mentioned before lymphocytes are cells that act as agents of the immune system. These cells form part of the white blood cells and circulate through the blood system. Immunologists argue that this is a very strategic position for these agents as, by circulating in the organism's blood vessels, they are able to reach any organ, extremity or position with the organism that they belong to. The next step is to look at how these agents are able to bind themselves to the antigens mentioned in the previous section.

Lymphocytes are a unicellular microorganism. On the surface of these cells, a series of receptors are found. These receptors are the component that take part in the the recognition process towards a pathogenic substance. They perform such action by binding with complementary antigenic protein structures on the surface of the pathogenic substance. There are different types of lymphocytes present within the blood stream and we will look at these in more detail later.

1.4.3 The Recognition Process

Extracted from [12][10].
A recognition process occurs when a pathogenic substance is individuated within the organism by the lymphocyte cells of the immune system. Now, the question to be asked at this point is obviously; How is it possible for these cells to recognize something that they have not seen before and still be able to detect it as foreign and threatening?

I would ask the reader at his point to recall the concepts of antigenic protein structures found on the surface of pathogenic substances and the receptors found on the surface of lymphocyte cells.

A **recognition**, or for the purpose of this study let us call this **detection**, event occurs when chemical bonds are established between these receptors and the antigens. For a chemical bond to be possible between these two components, they need to be complementary in structure to each other. This bonding procedure can be seen as a matching through pattern recognition.

The bond that is established between the antigen and the receptor of the lymphocyte is almost never perfect, especially in the case of introduction on novel pathogenic substances. The extent to which this bond between the complementary structures is created, the strength of it if you like, is called **affinity**. The affinity establishes a threshold level for the binding process. This means that it establishes how strong, or to what level, these structures must be complementary to each other in order for the event to be classified as a

detection of a pathogenic substance.

The next figure, Figure 1.2, displays a visual representation of a lymphocyte and an number of pathogenic substances showing the different affinity levels and bonding process.

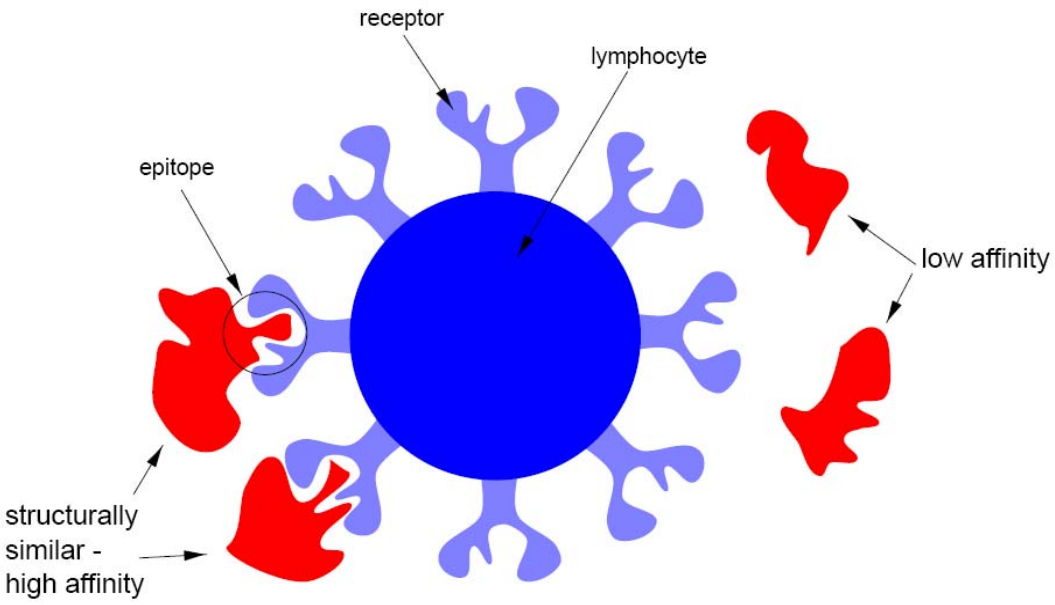


Figure 1.2 A visual representation of a lymphocyte and it's affinity to a set of pathogenic substances. Receptors on the lymphocyte are identical and they are high specific to certain types of pathogenic substances. This is the affinity level. Extracted from [10].

1.4.4 Specificity of Lymphocytes

Extracted from [10][11][12].

The reader will know by now, that lymphocytes are part of the adaptive immune system. This system, as stated previously, has the property of high specificity towards pathogenic substances. Meaning that its agent cells, the lymphocytes, are each specific to a certain type of pathogenic substance. This phenomenon constitutes what immunologists call **high specificity** of the adaptive immune system. One lymphocyte can only bind to a set of antigenic structures. It is not possible for a lymphocyte to bind itself or better, recognize, a pathogenic substance that presents antigenic structures outside of the ones that this lymphocyte is trained to recognize. Let us shed some more light into why this is and how it is accomplished by the immune system.

As mentioned in previous sections lymphocytes bond to the antigenic structures of pathogenic substances through the receptors situated on the surface of such lymphocytes.

The receptor structures on one given lymphocyte are identical to each other. However they differ between different lymphocytes. This is what makes a lymphocyte specific to a given type of antigenic structure.

This property of lymphocytes is very often referred to in immunology as **monospecificity** of lymphocytes.

The next issue to be dealt with at this point is the ability of lymphocytes to recognize pathogenic substances, and how this is accomplished when the actual structure is unknown. Let us take a closer look at this in the next section.

1.4.5 Self/Non-Self Discrimination & Negative Selection

Extracted from [10][11][12].
In cell mediated immunity, the cell is the base structure. Pathogenic substances are made up of cells, however, so are all the other parts in the body. As the reader will be able to recall, lymphocytes, recognize foreign cells through the antigenic structures presented by them. The issue that the immune systems deals with at this stage is that, all cells, including the ones belonging to human body, present protein structures on their surface. The question to be asked at this point is: How is it possible for lymphocytes not to bond to protein structures presented by cells that belong to our organisms?

The solution to the above problem is accomplished by the process known as **discrimination of self and non-self**. Let me shed some more light into this very important concept. To make the reader gain a quick understanding of this concept, I would like to present a quick explanation of it:

The patterns of these protein structures are the base of the recognition process. **Self** protein structures are structures presented by the cells that do belong to the organism and should be there.
Non-self are structures presented by the cells that do not belong to the organism and should not be present. Now, the immune system is aware of what self structures are and only looks for non-self. When a non-self protein structure is found, that is classified as a pathogenic substance presenting an antigenic protein structure and is eliminated. This process is called **negative selection**.

Negative selection is a very important concept. This is simply the base concept of the entire working of the immune system through which the whole defense from foreign pathogenic substances is realized. There is one, very simple concept behind its workings. This is:

The system is aware of what should be there and only fights against those patterns that it does not recognize.

Let us present a quick overview of the negative selection process. The immune system is well aware of the the self protein structures. When lymphocytes are created, their receptors are created through a randomization process, if you will. They are created in such a way so they can be complementary to most protein structures, may they be self or non-self. After the creation of these cell, they are negatively selected in such a way that those with receptors that are complementary to self protein structures are killed. This is the **negative selection**. After this, only lymphocytes that do not bind to self structures are released into the bloodstream. Let us, at this point go onto the next section of this chapter to look at the intricate workings of the cells of the immune system and how they realize the phenomenons of negative selection and self/non-self discrimination and how all of this makes the adaptation and learning process of the immune system possible.

1.5 Adaptation and Learning

The next step into this research is to look at the whole precess by which the agent cells of the immune system learn to recognize the non-self or antigenic protein structures and how they do not bond to the self protein structures. As mentioned earlier, the lymphocytes are part of the white blood cells and they have receptors on their surface which bind to the antigenic protein structures of cells of pathogenic substances. There are two main kinds of lymphocytes present within the bloodstream. The **B lymphocytes** and the **T lymphocytes.** They are also known as **B-cells** and **T-cells**.

1.5.1 B Lymphocytes

Extracted from [10][11][12].
B lymphocytes or B-cells are those agents of the immune system which make the detection process possible. These cells are able to adapt to a specific type of antigenic structure and enable the efficient detection of the pathogenic cells that present such antigenic structures.

As all lymphocytes, B-cell, present receptors on their surfaces too. When these receptors bind to a protein structure, the structure is classified as antigenic and the event is defined as a match event. This match leads to the activation. Once a B-cell is activated there is a series of events that take place. This series of events develop the adaptive immune response that was mentioned earlier in the chapter. Let us take a look at these events in more detail.

Upon activation of a B-cell, a cell division process is started. This is the procedure by which cells in the human body multiply. This can be seen very well as a cloning process.
During the cloning of any B-cell, the receptors of the clones undergo a process that is known as **hypermutation**. This process makes the receptors of the cloned cells mutate at an unusually high rate. The reason for this is to ensure that the clones have much higher level of affinity with the antigenic structures encountered.

Now, during the cloning process, the affinity of the clones to the presented antigenic structure might decrease or increase. This is where the process of negative selection comes into play. The clones with higher affinity are the ones that survive this selection process. The ones with lower affinity are eliminated through a programmed death. This is called **apoptosis**. This death happens after a very short time and the reason for this is that a lower affinity towards a non-self structure might mean a higher affinity to a self protein structure, case in which the immune system would start killing cells that belong to the organism. This phenomenon is known as an **autoimmune response**. In medicine this is classified as a disease, however in the modern days it is very very rare [1][12].

The B-cell clones that survive the process of negative selection are then separated into two different types of B-cells themselves. This step is very important towards the adaptation phase of the immune system. The cells created in this phase are:

- **Memory cells** – these cells have a long life span and present receptors with much higher affinity

- **Plasma cells** – these cells have a shorter life span and very few receptors are presented on their surface. The purpose of these cells is the production of certain agents known as **antibodies**. These are a type of recognition molecule that is released into the bloodstream by the plasma cells.

The cycle described above is known as the adaptation cycle of the immune system. A simple view of the steps of this cycle would be as follows.

1. Activation of B Lymphocyte.
2. Cloning process and Hypermutation of receptors.
3. Negative Selection of Cells.
4. Separation of Memory Cells and Plasma Cells.
5. Process repeats as a loop until no more bonding is created between lymphocytes and other structures.

With each adaptation cycle the B-cells become more and more specific towards a type of pathogenic substance. The whole process is repeated over and over until the infection is cleared. This means until all cells presenting antigenic protein structures have been cleared. This process constitutes the primary immune response. Meaning that it only happens upon introduction of a previously not encountered pathogenic substance. In future occurrences of the same pathogenic structure the immune system will be ready and the secondary immune response will lead to a much quicker elimination of the threat.

So, as we can see, a relatively generalized response has evolved into becoming highly specific. As it can be seen the inner workings of this system are very dynamic and distributed.

1.5.2 Autoimmune Responses

Extracted from [1].
The reader will recall from previous sections the phenomenon of **autoimmune responses**. This occurs when the receptors of the B-cells start binding to protein structures defined as **self**. This is what gives the immune

system the power to entirely annihilate the organism it belongs to.

It is worth at this point having a closer look at how this phenomenon is avoided by our immune system.
The process by which the immune system avoids the initiation of auto immune responses is very interesting one, and it is a very important process too.

B-cells are created and matured in the bone marrow. This is the location where they undergo a rigorous selection process, by which the ones that present receptors that are complementary to self protein structures are programmed to die. However this is not sufficient to make these cells entirely tolerant to self protein structures. Also, the cloning process which they undergo upon activation starts a hypermutation of their receptors, which makes them even less tolerant to the self structures and more prone towards initiating autoimmune responses.

This is where another type of cell called the T-cell comes into play. A B-cell, to activate itself, must receive two different signals. One is the binding of its receptors to the antigenic protein structure. The other signal must come from a set of T-cells called the T Helper cells, also known as TH cells. If the second signal is not received the B-cell will die. This process prevents the autoimmune response. Let us take a look at these T-cells.

1.5.3 T Lymphocytes

Extracted from [10][11][12][4].
T lymphocytes, or T-cells, are also produced in the bone marrow. Their maturity occurs by migrating to the thymus. Here they are made tolerant to almost all self protein structures. The TH-cells ensure tolerance to all self protein structures. In the thymus these cells undergo the previously explained negative selection process. This process then only allows cells that are tolerant to self protein structures to live.
The difference between B-cells and T-cells is as follows. T-cells present a unique receptor binding to antigenic structures. B-cells instead are antibodies bound to membranes that can only bind to antigenic structures on pathogenic substances. T-cells instead, present receptors that recognize antigenic structures in combination with a certain set of molecules called **Major Histocompatibility Complex**, (**MHC**).
These molecules can be subdivided into two different types. The first type can be found in cells that present antigenic protein structures and shows to the cells of the immune system what they have detected. This type of cell I recognized by the TH-cells. The second type of MHC molecule is found in every cell of the body and

displays the genetic code. This code is what is recognized for the successful performance of the self/non-self discrimination. The immune system is aware of what code to expect. If the code is not what is expected, this is classified as an infection.
B-cells use the MHC type on cells that present antigenic structures for the recognition process and then refers to the TH-cells to get the secondary approval for this recognition to be interpreted as a detection of a pathogenic substance. This approval is known as the **costimulatory process**. If the approval is given the B-cell is activated and the adaptation process is started. Otherwise the cell dies through apoptosis.

1.6 A Summary

This chapter provided a good description of the immune system and the mechanisms used by it for the successful detection and neutralization of a threat. I would like, at this point, to present the reader with a brief summary of all that was discussed in the previous sections.

The immune system has a large amount of cells at its disposal. These cells act as agents and perform the tasks of detecting and neutralizing pathogenic substances. Each cell is entrusted with a certain task and the cooperation of all these cells ensures the correct working of this system.

The chapter has focused its attention towards the adaptive part of the human immune system. This system has a very good ability for performing the detection of novel threats. This process is performed through the ability of the system towards the discrimination of non-self form self. The system is aware of what belongs in the organism and anything that does not meet the previously specified requirements is classified as an anomaly and eliminated.

The base cell for the performance of such task is the lymphocyte. Lymphocytes are part of the white blood cells. They present a number of receptors on their surface which bond to antigenic protein structures of pathogenic substances. The receptors on one pathogen are similar between each other but they differ between different lymphocytes. This is what makes the high specificity of them.

The lymphocytes are subdivided into two groups. B-cells and T-cells. These are the cells that form the immune response upon introduction of a pathogenic substance. The response is built in such a way to avoid autoimmune responses. This fact clearly states the ability of the immune system to discriminate between self and non-self. T-cells ensure tolerance to the self elements, whereas the B-cells undergo cloning,

hypermutation and negative selection to increase their diversity specificity to pathogenic substances.
B-cells individuate antigenic structures and T-cells make sure that they are in fact non-self. B-cells can not act without approval from T-cells. This dual authentication includes the cloned B-cells.

Chapter 2

Issues, Principles & Guidelines

The previous chapter dealt with taking a deep and detailed look into the best intrusion detection and prevention system known to science. The human immune system. A number of mechanisms and principles were individuated. All these mechanism ensure our well being and protect us from most threats that could, if left undetected an not neutralized, annihilate our entire species for that matter.

The following chapter will deal with trying to adapt these concepts into the world of information security. Trying to use the mechanisms used by our immune system in an adapted way to ensure the three famous goals of information security, **Confidentiality, Integrity, Availability**, is not a trivial task, however, it is one very worth looking into. After all most of the mechanisms of the immune system can be viewed as possible analogies in the world of information security. Some of them are very good authentication methods, or pattern recognition problems, whereas other are problems of distinguishing self form non-self, just the way

information security methods attempt the discrimination between legitimate and non legitimate users [13], actions or access requests. Both problem which are dealt with to a great extent in computer science as a whole and more particularly in information security.

Even though the transition process for these principles is not easy a good analogy between the two can be made. Both systems deal with the same problem: Defending their host from threats. The analogy is indeed quite simple. The immune system protects the organism from foreign threatening intrusions and a computer security system also is there to protect the system from intrusions.

The next sections will deal with looking at the concepts, issues and design principles of both security systems and principles to be used in the design for a computer immune system.

2.1 Information Security Issues

There are three aspects which are faced by computer security and information security in general for that matter. **Confidentiality, Integrity, Availability**. We are all well aware of these words and are also very familiar with them, as they are presented on almost every first chapter of any information security book or paper. These are the three main issues that need to be addressed by any security system in the world of information security. [18]

- **Confidentiality –** Access to data should only be granted to authorized users and no one else. Any given user should only be able to access the data that he is entitled to and should not be able to get access to say another users personal files.

- **Integrity –** Data needs to be protected from unauthorized change. Any piece of data, weather during transit on a network, or simply during the period it is stored and not being used by its legitimate users, should not, in any way, be allowed to change. Data should only be altered by legitimate users..

- **Availability –** Data and resources should be available upon request by their legitimate users. This has to do with the prevention of denial of service attacks.

The above three are the main and most popular issues that information security deals with. However there are two other aspects, which sometimes are not mentioned. These are secondary characteristics that are in support of the main three described above.

- **Accountability** – In the eventuality that an intrusion or anomaly in a system has been detected, the system should be able to preserve sufficient information in the aid to backtrack to the origin of the intruders.

- **Correctness** – The classification of events should be as correct as possible. False alarms should be as minimal as possible. If a legitimate user's actions are too often wrongly classified, this will prevent a legitimate user from performing the tasks that he or she is actually entitled to.

The issues stated above play a great role in information security. It can be said that if the mechanisms of a security system address all of these issues, then that system is secure. However with the large complexity of nowadays systems, there is a great number of number of possibilities of inputs that need to be considered and covering all of these issues becomes a very time consuming and difficult task both in the design process and

implementation of any security system.
It is agreed that implementing and maintaining a secure system is difficult and really, there is no practical way of ensuring that the required level of security stated by the specification has been met [18].

Security vulnerabilities result as oversights and holes in the design of software. If these are exploited, the result leads to a breach of the security policy. However, there is also the concept of an accepted risk. The reader will be introduced to such concept in the following chapter through the acceptance of a failure probability.

Considering the size of software that is implemented at present it has become practically impossible to build a completely secure piece of software. Therefore it is considered more acceptable to design software and accept the possible flaws in it and have a separate system that deals with addressing the flaws in security. The task of these systems is to individuate matters such as misuse of the system or anomalous behavior, detect and possibly prevent the exploitation of flaws [18].

The next section presents the eight, very important, principles to be kept in mind during the design and implementation of a computer security system.

2.2 Design Principles of Computer Security

In an introduction to the protection of information in computer systems written by Saltzer and Shroeder in 1975 [19], eight design principles for a security system were presented. Even though this dates quite back, some, if not all, of them are as relevant today as they were then. These are presented below [19]:

- **Economy of mechanisms** – Keep the design of a protection mechanism as simple as possible. This is a design principle that should be applied in the design and implementation of any piece of software, however it it particularly important to the design of security systems. Over complicating the design and the implementation of any piece of code might lead to a larger number of vulnerabilities that can not be noticed during normal use but are exploitable by attackers.

- **Fail-Safe Defaults** – Access should be denied unless explicitly authorized. Therefore, in simple words, the design should not not be done in such a way to allow allow access to any data or resource unless explicitly denied. This is very important in the case of an oversight of an authorization policy

for any given object of the system.

- **Complete Mediation** – Every request for access to protected resources should be processed by the protection mechanisms and checked that it does not break the security policy. This is the most important principle of design for a security system.

- **Open Design** – Security by obscurity is bad security. In simple words, the strength of the security system should not be any weaker in the case that an intruder knows it's design. The security mechanisms should still be able to perform even if their design is well known.

- **Least Privilege** – Any program should only be given strictly the resources it needs in order to perform its functions and tasks. This will make sure that in case of mis performance by any program or vulnerability exploitation of any given program will result to as minimal damage as possible.

- **Least Common Mechanisms** – This principle was suggested earlier by Popek [20]. The idea behind it is that the use of shared resources should be minimized. The sharing of resources creates scenarios where information is passed between the different resources and this could lead to a compromise of security. Adopting this principle in practice however can be hard as suggested by [18].

- **Separation of Privilege** – If possible a method of dual authentication should be used for the granting of a request. An example would be the case of smart cards, where the user must hold the token itself and the PIN code for it.

- **Ease of Use** – This principle is of vital importance and is concerned with the human element. It is a very hard principle to be dealt with in practice as it involves awareness of the user. For example a vulnerability that is a breach of this principle is a poor choice of password by the user. And as any person who deals with information security will know that a large number of end users will choose easily guessable passwords. And not end users only for that matter.

2.3 Principles for Computer Immune Systems

Computer immune systems is one of the many names by which, the systems designed for the detection if intrusions and inspired by the human immune system, are known. This section will deal with the principles

deduced from the design of the human immune system. These principles can be very beneficial to computer security systems when analogized. As shown in the research done in the previous chapter the human immune system has an excellent intrusion detection technique for the individuation of a new, previously not encountered threat and a very good ability of improving itself towards the prevention of threats that have been encountered in the past.

There are of course some large differences between a human body and a computer. As stated by [17] we desire a digital system built out of signals and not cells and molecules. Furthermore, it is important to try avoiding the recreation of all the elaborate genetic controls, cell signaling and other aspects of the immune systems which are dictate by the physical constraints that such system has evolved under. The last and very important issue to be taken into consideration is that the human immune system is a system that has as a main objective the survival of its host. A computer security system has as an objective the prevention of any type of threat. However the set of principles below should be used as a guideline in the design of a computer security system inspired by immunological techniques.

The design principles presented below are as stated by a the paper of Somayaji in 1998. [17]

- **Distributadability** – The immune system consists of a large number of agents and components that are distributed throughout the body. These components interact locally to provide detection and protection mechanisms in a distributed manner and there is almost no centralized control or coordination. This means that there is **no single point of failure** in the system.

- **Multi-Layered** – In the immune system, no single mechanism of defense provides provides complete security. Multiple layers of different mechanisms are combined and complement each other to provide a high overall security. This is not a new concept in computer security and should be emphasized in system design.

- **Diversity** – One form of diversity can be viewed as the difference in the immune system of different individuals of the population. This improves the robustness of the population as a whole. The vulnerability of different individuals to the same disease will be at different extents. In an analogy, vulnerabilities found in one system would not be present in another. Another level of diversity is the difference between components of the immune system within the same individual. This provides protection against a large number of threats.

- **Disposability** – Individual components of the immune system are multitudinous and redundant. In simple words the loss of some of the components will have little or no effect in the correct functioning of the immune system. The fact that these components are disposable together with the lack of centralized control makes the immune system tolerant to failure of some of the components.

- **Autonomy** – The immune system does not require outside management or any type of maintenance. It works autonomously and successfully classifies pathogenic substances and repairs itself by replacing cells. However it is not yet expected that computer security systems have such a degree of independence. However, with the increase in CPU speeds and increase in the complexity of software it will be increasingly important for computers to menage security issues automatically.

- **Adaptability** – The immune system, as stated extensively in the previous chapter, is able to adapt towards the faster recognition and elimination of pathogenic substances. This speeds up the primary response and makes secondary responses exponentially more efficient. A computer security system should also have such properties. Learning to recognize new intrusions and be able to remember signatures of old ones.

- **No Secure Layer** – Any cell in the human body can be attacked by a pathogenic substance. Lymphocytes are also cells and they can be attacked too. However these will be attacked by healthy lymphocytes. In this concept of mutual protection we can analogize a secure code base.

- **Dynamically Changing Coverage** – The immune system does and can not maintain a large enough set of lymphocytes to cover the space of all pathogenic substances. I makes a space/time trade off in its detector set. At any time it maintains a set of randomly selected detectors. This sec is constantly changing.

- **Identity via Behavior** – In cryptography, identity is proven through the use of a secret. The human immune system does not depend on secrets however. Identity is verified through the presentation of protein structures. These can be though of as the "**running code**" of the body. A model for this is discussed in [7] and will be looked at in the next chapter.

- **Anomaly Detection** – The immune system has the ability to detect pathogenic substances that it has not encountered before. It is able to perform anomaly detection. The property of being able to recognize novel attacks is very important for any security system. This property is one of the key issues for this study and will be looked at into much greater detail in the next chapter.

- **Imperfect Detection** – By accepting imperfect detection, the immune system increases the flexibility with which it can allocate resources. Resources are allocated according to the need and the extent of the threat posed by the detected intrusion.

- **The Numbers Game** – The human immune system replicates detectors to deal with the replication of the cells of a pathogenic substance. It must do so as otherwise the pathogens would overwhelm

any defense. Computers are subject to a similar numbers game. Hackers freely trade newly found vulnerabilities and exploits. The success of one hacker can quickly lead to thousands of hosts being compromised. The analogy is that pathogens of the computer security world are playing the number game. However traditional systems are not. This is as proposed by [17], however it can be argued that the increasing of the number of identical defenses would not help in combating attackers. However, if the number of dynamically changing defenses is increased, then this could create a more favorable security scenario.

These principles as posed to us from [17] are very good set of guidelines. It is not possible to fully translate these principles and analogize them in order to create methods that can be used to address the issues of information security, as the difference in architecture of the two hosts has to be taken into account, together with the different architecture between the types of threats posed to the two different hosts. However it is useful to try and create analogies up the the extent which they are possible and prove to be useful.

2.4 Detection & Prevention

I would like to differ between these two different concepts at this point just to shed some more clarity in the matter of translating the design principles stated above.

Detection, as defined by [22], is the process of discovering the existence or presence of something. In immunology the detection process is simply the recognition of a the presence of a pathogenic structure. Similarly in information security, the process of detection is simply making the user or the system aware of the presence of a threat. This is not preventing the threat from doing any damage. In the next step, the immune system does neutralize the threat, however this does not mean that the threat has not in fact done any damage.

Prevention, as defined by [22], is the process through which a certain event is stopped from happening. From an immunological point of view, prevention is the process of neutralizing a pathogenic substance before this can pose a threat to the maintenance of homeostasis in the organism. As mentioned in the previous chapter, homeostasis is not entirely stable and it has certain levels of threshold between which it is considered normal. Therefore, is is very unlikely that a pathogenic substance will do no damage at all. The important thing is that it does not do enough to move the homeostasis level beyond its thresholds.

This can be analogized to some extent in information security too. If a vulnerability is exploited and the system is able to detect the anomalous behavior it can stop the process from completing itself. However this is a very dangerous approach.

In a computer security system, the optimal defense mechanism is the one of prevention. In immunology in contrast, detection is also acceptable after some damage has been done, as the body has the power to regenerate itself.

However, there is a point to be noted here. In information security detection and prevention are two concepts that go together. An intrusion is detected and possibly prevented from carrying any undesired activity. Given that the detection is done at an early enough stage, the system would be able to prevent that intrusion from executing unwanted actions. The detection could also be done at the point where some desired action is performed, however, this is not necessarily bad. At least the detection would prepare a ground base for the full prevention in the future.

2.5 The Security Policy of the Immune System

In the previous sections, namely section 2 and 3 of this chapter, the study has dealt with researching into concepts and general guidelines that have been developed by past and current research for the design and implementation of security systems.

This study, as previously stated, is dealing with the issue of intrusion detection. Mainly with the detection of novel types of intrusions for which specific signatures do not exist. Before looking at the different types of architecture for a system that addresses the above issue I would like to once again, take a look at the principles individuated in the third section of this chapter. These principles are also mentioned in the dissertation of Steven Andrew Hoffmeyr [10]. This paper presents an immunological model for distributed detection. Some of the principles stated in this paper have been adapted in a different manner and I think they bring out some very key issues. I would like to concentrate over one in particular:

- **Implicit policy specification – "***The definition of self used by the immune system is empirically defined. Self is implicitly defined by the "monitoring" of proteins that are currently in the body. The*

advantage of this approach is that self is equivalent to the actual normal behavior, and not a specification of what normal behavior should be. In other words The immunity system does not suffer from the problem inherent in trying to correctly specify a security policy." [10]

As stated in the previous chapter, self, is defined as the set of protein structures presented by cells that do belong in the organism. The immune system knows what self is right from the beginning. Now, as stated above, self is not defined through a security policy, but it is constructed by monitoring structures that do belong there already.

For the purpose of this study there are two key issues that need to be looked at into more detail and from an information security point of view. These issues are the very famous self/non-self discrimination and the negative selection.
Creating a security system is not a very difficult issue, however, creating a both efficient and effective security system, is not an easy issue at all. The above mentioned issues are very well addressed in the immune system. In fact they are mechanisms which make this system achieve the level of perfection that it does. If we can find a way to translate them in terms of information security, this would be a great step towards the realization of a good security system.

There are a different number of possible architectures and approaches towards the realization of the above goal. The next chapter provides an insight into the current research advancements for the application of negative selection and self/non-self discrimination in computer systems.

2.6 Self & Non-Self In a Computer

As stated in the first chapter, the human immune system makes use of two main techniques for providing the defense against pathogenic substances. The discrimination of what is self and non-self through using a number of agents that through the process on negative selection are made tolerant to self, in the sense that they will not detects self structures as and anomaly, and they will be able to detect what is non self.

A number of techniques have been devised to imitate this approach in the world of information security. The next chapter will go into detail about the description of these techniques and their methods of work.

As stated in the previous section, the immune system makes use of a given knowledge about what is self and what is not. For any of the given architectures above, the main difficulty would be the fact that we do not have a knowledge of what self is. Of course it can be argued that self will be what is defined as what is permitted by the security policy. However, as it is stated in [18] the security policy does not cover all of non-self in practice. Therefore a better method has to be created for the definition of self through which non-sels can be deduced [7]

In the principles mentioned above form [17] one of them is **identity via behavior**. This poses an issue. How is it possible then to gather a set of data that would offer the knowledge needed in order to determine what self is.

As an overview on the next chapter, this will go into looking a few negative selection techniques. Then some of the proposals for gaining a sense of self will be investigated. Also, a method devised for the self/non-self discrimination will be looked into.

2.7 Misuse or Anomaly Detection

This study is investigating into methods for the detection of intrusions through the use of methods modeled from the principles of the human immune system. It is of a great importance to classify what is meant by intrusion before continuing into the research of the methods.

Intrusions in computer systems are the cause of undesired effects. As deducted from [8], these intrusions can be classified into intrusions through misuse and intrusions in the form of an anomaly.

According to [8], misuse is referred to known types of intrusions that are tackled through the use of very specific methods such as signatures. These signatures detect the intrusions when they happen and do not let them carry any effect that they are supposed to do as intended by the payload of the intrusion.

Anomalous intrusions refer to the types of intrusion for which a specific signature for it's the prevention of it's effect does not exist. These are the novel intrusions that have not been encountered in the past in order for them to respectively exist a signature that can be implemented in security systems for it's detection and prevention of the damage. So, for an anomaly, the signature is not known, but the execution of an anomalous intrusion would result in undesired effects, behavior other than the one initially intended [8].

This study deals with solutions to the latter type of intrusions.

The methods explored in the following part of this study are methods that try to determine anomalies through the knowledge of what is normal. This is a very good imitation and analogy to the defense mechanism used in the human body.

It has to be noted that the detection of intrusion through signatures is quite good, given the rate of success. However, the performance of it is only noted when there are intrusions of a known kind and a specific signature exists for them. In the opposite case (where a signature is not known), the system becomes vulnerable. Therefore, a dynamic method of detection is needed. This problem is the one will be looked into in the remainder of the study.

2.8 Issues to be Addressed

As looked into previously, the immune system uses the processes of self non/non-self discrimination for achieving the goal of defending the mechanism from pathogenic substances. It recognizes what is non-self through having a clear knowledge of what is self. Agents that recognize non-self are selected through the negative selection method. These agents recognize non self through a pattern recognition process of strictures in non-self considering a level of affinity. This can be interpreted a a matching process given a matching rule.

The research presented in this chapter will deal with going into methods and techniques crated achieve a similar type of defense in computer systems.

As previously stated, the immune system uses a definition of self that is empirically defined. We do not have such an option when dealing with computer systems. Self is not defined in such a manner from current methods of security policies that it can be used to generate agents that would recognize non self.

The firs task of this chapter will be to investigate into the possibilities of generating a sense of self. Once this is created, the next steps can be taken. Imitating the methods of defense present in the immune system would include that self is given in some manner. Once that is given, then the generation of non-self detecting detecting techniques can be carried.
One of the issues to be addressed in the creation of a solution for such a security system, would be to decide

what to analyze. The immune system analyzes structures of proteins in cells. For our system, what would these structures be? The analysis in this chapter will present ways of generating the self structures, in order to determine what the non self structures would be.

The non-self detecting agents are created and selected through the negative selection process. A number of algorithms that analogize this task will be presented later in the chapter. However, these agents only bind to the non-self structures if a level of affinity, representing a threshold value, is met. This would be, in the context of information security, a matching rule, that would answer the following question. How complementary must the structures of of the agents and monitored elements be in order for any given element to be classified as non-self? The parameters set in the matching rule in this case would determine the level of affinity.

Therefore, the issues to be tackled are:

- How to define self.
- Given the definition of self, how to generate non-self complementary structure.
- Define the affinity needed between the non-self complementary structures created and the monitored structures for classification process during monitoring.

The following parts of the chapter will present the reader with a series of methods and algorithms devised to address the above three issues points.

Chapter 3

Current Research & Literature

This chapter will present some research and insight into the current progress of information security in the area of intrusion detection methods inspired by the principles of the immune system.

There are number of different names under which these systems can be found such **Artificial Immune Systems, Computer Immune Systems, Hybrid Intrusion Detection Systems, Immune System Inspired Intrusion Detection Systems** etc.

In the following sections of this chapter I intend to present the reader with some information and detail about the current research, algorithms and methods translated form the principles of human immunology and used in the field of intrusion detection.

The challenge of these systems is the determination of what can be classified as normal activity and what could be potentially harmful activity. In traditional intrusion detection and prevention systems, the only way that we are able to address the issues of information security is through systems that base they detection on a specific set of rules and only detect specific events. They rely on a predefined set of rules..

The ultimate goal is to design a method that can protect the system from previously encountered attacks through signature based anomaly detection (which is already a present and much used method) and also be able to detect attacks of any type that have not been encountered previously.

The immune system of the human body does not rely on such rules. This system uses knowledge of what is allowed in order to prepare for handling what is not allowed. This does not imply the use of a static set of rules. The human immune system makes use of the empirical definition of self. The reader will be introduced to a method, currently used by researchers, for the establishment of the definition of self that is not based on preset static rules.

A set of architectures will be presented in this chapter and it will be extracted from current research projects and papers. Following this, a more detailed analysis will be made, presenting methods and algorithms that simulate the principles of negative selection and self non self discrimination.

3.1 Architecture Proposals

In the paper by Somayaji, Hoffmeyr and Forrest [17] four possible approaches towards an architecture for a computer immune system are presented. These are some ideas derived by a direct mapping between immune system components and current computer system architectures. Let us take a look at them as presented by the above mentioned paper.

3.1.1 Four Analogized Architectures

- **Protecting static data** – [17] This approach begins at the level of static malware. These malicious pieces of software infect programs and boot sectors by inserting instructions into them. Then upon running of an infected program, the malicious set of instructions will create an undesired and abnormal event. The consequences of this might be of any kind, between a hardly noticeable and total corruption of data and/or security of the system.
  In this approach self is defined as uncorrupted data. Programs that have not been injected with malicious instructions.
  Non-self, in contrast, is simply any change to self. A number of algorithms have been devised for the addressing of this problem. Some of these are directly inspired by biology and they are shown in in greater detail in [17].
  However, judging from what is stated in [7], this architecture is not of great success, as the static data is not going to have any effect until it is run.

- **Protecting active processes on a single host** – [17] The human immune system is primarily made up of cells. These cells monitor other cells and interact with them. This type of view tries to analogize this concept as follows:

  - every active process in a computer is a cell
  - a computer running multiple processes is a multicellular organism
  - a set of computers is a population of such organisms.

  The traditional security systems, such as passwords, permissions etc, can be viewed as playing the same role that the physical and physiological barriers and the innate immune system play. They are only effective towards previously encountered attacks.

  The implementation of an adaptive security system, analogized from the adaptive immune system, could include the implementation of a lymphocyte process, which with help from the kernel, is able

to query other processes, and see whether they are running normally. Analogizing from immunology, if a process is running abnormally (it is non-self), it is either damaged or under attack. The response of the lymphocyte process to such an event could be to slow, kill, suspend or restart the misbehaving process. To complete the analogy, each lymphocyte process, similarly to the cells of the lymphocytes of the immune system, could have a detector or set of detectors that are created randomly. The lymphocyte process should live for a short time and then be replaced by another process with different detectors, as it happens with lymphocytes of the immune system. This constant change will lead to no predefined location or control thread at which such process could be attacked. Lymphocyte processes that prove to be quite useful and indeed do prevent intrusions during their lifetime could be allowed to have a larger lifespan and/or allowed to "clone" by creating related processes with similar but more specific detector sets. The autoimmune responses can also be prevented by imitating the elimination of lymphocytes that bind to self structures as it happens in the thymus.

As mentioned before, in this type of architecture self is the normal behavior of precesses and non-self is the abnormal behavior of processes. This can be an intrusion in either privileged or user processes. The ability of lymphocyte processes to replicate themselves makes the system adaptable to user behavior and system software. However this can also be used as a vulnerability by malicious users trying to train the precesses to be tolerant no non-self.

The level of security is also tunable. This can be done through an adjustment of the lifetime of lymphocytes, and the adjustment of the number and quality of their detectors [5][6].

The implementation of such an architecture will need a system for making the processes tolerant to self structures. This is similar to the process undergone by the cells of the immune system in the thymus. There is a number of papers that deal with presenting a model for this, and such issue will be discussed in later section of this chapter.

- **Protecting a network of mutually trusting computers** – [17] This approach has a very interesting view. Think of each computer on a network as an organ within an organism. Each process is still considered as a cell. An individual however, in this model of architecture, is a set of mutually trusting computers. In this model the innate immunity is constitute by traditional host based security mechanism like passwords and access controls, combined with network security mechanisms such as firewalls and Kerberos. Kerberos is a security system which uses a trusted entity to grant access to a certain server to a non trusted entity. For more details, the reader is directed to [23][18].

The adaptive layer of the security system will be similar to the one in the former architecture. Kernel assisted lymphocyte processes, however in this architecture the these processes can migrate between computers and as an analogy to the lymphocyte cells in the human blood stream, make them mobile.

One of the computes in this case, or indeed a set of them could act as the thymus in this architecture. The duty of this set would be to create and propagate these lymphocyte processes each of which searches for a specific pattern of abnormal behavior. If negative selection is used by these lymphocyte processes, a centralized server to coordinate a response to a security breach is not needed. The detecting process can take whatever action is necessary in this eventuality. A possibility for it is to replicate itself and circulate to find similar patterns of behavior on other hosts. This technique is inspired by the kill signal approach described in a paper by Kephart [15].

"*The similarity between this architecture and the previous one can be clearly seen. The difference in this approach, is that in this case the lymphocyte processes are roaming in the network. According to the source from which this approach is extracted [17], the process should be able to detect the same set of anomalies, however, on the plus side, anomalies found on one machine can be quickly eliminated form others.*"

There is one extra requirement in comparison to the previous architecture and this is that this type od system would heavily depend upon a robust framework for mobile agents. It also has the analogy form the immune system of lymphocytes monitoring each other as well as other cells. In this architecture, the lymphocyte processes are in fact processes running on each machine, so, given the free roaming of these processes from machine to machine, these processes would also be able to monitor each other. This will ameliorate also the danger of malicious mobile lymphocyte processes that self replicate. Therefore, a malicious type of attack that menages to inject malicious instructions in one of the processes, would quickly be stopped from other lymphocyte processes of the network.

- **Protecting a network of mutually trusting disposable computers** – [17] This proposal for an architecture is moving the analogy up by another level.

  This approach is also a network based one. Here each of the computers is regarded a cell. The computers on the network are mutually trusting between each other. Host-based security would, in this ca, analogize the normal defense a cell has against attack. The innate immune system would, as before consist of the traditional methods such as firewalls and Kerberos.

  The implementation of the adaptive immune system can be achieved through creating a set of machines that actually perform the task of lymphocytes. In contract to the previous architecture, where processes running on each machine, would monitor the state, in this proposal we have separate machines with the task of doing this.

  These, so called, lymphocyte machines, would in this case monitor the state of other machines on the network and when an anomaly is detected, they must respond. The response could be anything from isolating the problematic machine, which would need to be done by a dynamic reconfiguration of the network, or just simply shut down or reboot the machine. If the source of the anomaly is outside the network, the lymphocyte machine could stand in for the victimized machine and battle

the attack.

There are a set of problems that could be addressed in this architecture, such as denial of service attacks, network flooding and hardware failures These are not as well addressed by the other architectures as stated in [17]. The downside is that the requirements for the implementation of such an architecture are quite significantly large and difficult to achieve. The implementation would require an analogy of the epitope/protein structure at host level. This could be based and mirroring patterns of the machine's network traffic, or provide the behavioral patterns of the kernel of that machine. A further difficulty arises with the requirement of a dynamically configurable network. This is indispensable however for the lymphocyte machines in order for them to be able to isolate other machines in case an anomaly is found. Also, as in the previous architecture the need for a thymus analogy arises.
There is also another issue, which I think is a key one and makes this architecture almost impossible to implement. This assumes, that hosts are interchangeable as it analogizes them with cells of the body. In the human body cells can be quickly replaced, however in a network it is not so feasible to just loose machines as they are hardly ever interchangeable and redundant.

3.1.2 Limitations

I would like to present the reader with some view on the limitation that are to be kept in mind in the design of the above presented architectures. These limitation are as by [17].
It is believed that the imitation of the ways nature deals with the detection of intrusions is a very inspiring approach that could be applicable information security. However, this may not be fully possible. The reason for such a problem is that immunological solutions are not directly applicable to the computer systems that are present in nowadays technology. One other risk that is encountered is that these solutions might make us ignore the non-immunological solutions which would in some cases be more appropriate for tackling the issues that we are faced this in information security. Another risk that is posed by this imitation is the inheritance of the many assumptions of the immune system. A major one of these is the assumptions made in the process of self/non-self discrimination.
It also has to be noted that in information security, as pointed out in the precious chapter we are faced with the five main issues of confidentiality, integrity, availability, correctness and accountability. Whereas the immune system has only one issue to tackle in its approach. **Survival of the organism**. According to [17] this can be thought of as a combination of integrity and availability.

Viewing the memory of the immune system as a type of audit trail, might be looked as an analogy of accountability, however the immune system, only holds enough information about a new attack so that it can be recognized in the future. It does not hold information about the source of the attack.
Correctness and confidentiality are totally irrelevant to survival. Correctness, from an information security point of view, means that a certain piece of software meets the specified requirements. Immune systems are not formally specified systems. This makes the definition of these systems as “CORRECT” not possible. We define something as correct when it meets the required specification. If there is no specification, correctness can not be ensured.
In the context of confidentiality, this is not an issue at all for the immunity. The immune system is not concerned what so ever with the protection of secrets or privacy. This is definitely the largest limitation to be kept in mind in these architectures. [17]

Passwords, access control and good design are still needed for dealing with the issues of information security. As mentioned in the previously presented architectures, these measures can be analogized to the innate immunity present in our organisms.
The focus of this study towards adaptive measures of security does not imply that the traditional methods should be replaced. This type of security should instead be added as a new layer to make sure that the traditional methods are not being breached.

3.2 The Methods and Algorithms

Analogizing form the problems faced by the immune system of distinguishing self from non-self, as described through self/non-self discrimination in the first chapter of this study, most of the problems faced by information security can also be seen as a discrimination issue. Self are elements such as authorized users and actions, uncorrupted genuine data, original source code etc. Whereas, non-self would be elements such as intruders, viruses, worms and all other malicious attacks. [13]

As described in the first chapter, the methods of self/non-self discrimination can be divided into specific and non-specific, somewhat like the methods used in information security. We have specific methods such as signatures for virus checking and security analysis tools that use methods specific to one type of threat or vulnerability. Also there are non-specific methods such as good code hygiene, encryption, firewalls, etc, that are methods which tackle a wide variety of threats. However, the high specificity is not of high importance to this study. The concept of most relevance is the dynamical property of the immune system. The ability to create its own security system and adapt it to combat unknown attacks.

As stated in [24] in the security of present day we largely rely on methods of security for the prevention of security breaches. The problem with these methods is that they are quite passive. They play a role of prevention towards the intrusion of non-self elements into systems. These methods are unable to detect intrusions that are in progress in a system. These methods however are not inefficient. If we are to measure the efficiency of these methods by the rate of their success towards meeting the purpose they are designed for, then it can be said that they are very successful [24]. They are very specific to one specific type of intrusion and they provide a preventative measure towards that specific intrusion. These specific methods however, such as the ones through signatures for example, rely on non static recognition for the detection and prevention of intrusion. If an attack or attacker would be to change the intrusion technique, then the method, being static, would not have a very large success in the detection and prevention of this intrusion. These methods, or indeed the creators and menages of these methods and systems, have a very hard time keeping up with the constantly evolving and dynamic nature of attacks from malicious agents.

The problem with these systems is also the fact that they strictly rely on static rules for the detection of an intrusion and one rule is strictly a patch for one single possible breach of security. Due to this it is very hard to devise rules and patches for every hole in a computer system, especially keeping in mind the size of these in the present day.

As previously stated, the immune system uses a definition of self that is empirically defined. We do not have such an option when dealing with computer systems. Self is not defined in such a manner from current methods of security policies that it can be used to generate agents that would recognize non self.

The first task of this chapter will be to investigate into the possibilities of generating a sense of self. Once this is created, the next steps can be taken. Imitating the methods of defense present in the immune system would include that self is given in some manner. Once that is given, then the generation of non-self detecting detecting techniques can be carried.

One of the issues to be addressed in the creation of a solution for such a security system, would be to decide what to analyze. The immune system analyzes structures of proteins in cells. For our system, what would these structures be? The analysis in this chapter will present ways of generating the self structures, in order to determine what the non self structures would be.

The non-self detecting agents are created and selected through the negative selection process. A number of algorithms that analogize this task will be presented later in the chapter. However, these agents only bind to the non-self structures if a level of affinity, representing a threshold value, is met. This would be, in the context of information security, a matching rule, that would answer the following question. How complementary must the structures of the agents and monitored elements be in order for any given element to be classified as non-self? The parameters set in the matching rule in this case would determine the level of affinity.

Therefore, the issues to be tackled are:

- How to define self.
- Given the definition of self, how to generate non-self complementary structure.

The following parts of the chapter will present the reader with a series of methods and algorithms devised to address the above three issues points.

N.B. I will produce the analysis as shown in the papers in question for each one and where needed a brief summary will be made to illustrate a quick view for each one of them. For more details, the reader is asked to refer to the appropriate paper as mentioned in the references.
It has to be noted that the methods and algorithms presented below are the ground work for the development of the system. These are applicable to any of the architectures mentioned previously.

3.2.1 Defining Self

In the development of a security system for computers that is based on the methods that the immune system uses, it is important to be able to get a good sense of the self element in order to proceed into the classification of non-self.

What we are trying to do, is the discrimination of what is a normal behavior in a computer system and what is not normal should not be permitted. The question is what can be monitored in order to gain such knowledge.

In [7] we are presented with a method for defining a sense of self through the monitoring of behavior in system calls. The monitoring of program code in disk storage is of little use. This is not going to change until it is run. Intrusions are caused by running processes that execute system call

As studied in Computer Security [18] and stated in the [7], system calls are a mean of communication in the form of a low level function so high level processes such as executable commands can communicate with the lower level kernel of an operating system. They perform system actions through the kernel for example on behalf of the user. These are operations such as I/O calls for a file like read(), write() etc. Each of them has a slightly different atomic task, however their combination comes to a much different function. Depending on the complexity, applications initiate between tens and thousands of these per execution. From a security prospective according to [21], they are ideal for providing a detailed view for providing information on a process or system. It can be argued that these are too low level, however the paper in question, clearly states that they provide a good base for the "empirical" definition of self that we need.

Each process has a set of implicitly specified set of sequences of system system calls that can be produced, determined by the set of possible execution paths of the program. Normal execution will produce a certain number of subsets of this set. For nowadays applications, considering their size and complexity these sets will be very large. Every execution, furthermore will generate an entirely different set. However it has been noticed as stated in [7] that short ranges of these calls are consistent. This can be used in a definition of normal behavior, or self.

The process, is to initially scan the traces of the normal behavior in order to build a database of patterns for normal behavior. Entries of this database can be then used to monitor through the sets created by processes running and look for patterns that do not appear in the database of normal behavior. According to the affinity determined by the chosen matching rule, these will then be discriminated as self or non-self.

3.2.1.1 Building the Self Database

Being given a set of sequence calls for any given process, the method, as explained in [7], uses a window of

size k + 1 and slides it across the sequence of system calls and records the calls that fall within the window. The following example should provide an illustration of this.

We have k = 3 and a sequence of system calls that is a definition of normal behavior.

open, read, mmap, mmap, open, getrlimit, mmap, close

Sliding the window across the sequence, the first call is recorded and the calls that follow it in the next positions up to the size of the window. The first generated window would be as follows

Call	Position 1	Position 2	Position 3
open	read	mmap	mmap
read	mmap	mmap	
mmap	mmap		

Whenever a call occurs more than once, there is a number of possibilities as to what it can be followed by. These possibilities are recorded in the database and the repetitions of entries in respect to the call column and the position column are eliminated. From the above sequence, the database would look as follows.

Call	Position 1	Posiiton 2	Position 3
open	read	mmap	mmap
	getrlimit		close
read	mmap	mmap	open
mmap	mmap	open	getrlimit
	open	getrlimit	mmap
	close		
getrlimit	mmap	close	
close			

When the database of normal patterns is completed, the same windowing method is used, however this time for comparison against the database of self. This will determine whether the sequence of system call differs from the one defined in the self database.

3.2.1.2 Analyzing with Respect to Knowledge of Self

Suppose the following sequence is is being monitored and compared against the database in the previous example

open, read, mmap, open, open, getrlimit, mmap, close

According to the self database there are 4 cases of behavior that can be classified as abnormal. according to the self database, we have 4 cases that are not present in this sequence. The abnormalities that are not present in the self database are:

- open is not followed by open at position 3
- read is not followed by open at position 2
- open is not followed by open at position 1
- open is not followed by getrlimit at position 2

The decision of is made according to the affinity settings that we have decided to implemented and set in the matching rule.

However, in this prospect, we are just defining self and using it directly for the analysis. The immune system does not make use of self in order to decide upon the presence of non-self. The procedure it follows is that by using the knowledge of self, it creates detectors that are complementary to non-self in order to combat them. And then, once a match has been established between these detector and the non-self structure, according to the affinity level, that is predefined as a point of threshold, it makes the classification decision for the detected factor in order to decide towards it's elimination or not.

What the previous analysis lacks is a method for the generation of structures that are complementary to an unacceptable sequence of system calls. Given the self set, it is possible to generate a set of structures that are not self, but are complementary to non self. And, given a way to determine the affinity level needed, the threshold, threes structures, acting as detectors, would be able to detect non self sequences of system calls and act accordingly to what is specified that should be done in this case. This could be restarting the process, killing the process or any series of actions that is desired in the implementation of the system.

However, the previous analysis has defined a way that allows the gaining of knowledge of self that is not

predefined by a set of rules, like it is in the traditional methods of security such as policies. The immune system holds, in some way, an empirical definition of self. This is due to many factors, which are not relevant to this study, however, as a quick note, it has to be remembered that a human body and all the organs and the elements in it, evolve from a single cell, which holds the DNA, that is transferred to all other cells. Self is defined in some manner in the DNA. The system is aware of is right form the start.
It is not possible, due to the differences in construct and working, to have such a definition of self in information security. The previously introduce process however has helped us gain a sense of self through defining normal behavior.

3.2.1.3 Loss of Information

When the database for the definition of self is generated, there is a problem that is being encountered. We are essentially splitting the whole sequence of system calls into a set of subsequences. This will produce a loss of information in the context of what the entire initial sequence was. The method does not assume that there is a dependency between the generated substrings. The choice of the window must be picked carefully in order to minimize this loss of information.

The reader will be later introduced to an analysis for the calculation of loss of information that will aid in the decision for the parameters to be used in such a method. However, there is one problem to be noted here. The loss of information analysis through the splitting of the data as extracted from [6] and [5], assumes the use of a windowing method, where the window of size k is not moved by one position at a time. In this analysis the window is moved by k positions at a time. This would create a much smaller database and create a larger loss of information as the windows would be totally independent of each other.

I have not been able to find an analysis that assumes a k window that is moved by one position at a time, therefore, the assumption that the window will be moved by k positions will be made.

It can be argued that a large amount of storage is needed for this database. This entails a careful choice of parameters for the process and a technique for this will be introduced later.

3.3 Anomaly Detection Method

As examined previously in the first chapter, the immune system accomplishes the process of self/non-self discrimination through the monitoring of patterns in the strings of proteins presented in the epitopes of cellular structures. Analogizing from this, in information security self could be users logged in to the system, were detectors monitor random parts of their audit trail. This analysis is presented by [6][5].

Receptors of lymphocyte cells in the immune system bind to antigenic structures through patterns in shapes. This simulation is achieved in this model through a "matching rule" which acts on random strings over a given alphabet, by deciding weather the detector matches the string. There are a number of matching algorithms that can be used to get the result on which the decision will be based.

The choice of matching rule is very important, and to date, research has made use of the the "r contiguous-bits" matching rule.[6] Here, two strings are considered to match to each other in the case that at least **r** contiguous positions in the strings are identical. R is the parameter of the matching rule. For example, for r = 5 we will have

Case A:
0**1**0001**11000**101101
1**1**1010**11000**010010

Case B:
0**1010**100111001**010**
0**1010**001000110**010**

The two strings in case A, above, are said to match as there are substrings of 5 contiguous bits that are identical in both of them. This is not the the case in case B. Therefore, under the rule presented above there is no match in case B.

The choice of this rule in as stated, [6] was arbitrary. However, it was a decision made upon the fact that in [13] the use of this rule was successful in the simple detection of viruses. The rule is however used here for illustrative and explanatory purposes. In real life, other more efficient and more complex matching rules might be used.

Analogizing from the generation of B-cells in immune system, the generation of detector stings can be random. The strings that match any of the self strings, based upon the decision rule, should be eliminated as it is done in the negative selection of the immune system. The strings that remain will be the detector set to be used for the detection of non-self strings. This set of strings will form a repertoire that is named **R**. The process is repeated until a sufficient number of strings is generated to achieve the desired level of security.

The next figure (Figure 3.2) is a visual illustration of what should happen.

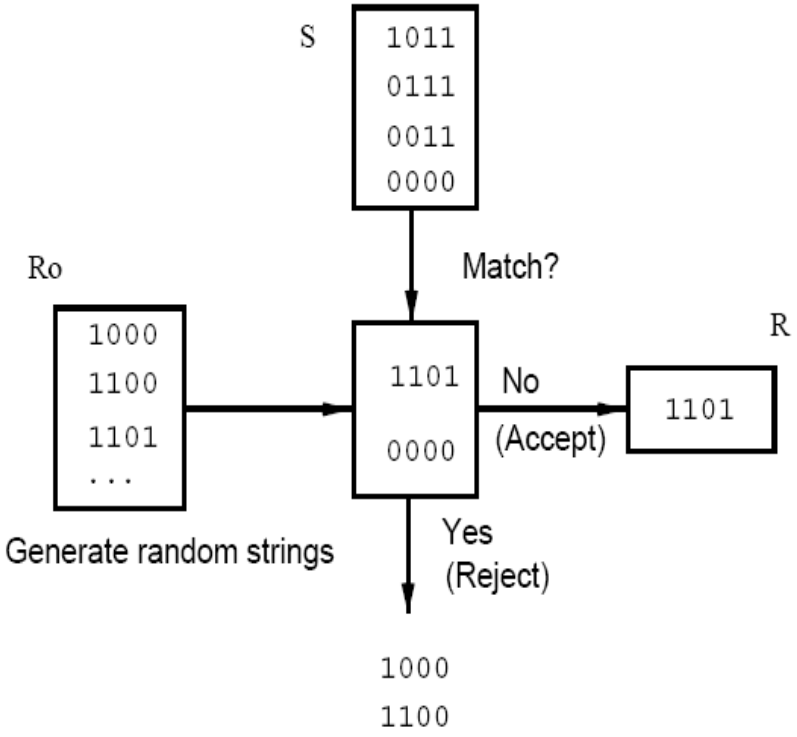


Figure 3.1: Constructing a set of detectors R for a set of self strings S, using **generate-and-test (algorithm discussed later)**. The matching rule used here is "r-contiguous-bits" with r = 2. The strings in Ro are generated at random. The ones that match any of the strings in S are rejected. The ones that do not will form the repertoire of detectors R Extracted form [6]

The problem of computational time arises when looking at the power and time needed to generate a sufficient set of detectors. Resources are wasted as many of these strings will be eliminated. Optimally we would like to be able to generate the end set immediately without going into the process of elimination. Later some algorithms will be presented as a solution to this problem.

The next figure (Figure 3.2) will provide a visual display of the matching process in sets of data strings and how they relate to each other.

Figure 3.2 provides a visual representation between the strings in question [6]. String space U and detector

space Ud are drawn separately for clarity, even though in this paper U = Ud.

The following notation is as presented in [6] and has to be kept in mind throughout the parts of this chapter:

- *Matches(P) = Q – means: "P matches Q" if Q contains all the strings matched by any detector in P.*
- *MatchedBy(Q) = P – means: "Q is matched by P" if P contains all the detectors matching any string in Q.*
- *Matching probability Pm : given a matching rule, probability for a match to happen between a randomly chosen string and detector*
- *Failure probability P_f: probability that a single random non-self string will not be matched by any of the detectors in R.*
- *We write N_X for the size (cardinality) of a set X. In particular, N_S is the size of the self set and N_R is the detector set size.*

- *P_f denotes the probability of failure that N_R detectors have in detecting an intrusion.*
- *N_S is the number of self strings.*
- *N_R is the number of detector strings .*
- *N_U is the total number of strings.*
- *P_m is the probability of a match between two random strings.*

Given a set of self strings S belonging to U, the goal is to find a detector set R belonging to U that matches as many of the non-self strings in N as possible (N = U-S), without matching any of the self strings in S. The detectors in R are chosen from the set of candidate detectors C, consisting of all strings that do not match any string in S (i.e., C = U - MatchedBy(S)). If we were to include all candidate detector strings in R, we would be able to detect all non-self strings in set N`, which may or may not be equal to N. Generally, only a small subset of candidate detectors will be included in R, allowing us to detect all non-self strings in D (with D belonging to N` belonging to N). Failure probability $P_f = N_D/N_N$, or $P_f \approx N_D/N_U$ if $N_S < N_U$. [6]

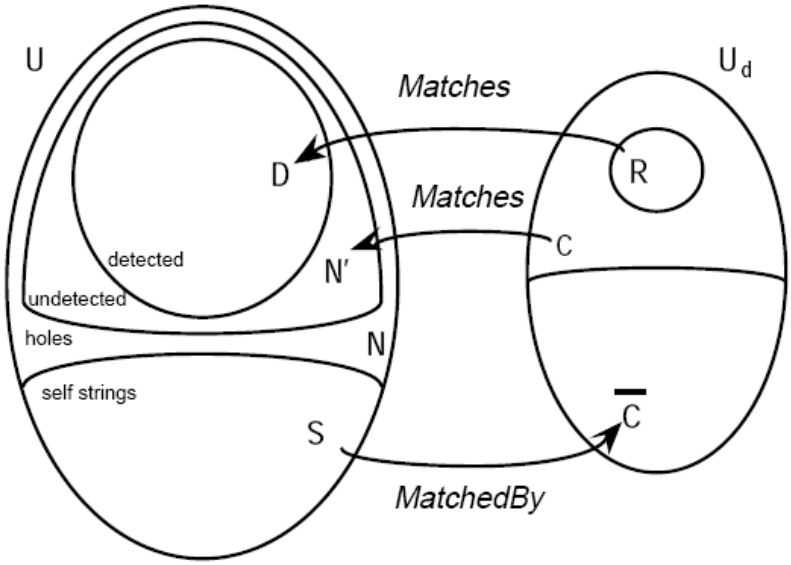


Figure 3.2: Relationships between the subsets of string space U and detector space U_d. String space is partitioned into self strings S and non-self strings N. Candidate detectors (C) are those that do not match any string in S. The detector set R is a subset of the candidate detectors. Non-self strings can be further subdivided in detectable non-self strings (N´) and holes (H = N - N´, not labeled). Detectable non-self strings can be detected (D) or not (F = N´ - D, not labeled), depending on the choice of R. Extracted form [5].

Further reading will introduce the reader to methods devised for the process of negative selection of the detectors. **The methods will be described assuming that a definition of self does exist**. This has been dealt with previously and assuming that a sense of self is gained, these methods are introduced.

The process of anomaly detection, assuming that self is defined, has the following advantages in comparison to the traditional menthols of intrusion detection. Theses are stated in [6].

- **There is no need for a prior knowledge of intrusion** as opposed to the traditional methods that rely on non dynamic methods..

- **Detection is probabilistic but tunable**. The human immune system does not hold a set of lymphocytes of such size, that all non-self structures can be recognized. Similarly, in the case of computer security, it is not possible to maintain a set of detectors that cover all non-self strings. The

set is given a limit in size according to a predefined failure probability that has been deemed acceptable.

- **The detection scheme is inherently distributable**. Small sections of the protected object can be checked separately. Different sites can have independently created sets of detector sets for the same object, and the detectors in the detector set can be run independently. Communication between sets is not needed until an anomaly is detected. This makes this a promising tool for use in networked environments or an architecture with autonomous agents such as the one presented in [6].

- **Detection is local**. Classical change detection methods, such as checksums or hash functions, need the entire data set o be checked at once. The method presented here allows for a small section of the data to be checked and when an anomaly is found, it can be localized to the strings that the detectors is checking [6].

- **The set of detectors at each site can be unique**. This analogizes the diversity of the immune system between different individuals of the population. A failure in one site does not mean that the other sites would fail too. This generates a level of diversity, therefore, if one site is attacked and penetrated successfully, other sites will not be vulnerable to the same intrusion [6].

- **The set of self strings are the set of detectors are mutually protective**, meaning that they protect each other from changes. Also detection is local. A string or detector that has been changed can be pinpointed locally [6].

A comment that can be made at this point given the above described method would be to just use checksums. [6] It has to be said at this point that the use of checksums is just a very specific case of the above description. With checksums there is only one object that is protected and one single detector, the checksum. Matching rule consists of analyzing the result obtained from calculating the checksum of the protected object in comparison to the one we already have form before.

The above method can be considered as a generalized checksum method. Given the appropriate choice of parameter, it can protect N_S (N number of self strings) files of file sections using N_R (N number detectors in the detector repertoire) of checksums. The detection would be, in this case, distributed and localized. There is the advantage of tunability through which, it can be used as an detection method for anomalies in dynamic environments. Admittedly, this is a very large advantage, as in the case of its application to intrusion detection, it is unreasonable to look for single bit changes.

A noticeable issue to be considered in the implementation of such a system is the degree of freedom in parameter choice. In the next sections we will be looking at analysis that is aimed to the derivation of

boundaries for these parameters.

Further sections will also deal with the presentations of some algorithms that have been designed for the purpose of efficient generation of detector sets as presented in [5].

3.3.1 Some Analytical Points

In the paper [6] that is being studied in this part of the chapter, the theoretical analysis mentioned above is divided into three sections that deal with three key issues for negative selection in the context of anomaly detection
As presented in [6] these issues are:

- Deriving an estimate of the amount of information that is being lost by the division of a data stream into unordered and independent strings.

- Deriving a set of bounds for the size of the detector set to be generated. As mentioned previously, a complete set can not be maintained due to computational power issues. The choices should be made according to an acceptable probability that a detector will match a non-self string. This is the probability previously mentioned P_m.

- The issue on non detectable non-self strings will be also shown. This will illustrate the fact that, independently from the size of the detector set, there will always be a set of non-self strings that is not possible to detect. I would ask the reader at this point to keep this part in mind as it will be shown later that this issue is also possible to tackle.

3.3.1.1 Splitting the Data Stream

First I would like to familiarize the reader with the concept of entropy. As defined by [25], entropy is a numerical measure of the uncertainty of an outcome. Keeping this in mind let us go into the analysis.

As previously mentioned, when gaining a sense of self, through the windowing process, when the database is created, we are splitting a large sequence or string S into multiple subsets or subsequences NS of that initial string.
The method presented previously for the detection of anomalies works over random and unordered sets of strings. In real life, data sets, such as files behavioral patterns of process, sequences of system calls or even running programs are not unordered. For the method to be applied in a real life situation, the data that is to be analyzed must be preprocessed. This means that the data must be split into strings of a certain size over an alphabet (in this example as by [6], the alphabet is binary). A negative consequence of this splitting is that some of the internal structure of the data, that could be essential to make the detection more efficient, is lost. Because we would like to gain as much information about self as possible, the information loss must be minimized. The analysis presented here assumes a binary alphabet. The method does however work for larger alphabets too [6].

It is stated in this paper that, the information lost by splitting a stream $\hat{S}$ into a set of unordered string S is equal to the amount of information (in bits) we would need to be given in order to reconstruct the original stream from the unordered set. This can be denoted $H(\hat{S} \mid S)$. This is the conditional entropy of $\hat{S}$ given S. In information theory this quantity is known as the **doubt**. This because it represents the amount of information about $\hat{S}$ that is missing in the encoding S. For a given string of length l, a stream $\hat{S}$ of LS bits will split up into a set S of N_S l-bit self strings. In most cases not all of the divided strings will be unique, therefore S is really a multi set. Suppose that we are faced with k unique l-bit strings, where each string si appears Ni times. To express this in a mathematical manner we would have the following formula:

$$\sum_{i=1..k} N_i = N_S \ .$$

Given this set of substrings, the original stream will be one of the possible following arangements:

$$\binom{N_S}{N_1, N_2 \cdots N_k} = \frac{N_S !}{N_1 ! \cdot N_2 ! \cdots N_k !}$$

Because we are not given information about the initial data stream, we must assume that each of the above arrangements is a possibility and all arrangements have the same probability of being the original stream. The following equation [6], in this case shows the amount of information i**n** **bits** lost in this

splitting process.

$$\Delta I_1 = H(\hat{S}|S) = \log_2 \begin{pmatrix} N_S \\ N_1, N_2 \cdots N_k \end{pmatrix}$$

where ΔI represents the amount in bits of the lost information.

If Sterling's approximation of the factorial function is applied to the above equation, we will have the following shown. This is similar to the reasoning in [6]

$$\begin{pmatrix} N_S \\ N_1, N_2 \cdots N_k \end{pmatrix} \leq 2^{N_S \cdot H(N_i/N_S)}$$

In the above equation, $H(N_i|N_S)$ represents the entropy associated with the relative frequencies of the occurrence of the sting in S. Taking this through a transformation through a base 2 logarithm, which is the size of our alphabet, this becomes:

$$\log_2 \begin{pmatrix} N_S \\ N_1, N_2 \cdots N_k \end{pmatrix} \leq N_S \cdot H\left(\frac{N_i}{N_S}\right)$$

"Or in the form of a big Θ notation" [6]:

As by [25], thin notation represents "The theoretical measure of the execution of an algorithm."

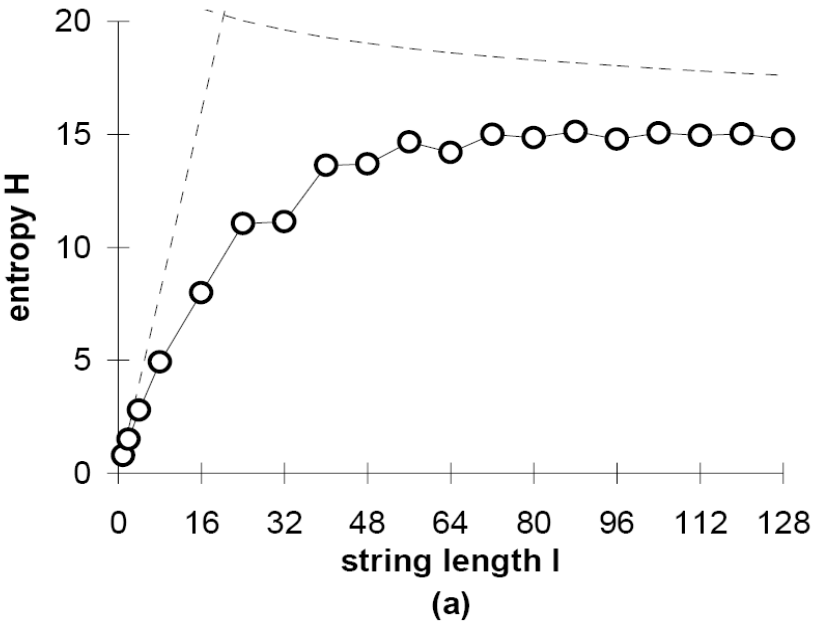
$$\log_2 \begin{pmatrix} N_S \\ N_1, N_2 \cdots N_k \end{pmatrix} = \Theta\left(N_S \cdot H(N_i/N_S)\right).$$

The reader will recall that we are trying to get a bound on the lost information. According to the above equations [6] this will be represented as follows, where ΔI_1.

$$\Delta I_1 = \Theta\left(N_S \cdot H(N_i/N_S)\right) = \Theta\left(L_S \cdot H(N_i/N_S)/l\right)$$

To explain it now in simple words, what the above analysis is trying to show is that if we measure the average entropy per bit on the sting S for different lengths l of this string, the information loss could be minimized by choosing the value of l in such a way that the entropy per bit is minimized.
A careful choice needs to be made in the process of choosing l, because if l is small, the entropy is minimal too, however if l is too large, beyond the point where most strings in S are unique, this will also reduce the entropy as it reduces the number of strings in S. But, choosing a large l will make the generation of detectors difficult in terms of space and time required for the generation process. Computational expenses will be too heavy.

I would like to drag the reader's attention to the following diagrams as shown in [6][5] for some visual clarity.



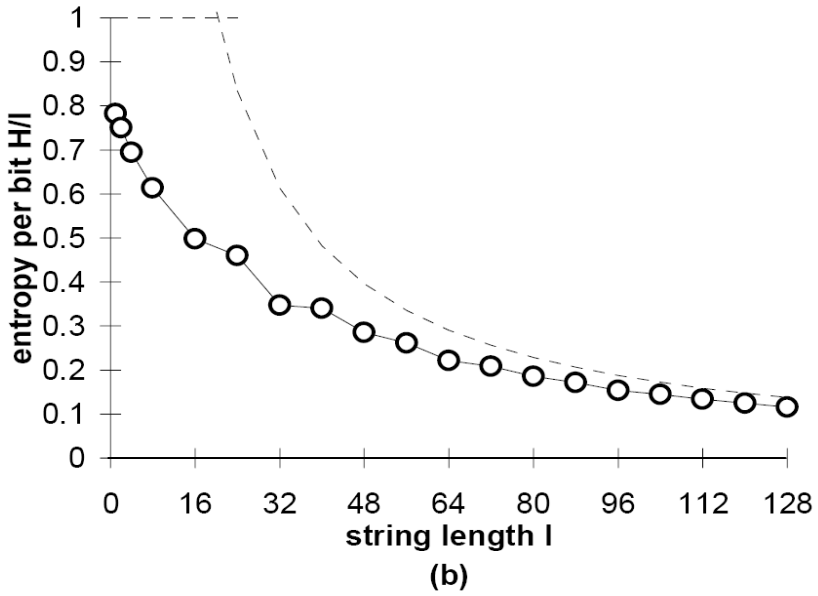


Figure 3.3. In the previous figures, it is shown, the entropy of the set of strings obtained by splitting a binary into l-bit strings. The dotted lines indicate the envelope within which H must fall. This is delimited by H/l = 1 and H = $\log_2(N_S)$. Figure (a) shows the entropy $H(N_i/N_S)$ of the resulting set of strings. Figure (b) shows the entropy per bit of these strings which is proportional to ΔI_1. Figure extracted from [6] [5]

As extracted from [6] the previous figure shows changes in the entropy with respect to l for a real data file.

This is what is said in [6]:

The data chosen was an emacs binary (GNU emacs v19.25.2 SGI binary, 3.2MB). The file was split up into strings of length l=1 (single bits), 2, 4, and multiples of 8 (i.e. at byte boundaries) up to 16 bytes long. For each value of l, the entropy of the resulting set of strings was calculated. Figure 3.3(a) shows clearly that there are minima in the entropy at multiples of 4 bytes, corresponding to the 32- bit RISC instruction used to compile the binary. The values plotted here at 8-bit intervals are themselves local minima as well: for intermediate values of l the entropy would be much higher, because this corresponds to cutting up the data at nonbyte boundaries, ignoring the natural byte structure present in the data. The entropy per bit plotted in Figure 3.3(b) is proportional to DI1 (apart from a constant factor LS). This graph is uniformly decreasing, so in order to minimize information loss for this data we would want to choose l as high as our other constraints allow, with a preference for multiples of 4 bytes. Note that most of the gain occurs at lower string lengths, so increasing string length beyond 64 bits (8 bytes) gives little extra improvement.

The calculation for the information loss by splitting the data at l-bit boundaries is a widely applicable concept that can be used in any method that transforms data into sets of unordered strings. It would particularly hold for methods that take into account the frequency of occurrence of the self strings.

Now, because this method ignores the frequency of concurrence of each unique l-bit sting s_i, we have a second source of information loss. Assuming that N_S is known, if given the set of unique k strings in S(s_1, s_2, ..., s_k), there are:

$$\binom{N_S - 1}{k - 1}$$

possible ways of assigning the values for N_i, in a way for them to sum up to N_S, and reconstructing the unordered multiset S. Again assuming no prior knowledge of the contents of the data stream, if each of those assignments is equally likely, the amount of information loss through ignoring the frequencies of the strings is shown by ΔI_2 as follows:

$$\begin{aligned} \Delta I_2 &= H\left(S \middle| s_i\right) \\ &= \log_2 \binom{N_S - 1}{k - 1} \\ &\leq \left(N_S - 1\right) \cdot H\left(\frac{k-1}{N_S - 1}\right) \\ &< N_S \, . \end{aligned}$$

ΔI_2 would generally be small compared to ΔI_1 from before and is ignorable in practice. If ΔI_2 is to be minimized, this can be done by minimizing NS, through choosing a large l. Duplicate strings would be reduced and therefore the loss of information through the ignoring of them. This however would cause an increase in ΔI_1 because of the increase in $H(N_i/N_S)$.

However, I would like to make a point here. I do not see why entropy loss should be the best way of choosing the size of l. Why not choose l in such a way to eliminate the concurrency of strings for example. This would immediately create a more diverse database, without redundant entries. However, it always has to be remembered that the choice of l must be done with computational expense in mind.

3.3.1.2 A Summery of the Splitting Process

In the previous section, the reader has been presented with a rather large and overwhelming amount of information and mathematical analysis about the whole procedure of how the given data segment should be processed in order to be split into independent strings. I would like to try and summarize it in a few simple words now without using mathematical formulas and equations simply to make the understanding easier.

There are two main thing to be kept in mind:

1. When dealing with processes that require a large amount of computational power and time, things have to be kept simple to minimize the use of resources. How good is an intrusion detection system that increases the time of any process running through the analysis it does by many fold, takes a lot of memory on both the hard drive and the RAM.

2. The splitting process described in the previous section assumes that no record is kept by the detector system about the previous state of the data set. Once the data given is split into strings, that is all the system has for the analysis. Before the splitting process occurs, these strings form a sequence that in some way creates a dependence between the substrings. The splitting creates a loss of information in the context of the previous amount of the data. Through the splitting we are loosing information about what is happening in these exchanges.

What is trying to be done in the above section is the derivation of a suitable length for the substrings into which the data set is being split. This split should be done in such a way that the loss of information that results from this splitting process is minimized. This is the information loss minimization concept mentioned above.
The splitting needs to be done in such away that the information loss is minimizer, therefore the importance of choosing the right size for l through the methods shown above. Also, in needs to be done in such a way to avoid the over repletion of the same sequences as this would not be beneficial to the computational time needed and also it would not be beneficial to the entropy either.
The equations seen above try to find the correct size for the measure l in the following manner. Given a choice of l, and the substrings constructed with this length, there are a number of possible reconstructions that can be done through the reordering of these strings in order to get the original data. The larger the number of these possible scenarios of reconstructions, the larger the information loss will be. The size of the length of strings in the division process needs to be chosen in such a way that the trade off between this number of reordering and the computational expense is acceptable.

3.3.1.3 Detector Set Size

As mentioned earlier, it is not feasible to maintain a detector set that covers all possible non-self strings derived from the data that is being analyzed. The detector strings are randomly generated. A way of generating these strings is needed in order to make the set as efficient as possible while trying to minimize the size of it for computational expense purposes.

The theory presented below is extracted form [5]. It will be presented as it is shown in the paper and following this is will male a brief summarization of it in simple words to explain the process.
It is possible to derive theoretical lower bounds for the needed number of detectors needed. This is an important factor in the decision of parameter setting. So, what is the number of detectors needed to generate a specified failure rate P_f? The answer to this question will affect time needed for the generation of these detectors, storage space, amount of non-self space covered by each detector, etc.

P_m will denote the probability that a randomly chosen detector and a non-self string with match according to the matching rule used [5]. From this, a first lower bound, N_R, can be derived. In the best case, we can spread the detectors apart in such a way so that no two detectors match the same non-self string. The amount of string space covered by N_R detectors is then $P_m N_R N_U$. This needs to be at least as big as $(1-P_f)N_U$, so that it can cover enough non-self string for a failure probability of P_f, so, ignoring N_S with respect to N_R:

$$N_R \geq \left(1 - P_f\right) / P_m$$

A further lower bound can be derived based on the amount of information about S that can be stored in R. A set of detectors R constructed for a given set of self strings S can be viewed as a message encoding information about this S self set. Given a "perfect" detector set R`, that is, a set of detector strings that exactly recognizes all non-self strings, it would be possible to reconstruct the original self set S (by checking for each string in the string space whether it is detected by any detector in R´). Therefore, this set R` would have to contain at least the same information as the original set S. By calculating the information content of the original file of self strings it is possible to get an absolute lower bound on the information content, and thus on the size, of the detector set. For a random self test, the information content will be of the same order as it's size in bits. The meaning of this is that the perfect detector set R` would have to have approximately the same bit size as the self set S. This is not an acceptable assumption in the case of very large self sets.

The problem above, can be alleviated through an allowance in the sense of an acceptable amount of error in the self/non-self discrimination performed by the detectors. In this analysis it is assumed that a fraction P_f of the non-self strings are not matched by the detectors. Denoting U as the total string space, and $N_U = |U|$, the

assumption allows at most $P_f(N_U - N_S)$ non-self strings left undetected. This is an allowance for false negatives, however false positives, self strings being detected as non-self is not allowed.
Given that an exact encoding of the self set S is not required, means a smaller information content therefore, a smaller detector set.

Choosing the self strings from the set of possible string U then there are:

$$M = \begin{pmatrix} N_U \\ N_S \end{pmatrix}$$

possible self sets of size N_S. Assuming (i) that the set of self strings is chosen at random out of the set of all possible strings, therefore meaning that they are independent, it means that any of those strings is equally likely. The average information content, or the entropy of the self test of size N_S will be:

$$H(S) = \log_2 \begin{pmatrix} N_U \\ N_S \end{pmatrix}$$

Given a detector set R constructed for this self test S, we could try and reconstruct S from R. Similar process to the one described earlier. Given that R has been constructed with a measure of fault tolerance in mind, it is not feasible or possible to reconstruct S exactly. Instead we would get a set S` that contains all the original self strings and in addition it will also contain a set of unmatched non self strings of $P_f(N_U - N_S)$.

In assumption (i) made above we have stated that the strings are independent of each other. A second assumption is made by the analysis at this point (ii). Given S` we have no knowledge of of which subject of N_S strings constitutes the original set S. The missing amount of information about S, which is not in the encoding R can be denoted by $H(S|R)$. This is the amount of information that would be gained if S was given to us when we already have R.

The assumptions made above, (i) and (ii) make us deduce that any subset of size NS taken form the set S` of size $NS + P_{f(}N_U - N_S)$, is equally likely to be the original self set S. Considering the following:

$$H(S|R) = \log_2 \begin{pmatrix} N_S + P_f(N_U - N_S) \\ N_S \end{pmatrix}$$

The difference between H(S) and H(S|R), where H(S) is the information is S and H(S|R) is the information about S missing in R, shows the information of S that is preserved in R. This is called the **Mutual information of S and R**. It is denotes by I(S;R). This can be worked in single details through the use of the following equation:

$$
\begin{aligned}
I(S;R) &= H(S) - H(S|R) \\
&= \log_2 \binom{N_U}{N_S} - \log_2 \binom{N_S + P_f(N_U - N_S)}{N_S} \\
&= \log_2 \left[\binom{N_U}{N_S} \Big/ \binom{N_S + P_f(N_U - N_S)}{N_S} \right] \\
&= \log_2 \left[\frac{N_U!}{N_S! \cdot (N_U - N_S)!} \Big/ \frac{(N_S + P_f(N_U - N_S))!}{N_S! \cdot (P_f(N_U - N_S))!} \right] \\
&= \log_2 \left[\frac{N_U}{N_S + P_f(N_U - N_S)} \cdot \frac{N_U - 1}{N_S + P_f(N_U - N_S) - 1} \cdots \frac{N_U - N_S + 1}{N_S + P_f(N_U - N_S) - N_S + 1} \right]
\end{aligned}
$$

The analysis at this point makes a third and last assumption (iii). This assumption is as follows:

$P_f N_U > N_S$ and therefore, $N_U > N_S$.

The above assumption simplifies the above equation as this:

$$
\begin{aligned}
I(S;R) &\approx \log_2 \left[\frac{N_U}{P_f N_U} \cdot \frac{N_U - 1}{P_f N_U - 1} \cdots \frac{N_U - N_S + 1}{P_f N_U - N_S + 1} \right] \\
&\approx \log_2 \left[\left(1/P_f \right)^{N_S} \right] \\
&\approx N_S \cdot \log_2 \left(1/P_f \right).
\end{aligned}
$$

Since R needs to contain this much information in the least, this is a lower bound on the size of the detector set in bits. If the detectors are strings of length l over an alphabet of size m the required size of the detector repertoire N_R is given by:

$$N_R \geq \frac{N_S \cdot \log_2\left(1/P_f\right)}{l \cdot \log_2(m)}$$

In assumptions (i) and (ii) this analysis states that the self strings are independent from each other. This is, almost always, not be the case in real life. Both assumptions will be affected by this issue. Assumption (i), that self stings are independent, led to the formula H(S) as it was allowed to assume that all self sets are equally likely. This will hardly ever be the case in a real life situation. Real life data is most likely going to show some kind of structure. This will lead to a larger similarity between self strings than random strings. This fact is something that will have an effect on H(S) as the real data will probably form only one of the subsets of all possible sets. Any single real data set may contain less information than a random one. However, by having only one data set, it is not possible to analyze how likely this set is and how much information it contains. A more high-level knowledge is required for the determination of the entropy of one single set. If this information is at hand, it could also lead to a smaller encoding, a smaller H(S).

The effect that non independent self strings would have in the second assumption (ii) are a bit unclear according to both [5][6]
As stated by both of these papers, if the $P_f(N_U - N_S)$ undetected non-self strings in S` are dependent we might find some information about which subset NS of strings are most likely to be the original self file S and therefore $H(S|R)$ can be smaller. The character of the non-self strings that are left undetected will be dependent on the matching rule and repertoire generation algorithm used in a specific case.

The third and final assumption (iii) made states that $P_f N_U > N_S$. It would be generally acceptable to assume that at least $N_U > N_S$. Should this not be the case then almost all possible detector strings would match to at least one self string and most non-self strings would in fact be undetectable holes.
This assumption holds depending on the acceptable rate of failure set. Take a look at the table below. It illustrates this theoretical lower bound for N_R as a function of string length l for a data stream of one million bits. Note that for a tenfold decrease in failure rate we only need a twofold increase in the number of detectors.

l	N_S	$N_R\ (P_f = 0.1)$	$N_R\ (P_f = 0.01)$
16	62500	12977	25953
20	50000	8305	16610
24	41667	5768	11535
28	35715	4238	8474
32	31250	3245	6489
36	27778	2564	5127
40	25000	2077	4152

Table 3.1: lower bounds of N_R for a 1 Mbit long data set for different sting lengths. As seen in [5][6]

For $P_f = 0.1$, every single non self string has a 10% chance of escaping from the detectors. If an intrusion consists of only one non changed string, this may be a large error rate and is not acceptable. However, intrusion consist of multiple non self strings and it is sufficient to recognize only one of these to activate a reaction. If we have to use a much lower failure rate the assumed approximation may no longer hold [6].

3.3.1.4 Detector Set Size – A quick summarization

Once again I would like at this point to make a quick summarization of what is happening in the above section, for the reader to have a more condensed view and simpler way of understanding the above described analysis and concepts.

In these type of systems we will often find the need for tunable parameter as they are dynamic systems. Their dynamical nature requires some settings in the form of these parameter for the to achieve the desired effect. This is also done so that they do not consume enormous amounts of resources that are needed to perform the self tasks.

The analysis presented above is trying to decide an appropriate size for the set of detector agents of the

intrusion detection system that will perform to the desired extent. The size should be chosen with two points in mind:

1. The size of the repertoire of these strings that are to be used for detection must be as small as possible in order to economize the resources and not match self strings.

2. The size should be chosen so that the above condition is satisfied and preferably if not all, a very large space of non self strings is covered.

The tunable parameter P_f denotes the acceptable failure rate, or more simply, the amount of non-self strings that can be left undetected. Given this acceptable failure rate and the set of self strings S the analysis is trying to define a method for the creation of a set of detectors R in such a way to minimize the loss of information from S in R. At the same time it is trying to keep the repertoire of R as small as possible keeping in mind the efficiency requirements. Given that we have an acceptable failure rate, some loss of information is acceptable. The amount of the information that can be lost is a main determinant of the size of the detector set.

Taking a look at the previous section again, $H(S|R)$ denotes the entropy that we would have supposing that S is given to us and we know R.
The average information of the self set N_S is given by H(S).
The difference between H(S) and $H(S|R)$ is the amount of information about S that is being preserved in the encoding R. This is denoted by I(S;R). Being able to tune this parameter would make it possible for us to define an acceptable set size for the repertoire of detectors given the acceptable failure rate.

However, as stated in the latter part of the previous section many assumption are made here which would not be acceptable in real life situation. Therefore the choice of the acceptable failure rate is what determines the size of the repertoire of the detectors. A visual example of this is given in the previous section in the table.

N.B. The acceptable failure rate is not a performance measure as such. It only states a percentage of acceptable failure in the detection of intrusions. So, for $P_f = 0.05$, we can assume that if 5% of the non-self string menage to intrude and are not detected, this is acceptable. However if, even for a 5% acceptability level of false negatives, the actual level of them is lower, this does not mean that there is a problem. All it means is that it is acceptable to decrease the detector set without breaching the requirements.

3.3.1.5 The Existence of Holes

It can be deduced from the previous two sections that in order to achieve the desired failure probability all that needs to be done is to choose a large enough detector set. However, there might be a lower bound on the level of achievable failure probability as well. This is largely dependent on the specific matching rule used. Earlier in this chapter a matching technique is shown, the "r-contiguous bits". There are a number of detector generating algorithms that can be used for the generation of detectors which are to be discussed later in the chapter and can be seen in [6][5]. These make it possible to generate a complete repertoire of detectors. This means that such algorithms can generate all non-self strings that can be covered.

However, according to [6], there is a set of non-self strings in the case of the r-contiguous bits matching rule, called "holes", for which it is not possible to generate valid detectors. The explanation of this is as follows and it is extracted from [6]
If S contains two strings s1 and s2 that match each other over at least (r-1) contiguous bits, they may induce two other strings h1 and h2 that can not be detected because any candidate detector would also match one of s1 or s2. This illustrated below for two 16 - bit strings with r = 7

s1: 0000**010101**000000
s2 1111 **010101** 111111

this would lead to:

h1: 0000**010101**111111
h2:1111 **010101**000000

The previous example as shown in [6], illustrates the existence of holes in the case of use of the r-contiguous bits matching rule which was chosen throughout the analysis.

However other matching rules may be used and, such as the **Humming Distance Rule**. It is also shown in [6] that holes exist in the case of use of this rule too.

For this illustration the reader must be familiar with two concepts as follows:

- **Cartesian Plane**: A two dimensional plane where all points can be represented as coordinates. [25]
- **Euclidean Distance**: The straight line distance between two points on a plane. [26]

When using a Hamming distance rule (two strings match if their Hamming distance is less then or equal to a

fixed radius r), such a situation is also possible. The next figure, extracted from [6], shows this point using a diagrammatic approach on a Cartesian plane. A match between a string and a detector is said to occur if the Euclidean distance between the corresponding points is less than a predefined radius r. Figure 3.4 illustrates holes visually.

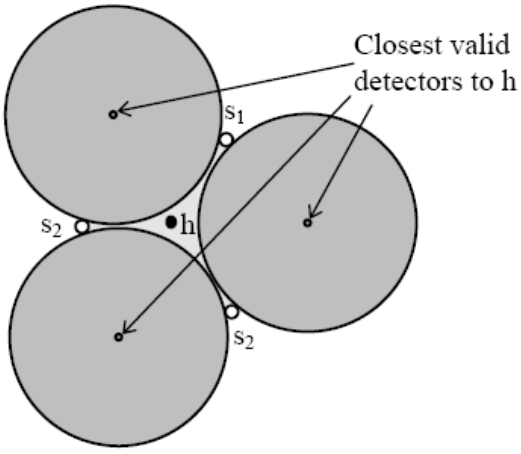


Figure 3.4. In this figure, if self points are presented as points in a plane and the matching rule is Euclidean distance, the three self strings s1, s2 and s3, may include a hole h for which no valid detector can be generated.[6]

The analysis tries to show that all practical matching rules with a constant matching probability, meaning that each detector matches equally many strings and vice versa, can exhibit holes even with non trivial set of self strings.
Let us have the following example as by [6]:

U = set of all strings
S = set of strings to be protected
Matches(d) = the strings matched by detector d
Matched(s) = the detector string that s is matched by

Given string h and a matching rule M with constant matching probability P_m, the existence of a hole can be shown by generating a set S such that h is a hole.

Starting with the empty set for each detector d matching h, a string s_d is picked so that it is matched by d and added to S. Then, by construction, for each possible detector h there is an existent self string matching that detector. Set S, constructed in this manner will be non trivial if it does not need to contain a significant portion of the total space denominated by U.

The existence of these holes imposes a lower bound for the probability of failure achievable with a given detection method. This happens because it will always fail to detect holes. At this point some techniques for working around this problem are presented.

If we calculate the required number of detectors to achieve a certain acceptable failure probability, without taking holes into account, the achieved probability in real life, would mos likely be higher than the desired one. Failure probability associated with holes, also does not improve by distributing the algorithm through different sites if the same matching rule is adopted throughout the sites. Even though detector sets,generated at different sites, would most likely be very diverse, holes would probably be quite similar. It might be useful to spread out the self strings apart as much as possible, thus reducing the holes between them. This is achievable through the randomization of each self string with a hash function. It is also possible to use detectors with a larger specificity. This would correspond to a smaller radius in the previous graphical representation. Of course this would mean a larger detector set.

A further approach for a solution is to try an eliminate holes entirely. This can be achieved locally or in a distributed fashion. [6]

Locally it can be achieved through the use of a matching rule that does not show holes. In the case of the much mentioned r-contiguous bits rule, we can choose a value for r so that 1(number one) <= r <= l(letter l). Where detectors will match anything between the entire string space and a single string, itself, such that potential holes can be filled by detectors with very high specificity. The value of r would have to be stored with each detector.
In a distributed fashion, where different machines run their own detector sets, a different matching rule can be chosen for each machine. In this case, it is thought that each machine would have a different set of holes that would be covered by the detectors form other machines.

The existence of holes is not necessarily something to be viewed negatively. Given that these holes are generated by interactions of self strings, they will tend to be very similar to them. Case in which, given a level of tolerance, we would not want to be detecting strings that are so similar to self. [5][6]

3.3.1.6 The Existence of Holes – A Summary

The previous analysis is trying to illustrate the existence of non-self strings that are not possible to detect. This meaning that a detector can not be generated for them as it will be matched by a string in the self set and therefore not accepted.

What the analysis is trying to prove also is that the existence of holes is not necessarily a negative thing. The reason for this is that we would not like to have detections that are so close to the self set. This could lead to a legitimate self string being rejected.

However, I would like the reader to keep the existence of holes in mind as further a method for dealing with such issue will be introduced. This is the counting the holes method.

3.4 Negative Selection Methods

This section is intended to present the reader with a number of algorithms that are designed for the purpose of generating the detector sets. Three different algorithms will be presented. These are extracted from publicly available research papers such as [5][6]. These algorithms present methods for the generation of the detectors to be used in the process of self/non-self discrimination. A good analogy is made here from the methods of work of the human immune system. The particular way that these algorithms work are not of a large importance as they are just based around the same principle, therefore it is not necessary to have a detailed knowledge of them One of them will be presented in detail. The next three are nothing more then step by step improvements of each other, each posing better ways to generate the detectors making improvements on the time and space needed for the generation of the detectors and improving the space of non-self strings covered by them. This study is dealing with the possibilities of embedding immune system principles into the are of intrusion detection. The existence of these algorithms will be a good knowledge in the evaluation to be done later. Also a good overview of the advantages of each will be deducted, however the detailed work of each and every single one of them is not of high relevance to this study. This will not be included in the report, but appropriate reference to the resources will be given.

3.4.1 The Generate-And-Test Algorithm

Initially presented in [5] and later also in [6], this algorithm analogizes the generation of lymphocytes in the immune system. The immunological research on this can be found in the first chapter.

The following is a presentation of the generate-and-test algorithm as shown in [5].

In this algorithm, strings are randomly chosen from the set of all strings U_d and then matched against the set of self strings S that are to be protected. If the randomly generated strings do not match any of the self strings they are kept as valid detectors. This process of random generation and checking is repeated until the desired amount of detector strings is created.

N.B. Recall the previous section on ***Detector Set Size.***

Given a predefined matching probability (which was dealt with previously) the Generate-and-Test algorithm requires the generation of a repertoire of random detector strings. This is the initial repertoire of strings before the negative selection. It is denoted by N_{ro}.

$$N_{R_0} = \frac{-\ln(P_f)}{P_m \cdot (1-P_m)^{N_S}}$$

For independent detectors it is feasible to approximate the failure probability P_f achieved by N_R detectors as follows:

$$P_f \approx (1-P_m)^{N_R}$$

For P_m sufficiently small and N_R sufficiently large, this gives the following:

$$N_R \approx -\ln(P_f) / P_m$$

The assumption that detectors are independent is not entirely valid. With the increase of NS or Pm, the candidate detector set (this would be C in fig. 2) will reduce in size, so the chosen detectors have a lower level of independence. Overlap among the detectors decreases the covered amount of string space. This results in a higher value of the probability failure rate than the one indicated previously in the following equation:

$$P_f \approx (1-P_m)^{N_R}$$

The time complexity of this algorithm, as indicated in the different references [5][6], is proportional to N_R, the number of candidate detectors that need to undergo negative selection, and N_S, as each candidate may have to be compared to all self strings. The time and space complexity as denoted by N_S are as follows:

Time:

$$\mathcal{O}\left(\frac{-\ln(P_f)}{P_m \cdot (1-P_m)^{N_S}} \cdot N_S\right)$$

Space:

$$\mathcal{O}(l \cdot N_S)$$

3.4.2 The Generate-And-Test Algorithm – A brief summarization

The generate-and-test-algorithm makes use of an exhaustive method for the generation of the non-self set. Meaning that all possible detectors are generated randomly and compared to the self string in order to be negatively selected. The generation stops when the desired size of the detector set has been reached.
The steps:

1. A repertoire of detectors is randomly generated.
2. Detectors are checked against self strings.
3. Detectors matching self strings are eliminated.
4. Detectors that do not match self strings are added to the negatively selected repertoire.

This algorithm seems a quite good and efficient one. On the plus side, it is simple. But, the computational expense is of exponential time to the self set. That is something that is generally not positive. Exponential times are not preferred. If the self set is too high, then the problem of computational time taken, grows by a lot more than the size of the self set.
However an inefficiency has to be noted in this algorithm. Most of the candidate detector strings are rejected, therefore an easily noticeable waste of precious resources is to be pointed out here.

3.4.3 Linear Time Algorithm

In this section, the reader is presented with another type of algorithm. This is called the “Linear Time Algorithm”. The name is derived from the time in which it runs with respect to the size of the input.

The algorithm is presented in detail in [5][6]. For the purpose of this study, the detailed description for it is non needed, however it is important to understand the advantages that it has over the previous, generate-and-test algorithm.

The generate and-test algorithm does and exhaustive generation for the detector set, therefore a lot of computational expense is given to the generation of detectors that are rejected through the negative selection method. The linear time algorithm generates the detector set in such a way to tackle the problem of rejected detectors. The algorithm consists of two phases:

- **Phase 1**: The first phase enumerates the way in which, according to the matching rule used, the strings S in the self set can be matched by randomly generated detectors.

- **Phase 2**: The second phase, uses the results from the first phase to generate detectors that are not matched by the self strings.

The time used for this way of generation of the detector set is better than the one of the generate-and-test-algorithm, however there is an increase in the space needed for this algorithm to be used. As stated in [5], the time needed fro this algorithm is proportional in linear time to the self set. The space needed however is exponential to the setting used for r in the matching rule.

There is however one problem with both of the above algorithms. They do not provide any guarantee as to what space of non-self is covered by the detectors generated. The detector set does in fact have a limited size, as analyzed preciously by the probabilistic approach. The greedy algorithm, used a generation method that spreads out the detectors in such a way to achieve a better coverage of the non-self set.

3.4.4 Greedy Algorithm

The previous two algorithms still have one problem. The generation of the detectors is entirely random. Their being random, does not guarantee that the space of non-self covered by them is adequate. Meaning that, if they are high in some level of similarity, sets of them will share coverage over the same amount of space in the non self set. The Greedy algorithm tries to address this issue by not doing a random selection of the detector set. Instead , the selection is done is such a way, that the detectors are as far apart as possible from each other. Details on this algorithm can be viewed in [5].

This algorithm is also composed of two phases:

- **Phase 1**: This phase assumes that there is an existing set of detectors. It makes use of the self set and the existing state of the detector set to set some parameter upon the set of detectors that is to be generated.

- **Phase 2**: The results from phase one are used in the second phase to generate the set of detectors in such a way that as many as possible non-self strings are covered by the detectors set.

The time complexity of this algorithm is quite a bit higher than the time complexity of the linear time algorithm. This is a downside that is to be taken in consideration. Clearly there is a decision to be made here upon a trade off between time an desired coverage by the detector set.
The space complexity of this algorithm is of the same order as the one of the linear time algorithm.

3.4.5 Counting the Holes Method

The previous algorithm is in some way an improvement of the linear time algorithm in the context of the space coverage that it provides for the detection of non-self strings. However, as previously discussed, [6] even with a set of detectors that would cover the entire non self strings, the existence of holes is not possible to avoid. The Counting the Holes method can be seen in detail in [5].

The paper in question [5] states that, if the greedy algorithm is run in an exhaustive manner, then there would be a clear way of establishing the number of holes. The number of non detectable non-self strings. However, [27] presents a good method for the counting of holes. It is stated in the paper that the use of this method, would help with the decision of choosing an appropriate set of parameters for the string length and the matching rule parameter.

It is also stated in [5] that, given a distributed type of architecture for the use of these methods, choosing different parameters for the length of the string and the parameter of the matching rule, would contribute towards a smaller number of holes. Preferably different matching rules should be used at different sites. As by the paper in question, this would create a situation where the holes in one site would be covered by the detectors of other sites and vice versa. This would possible in one of the distributed architectures mentioned previously in the chapter.

3.5 A General Summary

At this point I would once again like to make a summary of what has been dealt with. This chapter has introduced the reader to a large amount of information, however only some main points are to be kept in ming for the analysis in the next chapter.
Initially we dealt with investigating into the method for gaining a sense of self through the monitoring of system calls using a windowing method and creating a database that is to be used as a definition of normal behavior.

The next analysis looked into some detail about the setting of the parameters such as the ones used for the matching rule and the size of the detector set. The existence of holes was also shown and it was made clear that they are not possible to avoid.

The next section dealt with the investigation into the different negative selection algorithms created by current research that can be used for the generation of the detectors.

Also a method for counting the holes and dealing with them over multiple linked sites was introduced. However I would like to make a point here. The research extracted form [6] stated that the existence of holes is not necessarily a bad thing as we would not want to be detecting possible non-self strings that are so similar to the self strings. Then, the research extracted from [5] showed that these holes can be covered. I think that these two principles are contradictory to each other.

In the next chapter, I will deal with some personal analysis and present some personal thoughts on all these matters.

Chapter 4

Evaluation, Analysis & Ideas

The previous chapters introduced the reader to a wide selection of methods, ideas, algorithms etc extracted from existing research papers and books. Initially the study dealt with an investigation into the methods and techniques of work of the human immune system. Further into the study, the reader, was introduced to a number of principles, issues, architectures and algorithms that analogize the workings of the human immune system.

The purpose of this chapter will be to evaluate these methods and see to what extent they analogize the

methods of work of the human immune system. This evaluation will present the principles that have been implemented and the principles that have been left out in this analogy. I would like to present such an evaluation in the hope to extract some conclusions concerning the methods that not have been implemented by the previously presented algorithms and methods. The conclusions should deduce as to weather the non implemented methods are possible to analogize and would they even be beneficial at all.

The reader will also be presented to a number of attack examples that should clarify the weakness of these systems if applied in practice.

4.1 System Calls for a Sense of Self

In the previous chapter the reader was introduced to a method for deriving a definition of self through the use of sequences of system calls for each running process. In this method, we are only considering the system call itself. A number of downsides of this method are presented in [7], as follows:

This method for the definition of normal behavior, does not take into account a set of aspects that could be vital in the detection of an attack. Parameter values passed between system calls are totally ignored. Also the timing information for these calls is not taken into account. Timing here meaning the actual time that the call happens during process runtime. Also, this method totally ignores other instructions between these system calls, which are in fact quite numerous and could provide a large amount of information about the running of the process. [7]

To illustrate some of the above points through an example I would like to consider the following piece of code:

```
int main{
  char p[4];
  char user_p[4];
  p = “abcd”;

  get (user_p);
  if (!strcmp (p, user_p){
        allow ( ); }
  else{
        deny ( ); }
}
```

So, the program is creating a buffer of size four for the stored password p, and a buffer of size four for the user input user_p. If the two strings match, access is allowed and if they do not access id denied. So, if the

user enters the password as abcd, access is allowed.

However consider the following possible input by the user:

>>aaaaaaaa.

User inputs a string of eight a-s. Knowing the structure of this buffer, which is a stack. Meaning that is a last in first out type of structure, the first four a-s will get stored in the user_p buffer and the second four a-s will create a buffer overflow and overwrite the stored password p. This is supposing that there are no mechanisms for the protection of buffer overflows. When the comparison happens, the access will be granted. This attack would go totally undetected, as there are no system calls happening that could help it's detection.

This was purely an illustration to show that using system calls would in fact make us miss out on a lot of information that would otherwise be indispensable in the detection of an attack.

4.2 The Architectures

The third chapter introduced the reader to a set of possible architectures for the implementation of such a system. I would like to make a quick evaluation of them and try to detect some downsides for each in order to see which could possibly be the most optimal one.

- **Protecting static data** – This approach is one that is similar to the one currently used by traditional virus scanners when they do a check of the entire hard drive. Now, of course, with the implementation of immunologically inspired methods, the system would have some level of adaptability and not rely solely on known signature, however as stated by [7] this approach could turn out to be not very practical. First of all, it is slow as it is limited by the speed of the hard drive, and secondly there is not much point to it as the data is not going to change and the injected malicious instructions would not have any effect on anything until the infected program is loaded onto memory and run. I personally do not like this architecture either, simply because it would just take a lot of time to run due to the hard drive speed.

- **Protecting processes on a single host** – As clearly stated, in this approach we are monitoring the

active processes. This would create a great advantage in the sense of speed. Active processes would be monitored while in runtime and there would not be hard drive speed restrictions during the monitoring process. I personally prefer this approach quite more than the previous one. However, there are some downsides to this. As the reader will know by now, a training period will be initially needed for the establishment of the self database. Given that this is being done on a single host, and if there is only one single user, the database will be limited to the behavior of that particular user. Therefore the detectors will be created in such a way to be complementary to any other type of behavior. Should a different user need to get access to the machine, in order for his actions to be allowed he must have a very similar behavior to the one of the user that did the training process.

- **Protecting a network of mutually trusting computers** – As mentioned previously, this proposal for an architecture is a very interesting one. A set of computers, belonging to a network are mutually trusted. The processes acting as lymphocytes roam around the network and look for anomalous behavior. As stated in [17], there is one major problem with this system. A solid framework is needed for these processes to be allowed to share core information between all machines. Now, I would like to propose the following scenario. Let us suppose that one machine is compromised. Given that any lymphocyte process is able to obtain all information about any machine on the network, the user on the compromised machine would be able to get this information. This would lead to the entire network being compromised.

- **Protecting a network of mutually trusting disposable computers** – This architecture assumes that a set of machines will act as lymphocytes only. If needed one of the lymphocyte machines would even sacrifice itself in order to protect a machine that is being threatened. The same problem as with the previous scenario arises. What if one of the lymphocyte machines is broken. The reader has been introduced to the concept of holes (non detectable non-self strings). If the attacker is able to find one of these, compromising one of the lymphocytes would mean compromising all the lymphocytes, therefore the entire network. However, as stated by the counting the holes method, different matching techniques can be used on different lymphocytes to cover the possible holes of each other.

4.2.1 The Architectures – Some conclusions

There is one particular point which I would like to put some attention on. There will always be ways to break any type of security system, however, these architectures only pose a framework for an additional level of

security. If the existing methods have some holes, then this extra layer would probably cover some of them. By no means would it make the system perfects, however it would make it better.

The choice of architecture is not one that is to be decided upon which one is the best. I can say that the best would be the one that is possible to implement according to the needs of the user and the implementation specifications and requirements.

4.3 Data Splitting & Detector Set Size

I would like to pay some attention to the process of splitting the data stream into, what then are assumed to be, independent strings, the decision for the size of the detector set and the negative selection methods to see to what extent these methods, as created by research, have imitated the adaptive immune system of our bodies. In particular I would like to take a look at the concept of hypermutation. The reader will be able to recall these concepts from chapter one and the role they play in the adaptive immune system.

4.3.1 Splitting the Data

As shown in the analysis portrayed in chapter three, the splitting of the data stream into independent strings creates a loss of information. I think there is a major issue here for which I have not been able to find any articles in any research papers, however I will try and bring forward some thought of my own.

Given a data stream that we have decided to use for the definition of the self set, whatever this may be, (we have used system calls, however other possibilities are present, e.g. Key typing behavior) we split that data stream into strings and assume that there is no dependence between these strings. The reason for this is that we need to create structures that we can use for the monitoring process. This analogizes the protein structures used for the monitoring process in the human immune system. However, as described in the previous chapter, this data stream splitting creates a loss of information.

The immune system bases this recognition process around the recognition of some structures, however even though these protein structures do not contain all possible information needed, and neither do the detectors on each lymphocyte, the DNA inside each lymphocyte cell does. So, analogizing with our example, the DNA contains all possible, acceptable full streams of system call sequences. These, are then split into subsequences portrayed in the detectors.

Our system does not, in any way, hold the entire set of all acceptable self, full sequences. Given this gap between the two I think some analysis should go into this in the future work of this research field.

4.3.2 Detector Set Size & Holes

As in the human immune system, the methods described in the previous chapter, do not maintain a full detector set that is capable of detecting all possible non-self strings. Even if a full set was maintained, the existence of holes, as previously shown, would still make the coverage of all non-self strings impossible. However, when accepting a failure probability, and given that we decide upon the size of the detector set, do we then maintain this set constant, or is this set renewed from time to time. The current research papers do not state anything about this fact, so I would like to make some points from my own thought.

If a set of detectors is constant, then it is more vulnerable. An attacker, assuming he has the time and possibility to carry out numerous attacks, would be able to gain an idea upon the detector set after a while if this set stays constant. Also it has to be remembered that if the attacker is able to execute the protection process, he will be able to read the binary. Therefore he will be able to eventually create the detector set and get a clear idea of what is detectable or not. Therefore, it would be a wise decision to make the detector set dynamically change through time.

4.3.3 Hypermutation

The concepts of hyper mutation and immunological memory are two concepts that I have not been able to find implemented in the current methods researched in the previous chapter. To remind the reader, the process of hyper mutation is the process that the detectors of cloned lymphocytes undergo when a bind has been established with an antigenic structure. This increases their specificity to that pathogenic structure to make it easier for them to recognize future occurrences of that or a similar kind of antigen.

The reader should at this point recall the linear time algorithm and the way that it enumerates the all possible matches in the self set in order to create detectors that will not bind to the self strings.
A similar method can be created for the purpose of hypermutation. When a detector is activated and binds to

a non-self strings, that detector should undergo a hypermutation process to generate clones that will bind to non-self strings that could be similar to the one detected.

So, suppose the detected non self strings is:

111**01010**100

where the underlined bits are the bits that caused the detection through the matching rule (in this case r-contiguous bits), a windowing process can be used to enumerate all the possible ways in which this string can be matched. This is by sliding a window of size r (where r is the parameter of the matching rule) over it and moving it one position at a time. This would create the following (in our case r = 5):

11101******
*11010*****
10101** ..... and so on

Detectors could be created that contain these substrings. Of course, the cloned detectors that have been created through this hypermutation must undergo the negative selection process too in order to make sure they are tolerant to the self strings and will not detect any of them.

4.4 Dual Authentication

I would ask the reader at this point to recall the dual authentication method of the immune system when a possible antigenic structure is found. The B-cells recognize the non-self structure, however, they still need the authorization of TH-cells. These latter ones make sure that the bind is legitimate and this prevents autoimmune responses.

This is a concept that has been entirely over looked in the methods presented in the previous chapter. Yes, of course the detectors are made tolerant to the self strings through negative selection, however the actual dual authentication does not happen anywhere.

Let us consider the following scenario. A machine has undergone the training process through use by one

person only. This has made it possible to create a self set and the adequate detectors through any chosen negative selection algorithm. A second person sits in front of the machine and starts using it with totally different behavioral patterns. The actions of this person will be legitimate, however there is a good chance that some of them may be detected as non-self. Having a dual authentication in this case, by making use of the detector set for detecting possible intrusions and also by using the self set with an appropriate matching rule for authorization might solve the problem to some extent.

So, when a detector is activated, instead of immediately rejecting the string, a secondary comparison with the chosen matching rule is made against the self set. If the string does not appear in the self set either it is rejected. This process however will increase the time and space complexity for the detection process as now two databases have to be consulted instead of just one.

4.5 General Conclusions and Opinions

I would like to present the reader at this point with a set of conclusions and some personal opinions that I have gained during the course of this study.

I have to agree that the study of the human immune system and the analogies made from it into methods for the defense of computer systems from malicious attacks can be a largely beneficial discipline. A set of good knowledge has been gained by myself through conducting this study and the research needed for it. I would like however, to express my own opinion concerning the possibilities of applying such concepts in real life situation.

The inspiration of all these methods comes from the human immune system and its cellular based defense mechanisms. There is one major point to be made here. The immune system is based around the principle that all cell are redundant. If one or a few get destroyed it is not a problem. They can be regenerated. This is not the case with most computer systems. Our computer systems are not redundant. If a computer in a network gets compromised, the user will not be able to just relocate to a different one immediately, as precious data is lost. Because of the above, I do not think that there is a real life future for the methods covered in this study.

However, there is in fact one real life example where such a system would have a possibility of success I think. We are all familiar with computer farms. A large set of computers linked together for the purpose of

sharing the computational expense of one or more processes. In this case it can be considered that they are redundant. In fact, if one of them is compromised, then the only damage is that the same amount of computation must be shared between a smaller number of machines. Given this case, such a system could be applied in this case.

Bibliography

[1] – "Immunology Basics" Prof Genc Sylcebe and Ass. Prof. Arben Hoxha – Base text for the faculty of medicine Tirana. - 2001.

[2] – "The Immune System" - www.library.thinkquest.com.

[3] – " Immunology Overview" - Armand S. Goldman and Bellur S Prabhakar.

[4] – "MHC Complex" - www.keratin.com/am/am022.html.

[5] – "An Immunological Approach to Change Detection: Algorithms, Analysis and Implications" Patrik D'haeseleer, Stephanie Forrest, Paul Helman.

[6] – "A Distributes Approach to Anomaly Detecton" Patrik D'haeseleer, Stephanie Forrest, Paul Helman.

[7] – "A Sense of Self for UNIX Processes" Stephanie Forrest , Steven A. Hoffmeyr and Anil Somayaji – In proceeding of the 1996 IEEE symposium on security and privacy.

[8] – "Classification and Detection of Computer Intrusions" S.Kumar – PhD Thesis – Department of Computer Science, Purdue University – August 1995.

[9] – "Automated Detection of Vulnerabilities in Privileged Programs by Execution Monitoring" C. Ko, G. Fink and K. Levitt – In proceedings of the 10-th annual Computer Security Applications Conference – December 1994.

[10] – "An Immunological Model of Distributed Detection and its Application to Computer Security" Steven Andrew Hofmeyr – May 1999.

[11] – "Implementation of a Computer Immune System for Intrusion and Virus Detection" Markus Christoph Unterleitner – February 2006.

[12] – "Cellular and Molecular Immunology" Abul K. Abbas, Andrew H. Lichtman and Jordan S. Pober.

[13] – "Self/NonSelf Discrimination in a Computer" Stephanie Forrest, Alan S. Perelson, Lawrence Allen. and Rajesh Cherkuri – In Proceedings of 1994 IEEE Symposium on Research in Security and Privacy.

[14] – "Computer Immunology" Steven A. Hofmeyr, Stephanie Forrest and Anil Somayaji.

[15] – "Biologically Inspired Immune System for Computers" Jeffrey O. Kephart – Fourth International Workshop on the Sysnthesis and Simulation of Living Systems.

[16] – "Comparing Immunological and Rule Based Intrusion Detection Methods" John Hall.

[17] – "Principles of a Computer Immune System" Anil Somayaji, Steven Hofmeyr and Stephanie Forrest – 1997 New Security Paradigms Workshop, Langdale, Cumbria.

[18] – "Computer Security" a.k.a. "The Yellow Book" Jason Crampton – 2006.

[19] – "The Protection of Information in Computer Systems" J. Saltzer and M. Shroeder – In Proceedings of the IEEE – 1975.

[20] – "A Principle of Kernel Design" G. J. Popek – In Proceedings of the 1974 NCC AFIPS Conference, Volume 43 – 1974.

[21] – "Intrusion Detection Using Sequences of System Calls" Steven A. Hofmeyr, Stephanie Forrest, Anil Somayaji – December 1997.

[22] – "Oxford Advanced Learners Dictionary of Current English" - AS Hornby – New Edition – 1974.

[23] – "Kerberos: An Authentication Service for Computer Networks" B. C. Newman and T. Ts'o – IEEE Communications Magazine – 1994.

[24] – "Comparing Immunological and Rule Based Intrusion Detection Methods" John Hall.

[25] – "The Free On-Line Dictionary" - www.thefreedictionary.com.

[26] – "NIST Official Website" - www.nist.org.

[27] – "Further Efficient Algorithms for Generating Antibody Strings" P. D'haeseleer – Technical Report CS95-3 – 1995.