Contents

Preface

The most terrible threats to the security of computers and networking systems are just the so-called computer virus and unknown intrusion. The rapid development of evasion techniques used in viruses invalidate the famous signature based computer virus detection techniques, so a number of novel virus detection approaches have been proposed continuously to cope with the vital security issue. Because the natural similarities between the biological immune system (BIS) and computer security system, the artificial immune system (AIS) has been developed as a new field in the community of anti-virus researches. The various principles and mechanisms in BIS provide us hard-to-get opportunities to build novel computer virus detection models with abilities of robustness and adaptiveness in detecting the known and unknown viruses.

Biological immune system (BIS) is a hierarchical natural system featured high distribution, parallelization and being able to process complex information, etc. It is also a dynamically adjusting system which is characterized by the abilities of learning, memory, recognition and cognition, such that the biological immune system is good at recognizing and removing antigens effectively for the purpose of protection of the organism. The BIS makes full use of various intelligent ways to react to antigen's intrusions, producing accurate immune responses by means of intrinsic and adaptive immune abilities. Through mutation, evolution and learning to adapt new environments, along with memory mechanisms, BIS can react much stronger and faster against their met foreign antigens and their likes. The BIS is consisted of intrinsic immune (i.e., non-specific immune) and adaptive immune (i.e., specific immune) which are mutually cooperated to defense foreign antigens together.

Artificial Immune System (AIS) is an adaptive systems inspired by theoretical immunology and observed immune functions, principles and models, which is applied for problem solving. In another word, the AIS is a computational system inspired by the BIS, sometime also referred to as the second brain, and becomes one of computational intelligence paradigms. The AIS is a dynamic, adaptive, robust and distributed learning system. As it has the ability of fault tolerant and noise resistant, it is very suitable for the applications in time-varying unknown environment. Currently, the AIS has been applied to many complex problem fields, such as optimization, pattern recognition, fault and anomaly diagnosis, network intrusion detection, and virus

detection, etc.

Generally speaking, the AIS could be roughly classified into two major categories: population based and network based algorithms. Network based algorithms make use of the concepts of immune network theory, while population based algorithms use the theories and models like negative selection principle, clonal principle, danger theory, etc. During the past decades, there are a large number of immune theories and models, such as "self and non-self" model, clonal selection algorithm, immune network, dendritic cell algorithms, danger theory and so on. By mimicking BIS's mechanisms and functions, AIS is developed and now widely used in anomaly detection, fault detection, pattern recognition, optimization, learning, and so on. Like its biological counterpart, AIS is also characterized by the abilities of noise-tolerance, unsupervised learning, self-organization, memorizing, recognition, etc.

In particular, the anomaly detection techniques are to decide whether an unknown test sample is produced by the underlying probability distribution that corresponds to the training set of of normal examples. The pioneering work of Forrest et al. led to a great deal of research and proposals of immune inspired anomaly detection systems. For example, as for the self and nonself model proposed by Forrest et al., the central challenges with anomaly detection is determining the difference between normal and potentially harmful activity. Usually, only self (normal) class is available for training the system regardless of nonself (anomaly) class. Thus the essence of the anomaly detection task is that the training set contains instances only from the self class, while the test set contains instances of both self and nonself classes. Specifically, computer security and virus detection should be regarded as the typical examples of anomaly detection in artificial immune system whose task of protecting computers from viruses, unauthorized users, etc. In computer security, AIS have a very strong capability of anomaly detection for defending unknown viruses and intrusions. Besides, the adaptability is also a very important necessary feature for AIS to learn unknown viruses and intrusions as well as quickly reacting to the learned ones. Other features of AIS like distributability, autonomy, diversity and disposability are also required for the flexibility and stability of AIS.

Therefore, the features of the BIS are just what a computer security system needs, meanwhile the functions of BIS and computer security system are similar each other to some extent. Therefore, the biological immune principles provide effective solutions to computer security issues. The research and development of AIS-based computer security detection are receiving extensive attention increasingly. The application of immune principles and mechanisms can better protect the computer and improve the network environment greatly.

In recent years, computer and networking technologies have developed rapidly and been used more and more widely in our daily life. At the same time, computer security issues appear frequently. The large varieties of malwares, especially new variants and unknown ones, always threaten computers seriously. What is the worst is that malwares are getting more complicated and delicate, with faster speed and greater damage. Meanwhile, huge number of spam not only occupy storage and network bandwidth, but also waste users' time to handle them, resulting in a great loss of productivity. Although many classic solutions have been proposed, there are still lots of

limitations in dealing with the real-world computer security issues.

It is well-known that a computer virus is referred to as a program or a piece of codes that can infect other programs by modifying them to include a possibly evolved copy of it. Broadly, one can regard the computer virus as the malicious code designed to harm or secretly access a computer system without the owners' informed consent, such as viruses, worms, backdoors, Trojans, harmful Apps, hacker codes, etc. In one word, all the programs that are not authorized by users and perform harmful operations in the background are referred to as viruses, which is characterized by several salient features including infectivity, destruction, concealment, latency and triggering, etc.

Since the appearance of computer viruses, they are evolved with the computer technologies and systems all the time. Generally speaking, the development of the viruses mainly went through several typical phases, including DOS boot phase, DOS executable phase, virus generator phase, macro virus phase, as well as virus techniques merging with hacker techniques. As the development of the computer viruses, they has become the main urgent threat to the security of computers and Internet.

As for computers and Internet paradigms, the fighting between viruses and anti-virus techniques will be an endless warfare. In one hand, computer viruses disguise themselves as possible as they could by means of various kinds of evasion techniques including metamorphic and polymorphous techniques, packer and encryption techniques, to name a few. On the other hand, to confront these critical situations, anti-virus techniques have to unpack the suspicious programs, decrypt them and try to be robust to those evasion techniques. On a contrary, the viruses are also trying to evolve to anti-unpack, anti-decrypt and develop into obfuscate the anti-virus techniques. The fighting between viruses and anti-virus techniques is very serious and will last forever.

Nowadays, varieties of novel viruses' techniques are continuously emergent and have a priority of leading the anti-virus techniques one-step ahead. A good anti-virus technique should have to increase the difficulty of viruses' intrusion, decrease the losses caused by the viruses, and react to outbreak of viruses as quickly as possible.

A lot of host-based anti-virus solutions have been proposed by many researchers and companies, which could be roughly classified into three categories, i.e., static techniques, dynamic techniques and heuristics.

Static techniques usually work on bit strings, assembly codes, and application programming interface (API) calls of a program without running the program. One of the most famous static techniques is the signature based virus detection technique, in which a signature usually is a bit string divided from a virus sample and can identify the virus uniquely.

Dynamic techniques keep watching over the execution of every program in real time and observe the behaviors of the program. The dynamic techniques usually utilize the operating system's API sequences, system calls and other kinds of behavior characteristics to identify the purpose of a program.

Heuristics approaches make full use of various heuristic knowledge and information in the program and its environments, by using intelligent computing techniques such as machine learning, data mining, evolutionary computing, AIS, etc., for de-

tecting viruses, which not only can fight the known viruses efficiently, but also can detect new variants and unseen viruses.

Because classic detection approaches of computer viruses are not able to efficiently detect new variants of viruses and unseen viruses, it is urgent to study novel virus detection approaches in depth. As for this point, the immune principle based computer virus detection approaches have being becoming a priority choice in the community of the anti-virus researches as it is characterized of the strong detection capability for new variants of viruses and unseen viruses. The immune based computer virus detection approaches are able to detect new variants and unseen viruses at low false positive rates with limited overheads. These approaches have developed into a new field for computer virus detection and attracted more and more researchers and practitioners.

As we know, the computer virus is called after biological virus because of their similarities, such as parasitism, propagation, infection, hidden and destruction. The BIS has protected body from antigens from the very beginning of life successfully, resolving the problem of defeating unseen antigens. The computer security system has the similar functions with the BIS. Furthermore, the features of the AIS, such as dynamic, adaptive, robust, are also needed in the computer anti-virus system. Applying immune principles to detect virus enables us to recognize new variants and unseen viruses by using existing knowledge. The immune principle based virus detection approaches would own many finer features, such as dynamic, adaptive and robust. It is considered to be able to make up the faults of the signature based virus detection techniques. The immune based computer virus detection approaches have paved a new way for the anti-virus research in the past decades.

Although a number of virus detection models based on immune principle has been achieved great success, in particular, in detecting new variants and unseen viruses under unknown environments, but there exist a few of drawbacks in AIS-based virus detection, such as, a lack of rigorous theoretical analysis, very simple simulations between the AIS and the BIS. Therefore, there is still a long way to go for us to apply the immune based virus detection approaches to the real-world computer security systems.

The objective of this book is to present our proposed major theories and models as well as their applications in malware detection in recent years, for academia, researchers and engineering practitioners who are involved or interested in the study, use, design and development of artificial immune systems (AIS) and AIS-based solutions to computer security issues. Furthermore, I want to provide a single record of our achievements in computer security based on immune principles here.

This book is designed for a professional audience who wishes to learn about the state of the art artificial immune systems and AIS-based malware detection approaches. More specifically, the book offers a theoretical perspective and practical solutions to researchers, practitioners and graduates who are working in the areas of artificial immune system based computer security.

The organization of this book is arranged in a manner from simple to complex. In order to understand the contents of this book comprehensively, the readers should have some fundamentals of computer architecture and software, computer virus, arti-

ficial intelligence, computational intelligence, pattern recognition and machine learning.

I hope this book can shape the research of AIS-based malware detection appropriately and can give the state of art AIS-based malware detection methods and algorithms for the interested readers who might find many algorithms in the book which are directly helpful for their projects in hand, furthermore, some algorithms can also be viewed as a starting point for some active researchers to work with.

In addition, the author presents many newly proposed malware detection methods in didactic approach with detailed materials and shows their excellent performance by a number of experiments and comparisons with the state of the art malware detection techniques. Furthermore, a collection of references, resources and source codes is listed in some webpages which are available freely at http://www.cil.pku.edu.cn/research/anti-malware/index.html, http://www.cil.pku.edu.cn/resources/ and http://www.cil.pku.edu.cn/publications/.

Specifically, this monograph is organized into 11 chapters for easy seizure, which will be briefly described below one by one.

In Chapter 1, artificial immune system (AIS) is mainly presented after a brief introduction of biological immune system (BIS), in particular, several typical AIS algorithms are presented in detail, then followed by features and applications of AIS.

In Chapter 2, introductions to malware and its detection methods are described in detail. As the malware has become a challenge to the security of the computer system, a number of detecting approaches have been proposed to cope with the situations, which are mainly classified into three categories: static techniques, dynamic techniques and heuristics. In this chapter, the classic malware detection approaches and immune based malware detection approaches are briefly introduced after the background knowledge of malware is given. The immune-based malware detection approaches have paved a new way for anti-malware research.

Because the detection of unknown malware is one of most important tasks in Computer Immune System (CIS) studies, by using non-self detection technique, the diversity of anti-body (Ab) and neural networks (NN), an NN-based malware detection algorithm is proposed in Chapter 3. A number of experiments are conducted to illustrate that this algorithm has a high detection rate with a very low false-positive rate.

In Chapter 4, by using the negative selection principle in BIS, a novel generating algorithm of detector, i.e., multiple-point bit mutation method, is proposed, which utilizes random multiple-point mutation to look for non-self detectors in a large range in the whole space of detectors, such that we can obtain a required detector set in a reasonable computational time.

A virus detection system (VDS) based on AIS is proposed in Chapter 5. The VDS at first generates the detector set from virus files in the dataset, negative selection and clonal selection are applied to the detector set to eliminate auto-immunity detectors and increase the diversity of the detector set in the non-self space, respectively. Two novel hybrid distances called hamming-max and shift r-bit continuous distance are proposed to calculate the affinity vectors of each file using the detector set. The VDS compares the detection rates using three classifiers, k-nearest neighbor (KNN), RBF

networks and SVM when the length of detectors is 32-bit and 64-bit, respectively. The experimental results show that the proposed VDS has a strong detection ability and good generalization performance.

As viruses become more complex, existing anti-virus methods are inefficient to detect various forms of viruses, especially new variants and unknown viruses. Inspired by immune system, a hierarchical artificial immune system (AIS) model, which is based on matching in three layers, is proposed to detect a variety of forms of viruses in Chapter 6. Experimental results demonstrate that the proposed model can recognize obfuscated viruses efficiently with an averaged recognition rate of 94%, including new variants of viruses and unknown viruses.

In Chapter 7, a malware detection model based on the negative selection algorithm with penalty factor was proposed to overcome the drawbacks of traditional negative selection algorithms (NSA) in defining the harmfulness of "self" and "nonself". Unlike danger theory, the proposed model is able to detect malware through dangerous signatures extracted from programs. Instead of deleting "nonself" that matches "self", the negative selection algorithm with penalty factor (NSAPF) penalizes the "nonself" using penalty factor C and keeps these items in a library. In this way, the effectiveness of the proposed model is improved by the dangerous signatures that would have been discarded in the traditional NSA.

A danger feature based negative selection algorithm (NFNSA) is presented in Chapter 8, which divides the danger feature space into four parts, and reserves the information of danger features to the utmost extent, for measuring the danger of a sample efficiently. Comprehensive experimental results suggest that the DFNSA is able to reserve as much information of danger features as possible, and the DFNSA malware detection model is effective to detect unseen malware by measuring the danger of a sample precisely.

In Chapter 9, the immune concentration is used to detect malwares. The local concentration based malware detection method connects a certain number of two-element local concentration vectors as feature vector. To achieve better detection performance, particle swarm optimization (PSO) is used to optimize the parameters of local concentration. Then the hybrid concentration based feature extraction (HCFE) approach is presented by extracting the hybrid concentration (HC) of malware in both global and local resolutions.

In Chapter 10, inspired from the immune cooperation (IC) mechanism in BIS, an IC mechanism based learning (ICL) framework is proposed. In this framework, a sample can be expressed as an antigen-specific feature vector and an antigen-nonspecific feature vector at first, respectively, simulating the antigenic determinant and danger features in BIS. The antigen-specific and antigen-nonspecific classifiers score the two vectors and export real-valued Signal 1 and Signal 2, respectively. In collaboration with the two signals, the sample can be classified by the cooperation classifier, which resolves the signal conflict problem at the same time. The ICL framework simulates the BIS in the view of immune signals and takes full advantage of the cooperation effect of the immune signals, which improves the performance of the ICL framework dramatically.

Chapter 11 presents a new statistics named class-wise information gain (CIG).

Different from information gain (IG) which only selects global features for a classification problem, the CIG is able to select the features with the highest information content for a specific class in a problem. On the basis of the CIG, a novel CIG-based malware detection method is proposed to efficiently detect malware loaders and infected executables in the wild.

After that, several appendices are attached by giving the lists of figures, tables and symbols appeared in this book. Finally, the indices of keywords are drawn out at the end of this monograph.

Due to the limited specialty knowledge and capability of mine, a few of errors, typos and inadequacy must have in the book, the critical comments and valuable suggestions are warmly welcome to ytan(AT)pku.edu.cn.

Ying Tan
Beijing, China
March 30, 2015

About Author

Professor Ying TAN

Dr. Ying Tan is a full professor and PhD advisor at School of Electronics Engineering and Computer Science of Peking University, and director of Computational Intelligence Laboratory at Peking University (CIL@PKU: http://www.cil.pku.edu.cn). He received his BEng from Electronic Engineering Institute, MSc from Xidian University, and PhD from Southeast University, in 1985, 1988, and 1997, respectively.

His research interests include computational intelligence, swarm intelligence, data mining, machine learning, pattern recognition, intelligent information processing for information security, fireworks algorithm, etc. He has published more than 280 papers, and authored/co-authored 6 books and 10+ chapters in book, and received 3 invention patents.

He serves as the Editor-in-Chief of International Journal of Computational Intelligence and Pattern Recognition (IJCIPR), an Associate Editor of IEEE Transactions on Cybernetics (Cyb), IEEE Transactions on Neural Networks and Learning Systems (TNNLS), etc. He also served as an Editor of Springer's Lecture Notes on Computer Science (LNCS) for 10+ volumes, and Guest Editors of several referred Journals, including Information Science, Softcomputing, Neurocomputing, IEEE/ACM Transactions on Computational Biology and Bioinformatics (IEEE/ACM TCBB), Natural Computing, etc. He is the general chair of ICSI-CCI 2015 joint conference, and was the founding general chair of the series International Conference on Swarm Intelligence (ICSI 2010-2014), program committee co-chair of IEEE WCCI'2014, etc. He is a senior member of IEEE.

1
Artificial Immune System

1.1
Introduction

People have a keen interest on the biosphere since ancient times and have gotten inspiration from the structures and functions of biological systems and their regulatory mechanisms continuously. Since mid-20th century, researchers have focused on the simulation of the biological systems, especially the structures and functions of human beings. For examples, artificial neural network is to simulate the structure of the nerve system of human brain, fuzzy control is very similar to the fuzzy thinking and inaccurate reasoning of human beings, and evolutionary computation algorithms are the direct simulations of the evolved processes of natural creatures.

In recent years, biological immune system has become an emerging bio-informatics research area. The immune system is a complex system consisting of organs, cells and molecules. The immune system is able to recognize the stimulation of "self" and "non-self", make a precise response, and retain the memory. It turns out from many researches that the immune system is of a variety of functions such as pattern recognition, learning, memory acquisition, diversity, fault-tolerant, distributed detection and so on.

These attractive properties of the biological immune system have drawn extensive attention of engineering researchers who have proposed many novel algorithms and techniques based on those principles of immunology. After introducing the concept of immunity, many researches in engineering have obtained more and more promising results, such as computer network security, intelligent robots, intelligent control and pattern recognition and fault diagnosis. These researches and applications not only can help us to further understand the immune system itself, but also to reexamine and solve practical engineering problems from the perspective of information processing way in biological immune system.

Building a computer security system in principle of the immune system opens a new research field of information security. Many structure, functions and mechanisms of the immune system are very helpful and referential to the research of computer security, such as antibody diversity, dynamic coverage and distribution. We believe that the excellent features of the immune system are the roots and original

2
Malware Detection

Malware has become a challenge to the security of the computer system. The rapid development of evasion techniques makes the signature based malware detection techniques lose effectiveness. Many approaches have been proposed to cope with the situations, which are mainly classified into three categories: static techniques, dynamic techniques and heuristics. Artificial immune system (AIS), because of the natural similarities between the biological immune system and computer security system, has been developed into a new filed for anti-malware research, attracting many researchers. The immune mechanisms provide opportunities to construct malware detection models that are robust and adaptive with the ability to detect unseen malware. In this chapter, the classic malware detection approaches and immune based malware detection approaches are briefly introduced after the background knowledge of malware is presented. The malware detection approaches based on immune principles have paved a new way for anti-malware research.

2.1
Introduction

With the rapid development of computer technology and Internet, the computer has been a part of daily life. Meanwhile, the computer security garners more and more attention. Malwares, the new variations and unknown malwares in particular, have become the biggest threats to the computer systems. Nowadays the malwares are becoming more complex with faster breed speeds and stronger abilities for latency, destruction, and infection. Now a malware is able to spread over the globe in several minutes and results in huge economic losses. How to protect computers from various kinds of malwares has become one of the most urgent missions.

Many companies have released anti-malware software, most of which is based on signatures. The software detects known malwares very quickly with lower false positive rates and overheads. Unfortunately, the software fails to detect new variations and unknown malwares. Based on metamorphic and polymorphous techniques, even a layman can develop new variations of known malwares easily using virus automatons. For example, Agobot has been observed to have more than 580 variations since

its initial release, using polymorphism to evade detection and disassembly [1]. Thus, traditional malware detection approaches based on signatures are no longer fit for the new environments; as well, dynamic techniques and heuristics have started to emerge.

Dynamic techniques, such as virtual machine, mostly monitor the behaviors of a program with the help of application programming interface (API) call sequences generated at runtime. However, because of the huge overheads of monitoring API calls, it is very hard to deploy the dynamic techniques on personal computers.

Data mining approaches, one of the most popular heuristics, try to mine frequent patterns or association rules to detect malwares using classic classifiers . These have led to some success. However, data mining loses the semantic information of the code and cannot easily recognize unknown malwares.

As we know, malware is similar to biological virus in many aspects, such as parasitism, breed, and infection. In nature, the biological immune system (BIS) protects body from antigens, resolving the problem of unknown antigens [2], so applying immune mechanisms to anti-malware has developed into a new field for the past few years, attracting many researchers. Forrest applied the immune theory to computer anomaly detection for the first time in 1994 [3]. Since then, many researchers have proposed various kinds of malware detection models and achieved some success; most of them are mainly derived from ARTIS [4, 5, 6].

With the time going on, more and more immune mechanisms become clear. Immune based malware detection approaches make use of more immune theories and the study deepens continuously. The simulations to BIS keep going ahead. Now the malware detection objects have included raw bit strings, process calls, and process call arguments.

2.2 Malware

2.2.1 Definition and Features

In a broad sense, malware includes viruses , worms , backdoors , Trojans , and so on [7]. With the development of malware, the lines between different types of malwares are not clear. Now all the software that is not authorized by users and performs harmful operations in the background is referred to as a malware [8, 9, 10, 11].

The features of malware are given as below.

- Infectivity: Infectivity is the fundamental and essential feature of malware, which is the foundation to classify a malware. When a malware intrudes in a computer system, it starts to scan the programs and computers on the Internet that can be infected. Then through self-duplicating, it spreads to the other programs and computers.
- Destruction: Based on the extent of destruction, malware is divided into "benign"

malware and malignant malware. "Benign" malware merely occupies system resources, such as GENP, W-BOOT. Nevertheless, malignant malware usually has clear purposes. They can destroy data, delete files, and format disks.

- Concealment: Malwares often attach themselves to benign programs and start up with the host programs. They perform harmful operations in the background hiding from users.
- Latency: After intruding in a computer system, malwares hide themselves from users instead of attacking the system immediately. This feature makes malwares have longer lives. They spread themselves and infect other programs in this period.
- Trigger: Most malwares have one or more trigger conditions. When these conditions are satisfied, the malwares begin to destroy the system.

Other features of the malwares include illegality, expressiveness, and unpredictability.

2.2.2 The Development Phases of Malware

The malwares are evolved with the computer technology all the time. The development of the malwares approximately goes through several phases as below.

- DOS boot phase: Figures 2.1 and 2.2 illustrate the boot procedures of DOS without and with boot sector virus, respectively. Before the system obtains right of control, the malware starts up, modifies interrupt vector and copy itself to infect the disk. These are the original infection procedures of the malwares. Furthermore, the similar infection procedures can be found in the malwares nowadays.

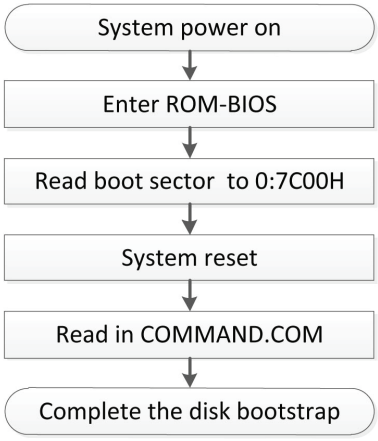

Figure 2.1 Normal boot procedure of DOS.

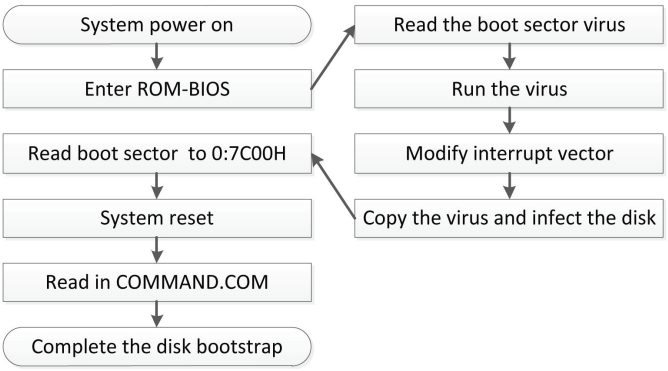

Figure 2.2 Boot procedure of DOS with boot sector virus.

- DOS executable phase: In this phase, the malwares exist in a computer system in the term of executable files. They control the system when users run applications infected by the malwares. Most malwares now are executable files.
- Malware generator phase: Malware generators, called malware automatons, can generate new variations of known malwares with different signatures. Metamorphic techniques are used here to obfuscate scanners based on signatures, including instruction reordering, register renaming, code expansion, code shrinking and garbage code insertion [12].
- Macro malware phase: Before the emerging of macro Malwares, all the malwares merely infect executable files because this almost is the only way for the malwares to obtain the right of execution. When users run a host of a malware, the malware starts up and controls the system. Infecting data files cannot help the malware to run itself. The emerging of macro malwares changed this situation and their punching bags are data files, mainly Microsoft Office files.
- Malware techniques merging with hacker techniques: Nowadays merging of malware techniques and hacker techniques has been a tendency. It makes the malwares have much stronger concealment, latency and much faster breed speed than ever before.

2.3 Classic Malware Detection Approaches

Malware has become a major threat to the security of the computer and Internet. A wide range of host-based solutions have been proposed by many researchers and companies [13-37]. These techniques are broadly classified into three types: static techniques, dynamic techniques, and heuristics.[13, 9, 14, 10, 15, 11, 16, 17, 18, 19, 19, 20, 21, 22]

The fight between the malwares and the anti-malware techniques is more violent

now than ever before. The malwares disguise themselves using various evasion techniques such as metamorphic and polymorphous techniques, packer, and encryption techniques. Coping with the new environments, the anti-malware techniques unpack the suspicious programs, decrypt them and try to be robust to those evasion techniques. Nevertheless, the malwares evolve to anti-unpack, anti-decrypt and develop into obfuscate the anti-malware techniques again. The fight will never stop and the malware techniques will always be ahead of the anti-malware techniques. What we can do is to increase the difficulty of intrusion, decrease the losses caused by the malwares and react to them as quickly as possible.

2.3.1 Static Techniques

Static techniques mostly operate on program bit strings and disassembled instructions. One of the most famous static techniques is signatures based detection technique.

Signatures based detection technique is the mainstream anti-malware method and most software now is based on signatures. A signature is a bit string splitted from a virus sample and it can identify a virus uniquely. The software based on signatures is referred to as scanner.

In order to extract a signature from a malware, the experts first disassemble the virus sample to assembly code. Then they analyze it in the semantic level to figure out the mechanisms and workflow of the malware. Finally, a signature is extracted out representing the malware sample uniquely.

This technique is able to detect known virus very quickly with lower false positive and high true positive rates. It is the simplest method with minimal overheads. Nevertheless, since a signature of a new malware can be only extracted after the break out of the virus by experts, it takes a long time and the losses caused by the virus cannot recover already. Furthermore, with the development of malware techniques, there are many evasion techniques to help the virus evade from the scanners based on signatures, such as metamorphic and polymorphous techniques, packer, and encryption techniques. Signatures based techniques are easily defeated by these techniques. For example, simple program entry point modifications consisting of two extra jump instructions effectively defeat most scanners based on signatures [23].

To conclude, signatures based techniques are vulnerable to evasion techniques. As a result, dynamic and heuristic approaches are developed to cope with these situations.

2.3.2 Dynamic Techniques

Malware should show some special behaviors when they infect other applications. For example, writing operation to executable files, dangerous operations taking formatting a disk as an example, and switching between malware and its host. These behaviors give us an opportunity to recognize the malwares. Based on the idea above,

dynamic techniques decide whether or not a code is infected by running the code and observing its behaviors. Usually dynamic techniques utilize the operating system's API sequences, system calls and other kinds of behavior characteristics to identify the purpose of a program [24].

There are two main types of dynamic techniques: behavior monitoring approach and virtual machine approach.

Based on the assumption that the malwares have some special behaviors that can identify themselves and never emerge in benign programs, the behavior monitors keep watch on any behavior of malware and wish to prevent destruction from the dangerous operations.

This approach owns the ability to detect known malwares, new variations and unknown malwares. However, let the malwares run in a real machine is very dangerous. If the behavior monitor fails to kill a malware, the malware takes control of the computer. Moreover, the overheads brought by a monitor are huge and unacceptable to ordinary computers. The false positive rate is high inevitably. And the approach cannot recognize the type and name of a malware, thus, cannot eliminate the malware from a computer. Furthermore, it is very hard to implement a relative perfect behavior monitor.

Virtual machine approach creates a virtual machine (VM) and let the programs run in it. The execution environment of a program here is the VM which is software instead of the physical machine, so the computer is safe even the VM is crashed by a malware. It is very easy to collect all the information when a program runs in a VM. If the VM finds some dangerous operations, it would give the users a tip. If it confirms that the running program is a malware, then it kills the malware.

Virtual machine is very safe and can recognize almost all the malwares, including encrypted and packed viruses. Now the VM approach has become one of the most amazing malware detection approaches. Whereas the virtual machine brings comparable overheads to the host computers. How to implement a relative perfect virtual machine is a new field to study. In addition, the virtual machine only simulates parts of the computer's functions and it provides opportunities for anti-virtual machine techniques.

Anti-virtual machine techniques have been used in many malwares recently. For example, inserting some special instructions into a malware may cause the crash of the virtual machine. Entry point obscuring is also involved by the malwares to evade from the virtual machine approach.

[16, 25, 26, 27, 17, 18] proposed some models based on dynamic techniques to detect the malwares Although these techniques have produced promising results, they can produce high rates of false positive errors, an issue which has yet to be resolved [28].

2.3.3 Heuristics

Sulaiman et al proposed a static analysis framework for detecting variations of malwares which was called disassembled code analyzer for malware (DCAM) [29]. Dif-

ferent from traditional static code analysis, the authors extracted signatures from disassembled codes generated by PE explorer instead of raw program bit strings. Each signature was a key/value pair where the key represented the label and the value represented the set of instructions associated with the label. The number of the instructions in a signature had to exceed a threshold in order to contain enough information. The files which got through three steps of matching were considered as benign programs; otherwise DCAM classified the files as malwares. The DCAM worked very well in the experiments taken by the authors and could prevent breakouts of previous identified malwares.

Henchiri and Japkowicz adopted a data mining approach to extract frequent patterns (FPs) for detecting malware [30]. Based on intra-family support and inter-family support, they filtered FPs twice, trying to obtain more general FPs. From the final set of FPs, traditional machine learning algorithms were involved to train the model and make classification, such as ID3, J48. They verified the effectiveness of their model using 5-fold cross validation, showing some good results. Nevertheless, FPs are merely fixed-length bit strings with no definite meaning and cannot represent real signatures of the malwares.

[31] proposed a malware detection model using cosine similarity analysis to detecting obfuscated viruses. Their work was based on the premise that given a variant of a malware, they can detect any obfuscated version of the malware with high probability. Actually, their model was only worked on code transposition. The biggest issue in this model was that how to extract functions within a program cannot be completed in real time.

Ye et al. used associative classification and post-processing techniques for malware detection based on the analysis of API execution sequences called by portable executable (PE) files [14]. First, they extracted the API calls from Windows PE files as the features of the samples and stored them in a signature database. Then they extended a modified FP-Growth algorithm proposed in [10, 15] to generate the association rules. Finally, by adopting post-processing techniques of the associative classification, the authors reduced the number of rules and got a concise classifier. Promising results demonstrated that the efficiency and ability of the model outperformed popular anti-malware software, as well as previous data mining based malware detection systems.

Tabish et al proposed a malware detection model using statistical analysis of the byte-level file content [11]. This model was not based on signatures. It neither memorized specific strings appearing in the file contents nor depended on prior knowledge of file types. As a result, the model was robust to the most commonly used evasion techniques. Although better results could be obtained in this way, there were high false positive rates, because this approach only used statistics from the training set.

Many researchers have proposed various kinds of heuristics to detect the malwares with some success [32, 19, 33, 34, 20, 35, 21, 22, 36].

2.4 Immune Based Malware Detection Approaches

2.4.1 An Overview of Artificial Immune System

2.4.1.1 Artificial Immune System

AIS is a computational system inspired by the biological immune system (BIS), which is referred to as the second brain. AIS as a dynamic, adaptive, robust, distributed learning system have the ability of fault tolerant and noise resistant, and is fit for the applications under various unknown environments.

Algorithm 5 General artificial immune algorithm

1: Input antigens.
2: Initialize antibody population.
3: Calculate the affinities of the antibodies.
4: Lifecycle event and update the antibodies - creation and destruction.
5: If the terminate criteria are satisfied, go to 6; otherwise, go to 3.
6: Output the antibodies.

AIS have been applied to many complex problem domains, such as optimization, pattern recognition, fault and anomaly diagnosis, network intrusion detection, and virus detection.

The steps of the general artificial immune algorithm are shown in Algorithm 5.

There are three typical algorithms in AIS: negative selection algorithm (NSA), clonal selection algorithm and immune network model.

2.4.1.2 Motivations of Applying Immune Mechanisms to Malware Detection

As we know, malware is similar to biological virus in many aspects, such as parasitism, breed, infection, and destruction. The biological immune system (BIS) protected body from antigens from the beginning of life, resolving the problem of unknown antigens [2]. The computer system is designed from the prototype of human beings and the computer security system has the similar functions with BIS. Furthermore, the futures of AIS, such as dynamic, adaptive, robust, are needed in the computer anti-malware system (CAMS). To sum up, applying immune mechanisms to computer security system, especially the CAMS, is reasonable and has developed into a new field for the past few years, attracting many researchers. The relationship of BIS and CAMS is given in Table 2.1.

Applying immune mechanisms to malware detection helps the CAMS recognize new variations and unknown malwares, using existing knowledge. The CAMS with immune mechanisms would be more robust to make up the fault of the signatures based malware detection techniques. The immune based malware detection approaches have paved a new way for anti-malware research[9, 37, 38, 39, 40, 41].

Table 2.1 The relationship of BIS and CAMS

BIS	CAMS
Antigens	Malwares
Antibodies	Detectors for the malwares
Binding of an antigen and an antibody	Pattern matching of the malwares and detectors

2.4.2 An Overview of Artificial Immune System for Malware Detection

With the development of immunology, immune mechanisms have begun to be applied in the field of computer security. Forrest et al. first proposed a negative selection algorithm to detect anomaly modification on protected data in 1994 [3] and later applied it to UNIX process detection [42]. It is the beginning of applying immune theory to the computer security system.

Kephart et al described a blueprint of a computer immune system in [5]. They set forth some criteria that must be met to provide real-world, functional protection from rapidly spreading viruses, including innate immunity, adaptive immunity, delivery and dissemination, high speed, scalability, safety and reliability as well as customer control. In fact, these criteria have become the standards for other computer immune systems from then on.

Based on the clonal selection theory of Burnet, the clone selection algorithm was presented by Kim and Bentley [43].

Matzinger proposed the "danger theory " in 2002 [44]. The danger theory believes the immune system is more concerned with entities that do damage than with those are foreign, which corrects the fault of traditional "self " and "nonself" model in defining of harmfulness of "self " and "nonself".

Since then, more and more researchers have devoted themselves to the study of computer immune systems based on immune mechanisms and various kinds of immune based malware detection models have been proposed [45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 12, 40, 41, 57].

Kenneth et al introduced a new artificial immune system based on REtrovirus ALGOrithm (REALGO) which was inspired by reverse transcription RNA as found in the biological systems [46]. In the learning phase, positive selection generated new antibodies using genetic algorithm based on known malware signatures and negative selection ensured that these antibodies did not trigger on "self". The REALGO provided a memory for each antibody in the genetic algorithm so that an antibody could remember its best situation. Under the help of the memory, the REALGO was able to revert back to the previous generation and mutate in a different "direction" to escape local extremum.

Li Zhou et al presented an immunity based malware detection approach with process call arguments and user feedback [47]. It collected arguments of process calls

instead of the sequences of process, and utilized these arguments to train detectors with real-valued negative selection algorithm [48]. In the phase of test, they adjusted the threshold between benign programs and viruses through user feedback. The detection rate achieved was 0.7, which proved the approach could cope with unknown malwares. However, let users distinguish a virus from normal files and give feedback was very difficult.

Li Tao proposed a dynamic detection model for malwares based on an immune system [50]. Through dynamic evolution of "self", an antibody gene library , and detectors, this model reduced the size of the "self" set, raised the generating efficiency of detectors, and resolved the problem of detector training time being exponential with respect to the size of "self".

Immune based malware detection techniques have the ability to detect new variations and unknown malwres and paved a new way for anti-malware research. These techniques have developed into a new field for malware detection and attracted more and more researchers. However, there is a lack of rigorous theoretical principle of mathematics. In addition, the simulations to BIS are still very simple. Combing with the characteristics of malware detection and the studies of immune algorithms are needed. There is still a long way to go to apply immune based malware detection techniques in the real world.

2.4.3 An Immune Based Virus Detection System Using Affinity Vectors

2.4.3.1 Overview

Aiming at building a light-weighted, limited computer resources and early virus warning system, an immune based virus detection system using affinity vectors (IVDS) was proposed [38]. At first, the IVDS generates a detector set from viruses in the train set, using negative selection and clonal selection. Negative selection eliminates autoimmunity detectors and ensures that any detector in the detector set does not match "self"; clonal selection increases the diversity of the detector set which helps the model obtain a stronger ability to recognize new variations and unknown viruses. Then two novel hybrid distances called hamming-max and shift r bit-continuous distance are presented to calculate the affinity vectors of each file. Finally, based on the affinity vectors, three classic classifiers, SVM, RBF network and k-nearest neighbor (KNN), are involved to verify the performance of the model.

2.4.3.2 Experiments and Analysis

The dataset used here is CILPKU08 dataset [58]. Three test datasets are obtained by randomly dividing CILPKU08 dataset as shown in Table 2.2.

Here, the percentage of training set $= NTS/(NTS+NDS)$. (NTS and NDS denote the number of programs in the training set and test set, respectively) The experimental results are shown in Figure 2.3.

As shown in Figure 2.3, IVDS achieves high accuracy in detecting unknown viruses when the percentage of the training set is 25%. RBF network has better performance than SVM and KNN for the training set and worse accuracy for the test set.

Table 2.2 The test datasets in the experiments

	Training set		Test set		The percentage
Datasets	Benign programs	Viruses	Benign programs	Viruses	of training set
Dataset 1	71	885	213	2662	0.25
Dataset 2	142	1773	142	1773	0.5
Dataset 3	213	2662	71	885	0.75

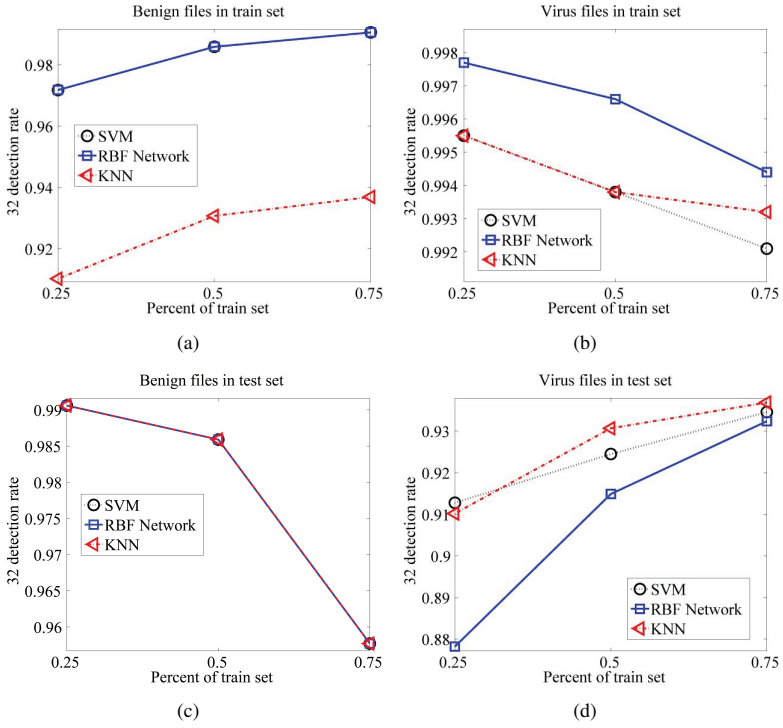

Figure 2.3 The detection accuracy of SVM, RBF network and KNN.

We can conclude from this phenomenon that the RBF network has weaker generalization ability here. Whereas the SVM and KNN have stable performances for the training set and test set with different percentages of the training set.

2.4.4 A Hierarchical Artificial Immune Model for Virus Detection

2.4.4.1 Overview

A hierarchical artificial immune model for virus detection (HAIM) was presented in [37]. The motivation of the HAIM is to make full use of the relativity between viruses' signatures. Generally speaking, a virus usually contains several heuristic signatures and a heuristic signature may appear in various viruses. We believe there is some relativity between these heuristic signatures and combination orderly of some signatures makes up a virus. The HAIM, taking a virus as an unit, detects viruses based on the simple relativity between signatures in a virus sample. The HAIM is composed of two modules: virus gene library generating module and self-nonself classification module. The first module is used to generate the detecting gene library to accomplish the training of given data. The second module is assigned as the detecting phase in terms of the results from the first module for detecting the suspicious programs. The processes of the two modules are given in Figures 2.4 and 2.5.

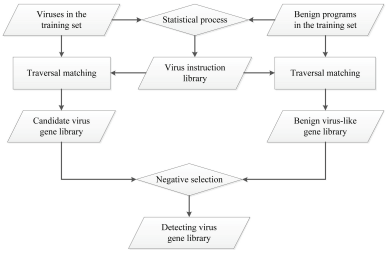

Figure 2.4 Virus gene library generating process.

The virus gene library generating module extracts a virus instruction library based on the statistics from the training set. Here, an instruction is a bit string with 2 bytes. Then a candidate virus gene library and a benign virus-like gene library are obtained by traversing all the viruses and benign programs in the training set using a sliding window, respectively. Finally, according to the negative selection mechanism, the candidate virus library is upgraded as the detecting virus gene library.

In the self-nonself classification module, suspicious virus-like genes are extracted from a suspicious program, and they will be used for the classification. The method to calculate the affinity of a suspicious program is illstrated in Figure 2.6. Getting through the matching processes with three levels, a wise decision is made to classify the program.

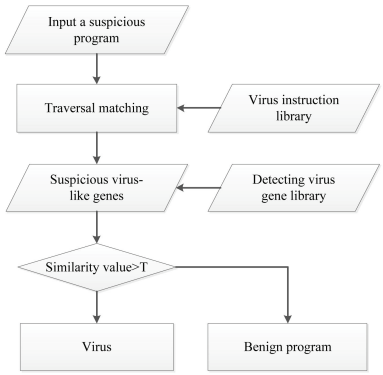

Figure 2.5 Self-nonself classification process.

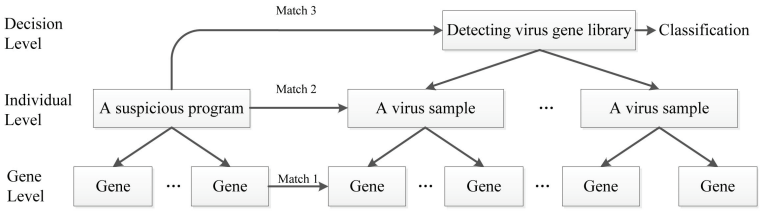

Figure 2.6 The hierarchical matching method.

2.4.4.2 Experiments and Analysis

The experiments are taken on the CILPKU08 dataset [58]. By randomly dividing the dataset into the training set and test set for nine times, nine tests have been down. The experimental results are shown in Table 2.3.

Table 2.3 Experimental results obtained by HAIM

	Overall accuracy	False positive rate	True positive rate
Test1	95.40%	1.80%	92.20%
Test2	96.30%	1.40%	93.70%
Test3	97.00%	1.30%	95.10%
Test4	98.30%	1.10%	97.60%
Test5	97.40%	1.40%	96.10%
Test6	96.90%	1.60%	95.30%
Test7	94.70%	0.70%	90.00%
Test8	93.60%	1.70%	88.90%
Test9	94.50%	1.50%	90.50%

It is easy to ascertain from Table 2.3 that the HAIM is a very stable model with good performance. It achieves high true negative rates for the unseen benign programs in the test sets with all the false positive rates lower than 2%. The average true positive rate achieves 93.27% for unknown viruses in the test sets which is comparable high.

2.4.5 A Malware Detection Model Based on a Negative Selection Algorithm with Penalty Factor

2.4.5.1 Overview

The negative selection algorithm is one of the most important algorithms in artificial immune systems. After deleting detectors that match "self", the NSA obtains a detector set, in which none of the items matches "self", and which is then used to detect virus[9]. A traditional NSA assumes that all "self" is harmless and all "nonself" is harmful. However, in organisms this is not always the case. Taking cancer cells as an example, not all "self" is harmless; and similarly, not all "nonself" is harmful, for example, food. A computer security system, therefore, only has to identify dangerous virus instead of reacting to all "nonself". Take formatting a disk as an example. This operation is dangerous; programs implementing this operation are considerably "dangerous". If a program implementing this operation neither reads any command line parameters nor asks the user to confirm, it could be malware. This type of dangerous signatures provides some useful information. In fact, the operation of formatting a disk can be included in both malware and benign programs. Deleting such dangerous code snippets, as is done by the traditional NSA, destroys useful information, which is obviously a disadvantage for the malware detection model. Theoretically, every program, regardless of whether it is a benign program or malware, can use almost

any of the instructions and functions in a computer system. Moreover, almost all the functions used in malware are also used by specific benign programs, for example, formatting a disk, modifying the registry. If a "perfect" "self" set is given, the traditional NSA would be ineffective due to delete too many detectors. Based on the analysis, a malware detection model based on a negative selection algorithm with penalty factor (MDM-NSAPF) was proposed to overcome the drawback of traditional negative selection algorithms in defining the harmfulness of "self" and "nonself" [13]. The MDM-NSAPF consists of a malware signature extraction module (MSEM) and a suspicious program detection module (SPDM). A flowchart for the MSEM is shown in Figure 2.7.

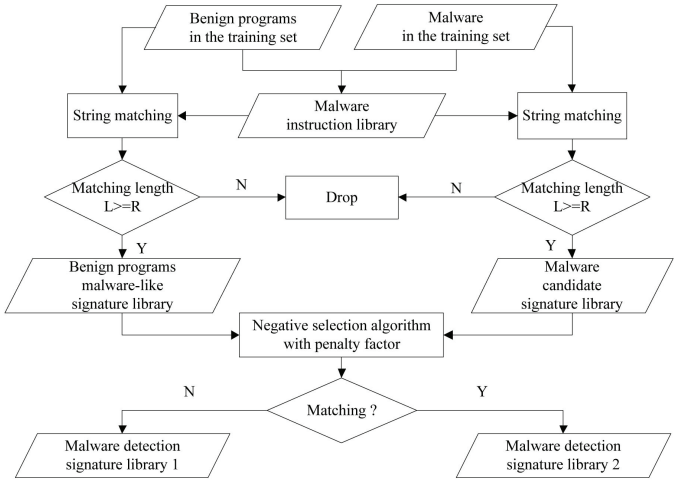

Figure 2.7 Flowchart for MSEM.

In the MSEM, a malware candidate signature library (MCSL) and a benign program malware-like signature library (BPMSL) are extracted, respectively, from the malware and benign programs of the training set after generating the malware instruction library (MIL). Taking the MCSL as "nonself" and the BPMSL as "self", a NSAPF is introduced to extract the malware detection signature library (MDSL) consisting of MDSL1 and MDSL2. Signatures in the MDSL1 are characteristic signatures of "nonself", whereas signatures in the MDSL2 are dangerous ones belonging to both "self" and "nonself", and which should be penalized by penalty factor C after ascertaining, through probabilistic methods, to what extent they represent malware. In the SPDM, signatures of suspicious programs are extracted using the MIL. Then r-contiguous bit matching is computed between the signatures of the suspicious program and the MDSL. If the matching value exceeds the given program classification threshold, we classify the program as malware; otherwise it is considered a benign program.

5.3.2. Experiments and analysis: Experiments in this paper were conducted using

the three datasets: Henchiri dataset, CILPKU08 dataset, and VX Heavens dataset [59]. The results on the Henchiri dataset and CILPKU08 dataset are shown in Figures 2.8 and 2.9.

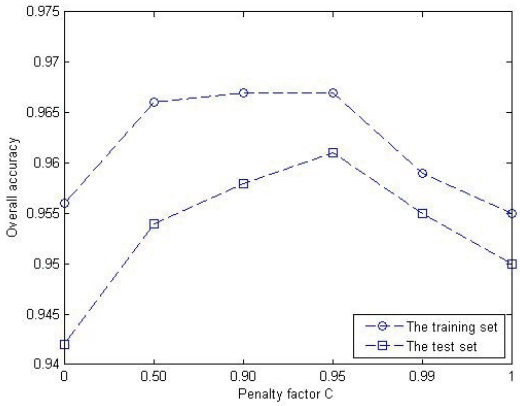

Figure 2.8 Results for Henchiri dataset.

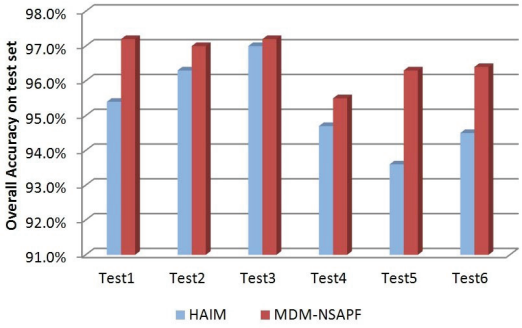

Figure 2.9 Results for CILPKU08 dataset.

From Figure 2.8, the optimal overall accuracy of the MDM-NSAPF achieves 96% on the test set with penalty factor C=0.90. With a decrease in penalty factor C, the penalty to signatures in MDSL2 decreases. As a result, the MDSL2 provide helpful information with more and more false information. The overall accuracy increases at first and drops at last. The results confirm that the MDSL2 plays a positive role to improve the effectiveness of the MDM-NSAPF. As illustrated in Figure 2.9, the overall accuracy of MDM-NSAPF is about 1% to 3% higher than that of HAIM. The area under the receiver operating characteristic curve (AUC) was set as the measure of the effectiveness of MDM-NSAPF on the VX Heavens dataset. The results are

shown in Figure 2.10. Compared with the results of Tabish [11], the optimal AUC of the MDM-NSAPF is on average 0.04 higher.

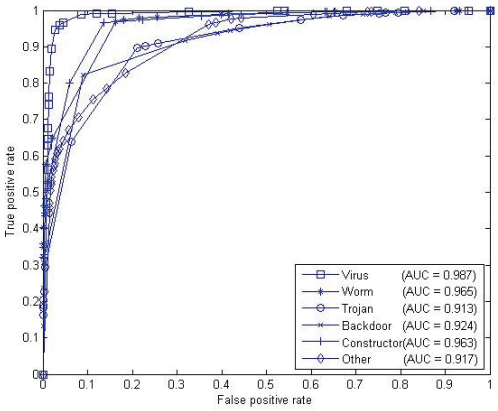

Figure 2.10 Experimental results for VX Heavens dataset.

The MDM-NSAPF focuses on the harmfulness of the code and extracts dangerous signatures, which are included in the MDSL. By adjusting the penalty factor C, the model achieves a tradeoff between the true positive and false positive rates to satisfy the requirements of various users. Comprehensive experimental results confirm that the proposed model is effective in detecting unknown malware with lower false positive rates.

2.5 Summary

The classic malware detection approaches cannot detect new variations and unknown malwares effectively. New malware detection methods are needed urgently. Immune based computer malware detection approaches, because of the ability to detect unseen malwares, have developed into a new field for anti-malware research. Many researchers have proposed lots of malware detection models based on immune mechanisms and achieved some success. However, there is a lack of rigorous theoretical principle of mathematics. The simulations of AIS to BIS are still very simple. There is still a long way to go to apply immune based malware detection approaches in the real world.

References

1 Bailey, M., Oberheide, J., Andersen, J., Mao, Z.M., Jahanian, F., and Nazario, J. (2007) Automated classification and analysis of internet malware, in *Recent Advances in Intrusion Detection*, Springer, pp. 178–197.

2 Perelson, A.S. and Weisbuch, G. (1997) Immunology for physicists. *Reviews of modern physics*, **69** (4), 1219.

3 Forrest, S., Perelson, A., Allen, L., and Cherukuri, R. (1994) Self-nonself discrimination in a computer, in *Research in Security and Privacy, 1994. Proceedings., 1994 IEEE Computer Society Symposium on*, IEEE, pp. 202–212.

4 Kephart, J.O. and Arnold, W.C. (1994) Automatic extraction of computer virus signatures, in *4th virus bulletin international conference*, pp. 178–184.

5 Kephart, J., Sorkin, G., Swimmer, M., and White, S. (1999) *Blueprint for a Computer Immune System**, Springer.

6 Okamoto, T. and Ishida, Y. (2000) A distributed approach against computer viruses inspired by the immune system. *IEICE transactions on communications*, **83** (5), 908–915.

7 Fu J M, Peng G J, Z.H.G. (2009) *Computer virus analysis and confronting*, Wuhan university press, China.

8 Wikipedia. URL `http://en.wikipedia.org/wiki/Malware`.

9 Zhang, P., Wang, W., and Tan, Y. (2010) A malware detection model based on a negative selection algorithm with penalty factor. *Science China Information Sciences*, **53** (12), 2461–2471.

10 Ye, Y., Wang, D., Li, T., and Ye, D. (2007) Imds: Intelligent malware detection system, in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, pp. 1043–1047.

11 Tabish, S.M., Shafiq, M.Z., and Farooq, M. (2009) Malware detection using statistical analysis of byte-level file content, in *Proceedings of the ACM SIGKDD Workshop on CyberSecurity and Intelligence Informatics*, ACM, pp. 23–31.

12 Al Daoud, E. (2009) Metamorphic viruses detection using artificial immune system, in *Communication Software and Networks, 2009. ICCSN'09. International Conference on*, IEEE, pp. 168–172.

13 Zhang, P., Wang, W., and Tan, Y. (2010) A malware detection model based on a negative selection algorithm with penalty factor. *SCIENCE CHINA Information Sciences*, **53** (12), 2461–2471.

14 Ye, Y., Jiang, Q., and Zhuang, W. (2008) Associative classification and post-processing techniques used for malware detection, in *Anti-counterfeiting, Security and Identification, 2008. ASID 2008. 2nd International Conference on*, IEEE, pp. 276–279.

15 Ye, Y., Wang, D., Li, T., Ye, D., and Jiang, Q. (2008) An intelligent pe-malware detection system based on association mining. *Journal in Computer Virology*, **4** (4), 323–334.

16 Christodorescu, M., Jha, S., Seshia, S.A., Song, D., and Bryant, R.E. (2005) Semantics-aware malware detection, in *Security and Privacy, 2005 IEEE Symposium on*, IEEE, pp. 32–46.

17 Xiaosong, Z., Xiaohui, P., and Xiaoshu, L. (2009) Analysis of virtual machine applied to malware detection system, in *Information Engineering and Electronic Commerce, 2009. IEEC'09. International Symposium on*, IEEE, pp. 290–294.

18 Wang, C., Pang, J., Zhao, R., Fu, W., and Liu, X. (2009) Malware detection based on suspicious behavior identification, in *Education Technology and Computer Science, 2009. ETCS'09. First International Workshop on*, vol. 2, IEEE, vol. 2, pp. 198–202.

19 Han, Q.L., Hao, Y.J., Zhang, Y., Lu, Z.P., and Zhang, R. (2008) A new malware detection method based on raw information, in *Apperceiving Computing and Intelligence Analysis, 2008. ICACIA 2008. International Conference on*, pp. 307–310, doi:10.1109/ICACIA.2008.4770030.

20 Gavrilut, D., Cimpoesu, M., Anton, D., and Ciortuz, L. (2009) Malware detection using machine learning, in *Computer Science and Information Technology, 2009. IMCSIT'09. International Multiconference on*, IEEE, pp. 735–741.

21 Zolkipli, M.F. and Jantan, A. (2010) A framework for malware detection using combination technique and signature generation, in *Computer Research and*

Development, 2010 Second International Conference on, IEEE, pp. 196–199.

22 Komashinskiy, D. and Kotenko, I. (2010) Malware detection by data mining techniques based on positionally dependent features, in *Parallel, Distributed and Network-Based Processing (PDP), 2010 18th Euromicro International Conference on*, IEEE, pp. 617–623.

23 Xu, J., Sung, A.H., Mukkamala, S., and Liu, Q. (2007) Obfuscated malicious executable scanner. *Journal of Research & Practice in Information Technology*, **39** (3).

24 Kerchen, P., Lo, R., Crossley, J., Elkinbard, G., and Olsson, R. (1990) Static analysis virus detection tools for unix systems, in *Proceedings of the 13th National Computer Security Conference*, pp. 350–365.

25 LISTON, T. (2007) Hiding virtualization from attackers and malware.

26 Willems, C., Holz, T., and Freiling, F. (2007) Toward automated dynamic malware analysis using cwsandbox. *IEEE Security and Privacy*, **5** (2), 32–39.

27 Yan, W., Zhang, Z., and Ansari, N. (2008) Revealing packed malware. *Security & Privacy, IEEE*, **6** (5), 65–69.

28 Hofmeyr, S.A., Forrest, S., and Somayaji, A. (1998) Intrusion detection using sequences of system calls. *Journal of computer security*, **6** (3), 151–180.

29 Sulaiman, A., Ramamoorthy, K., Mukkamala, S., and Sung, A.H. (2005) Disassembled code analyzer for malware (dcam), in *Information Reuse and Integration, Conf, 2005. IRI-2005 IEEE International Conference on.*, IEEE, pp. 398–403.

30 Henchiri, O. and Japkowicz, N. (2006) A feature selection and evaluation scheme for computer virus detection, in *Data Mining, 2006. ICDM'06. Sixth International Conference on*, IEEE, pp. 891–895.

31 Karnik, A., Goswami, S., and Guha, R. (2007) Detecting obfuscated viruses using cosine similarity analysis, in *Modelling & Simulation, 2007. AMS'07. First Asia International Conference on*, IEEE, pp. 165–170.

32 Bruschi, D., Martignoni, L., and Monga, M. (2007) Code normalization for self-mutating malware. *IEEE Security and Privacy*, **5** (2), 46–54.

33 Li, J., Mao, J., Wei, T., and Zou, W. (2009) A static method for detection of information theft malware, in *Electronic Commerce and Security, 2009. ISECS'09. Second International Symposium on*, vol. 1, IEEE, vol. 1, pp. 236–240.

34 Treadwell, S. and Zhou, M. (2009) A heuristic approach for detection of obfuscated malware, in *Intelligence and Security Informatics, 2009. ISI'09. IEEE International Conference on*, IEEE, pp. 291–299.

35 Ye, Y., Li, T., Jiang, Q., and Wang, Y. (2010) Cimds: adapting postprocessing techniques of associative classification for malware detection. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, **40** (3), 298–307.

36 Shaorong, F. and Zhixue, H. (2010) An incremental associative classification algorithm used for malware detection, in *Future Computer and Communication (ICFCC), 2010 2nd International Conference on*, vol. 1, IEEE, vol. 1, pp. V1–757.

37 Wang, W., Zhang, P., Tan, Y., and He, X. (2009) A hierarchical artificial immune model for virus detection, in *Computational Intelligence and Security, 2009. CIS'09. International Conference on*, vol. 1, IEEE, vol. 1, pp. 1–5.

38 Chao, R. and Tan, Y. (2009) A virus detection system based on artificial immune system, in *Computational Intelligence and Security, 2009. CIS'09. International Conference on*, vol. 1, IEEE, vol. 1, pp. 6–10.

39 Wang, W., Zhang, P., and Tan, Y. (2010) An immune concentration based virus detection approach using particle swarm optimization, in *Advances in Swarm Intelligence*, Springer, pp. 347–354.

40 Guo, Z., Liu, Z., and Tan, Y. (2004) An nn-based malicious executables detection algorithm based on immune principles, in *Advances in Neural Networks-ISNN 2004*, Springer, pp. 675–680.

41 Tan, Y. and Guo, Z. (2005) Algorithms of non-self detector by negative selection principle in artificial immune system, in *Advances in Natural Computation*, Springer, pp. 867–875.

42 Forrest, S., Hofmeyr, S., Somayaji, A., and Longstaff, T. (1996) A sense of self for unix processes, in *Security and Privacy, 1996. Proceedings., 1996 IEEE Symposium on*, IEEE, pp. 120–128.

43 Kim, J. and Bentley, P.J. (2001) Towards an artificial immune system for network intrusion detection: An investigation of clonal selection with a negative selection operator, in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, vol. 2, IEEE, vol. 2, pp. 1244–1252.

44 Matzinger, P. (2002) The danger model: a renewed sense of self. *Science's STKE*, **296** (5566), 301.

45 Lee, H., Kim, W., and Hong, M. (2004) Artificial immune system against viral attack, in *Computational Science-ICCS 2004*, Springer, pp. 499–506.

46 Edge, K.S., Lamont, G.B., and Raines, R.A. (2006) A retrovirus inspired algorithm for virus detection & optimization, in *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, ACM, pp. 103–110.

47 Li, Z., Liang, Y., Wu, Z., and Tan, C. (2007) Immunity based virus detection with process call arguments and user feedback, in *Bio-Inspired Models of Network, Information and Computing Systems, 2007. Bionetics 2007. 2nd*, IEEE, pp. 57–64.

48 González, F.A. and Dasgupta, D. (2003) Anomaly detection using real-valued negative selection. *Genetic Programming and Evolvable Machines*, **4** (4), 383–403.

49 Balachandran, S., Dasgupta, D., Nino, F., and Garrett, D. (2007) A framework for evolving multi-shaped detectors in negative selection, in *Foundations of Computational Intelligence, 2007. FOCI 2007. IEEE Symposium on*, IEEE, pp. 401–408.

50 Li, T. (2008) Dynamic detection for computer virus based on immune system. *Science in China Series F: Information Sciences*, **51** (10), 1475–1486.

51 Harmer, P.K., Williams, P.D., Gunsch, G.H., and Lamont, G.B. (2002) An artificial immune system architecture for computer security applications. *Evolutionary computation, IEEE transactions on*, **6** (3), 252–280.

52 Marhusin, M.F., Cornforth, D., and Larkin, H. (2008) Malicious code detection architecture inspired by human immune system, in *Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2008. SNPD'08. Ninth ACIS International Conference on*, IEEE, pp. 312–317.

53 Gong, T. (2008) Unknown non-self detection & robustness of distributed artificial immune system with normal model, in *Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on*, IEEE, pp. 1444–1448.

54 Zhang, Y., Li, T., and Qin, R. (2008) A dynamic immunity-based model for computer virus detection, in *Information Processing (ISIP), 2008 International Symposiums on*, IEEE, pp. 515–519.

55 Qin, R., Li, T., and Zhang, Y. (2009) An immune inspired model for obfuscated virus detection, in *Industrial Mechatronics and Automation, 2009. ICIMA 2009. International Conference on*, IEEE, pp. 228–231.

56 Zeng, J. and Li, T. (2009) A novel computer virus detection method from ideas of immunology, in *Multimedia Information Networking and Security, 2009. MINES'09. International Conference on*, vol. 1, IEEE, vol. 1, pp. 412–416.

57 Somayaji, A., Hofmeyr, S., and Forrest, S. (1998) Principles of a computer immune system, in *Proceedings of the 1997 workshop on New security paradigms*, ACM, pp. 75–82.

58 Cilpku08 malware dataset, `http://www.cil.pku.edu.cn/malware/`.

59 Resource, `http://www.cil.pku.edu.cn/resources/`.

9
Immune Concentration Based Malware Detection Approaches

In this chapter the immune concentration is applied to malware detection . The local concentration based malware detection method connects a certain number of two-element local concentration vectors as feature vector. To achieve better detection performance, particle swarm optimization (PSO) is used to optimize the parameters of local concentration. Then the hybrid concentration based feature extraction (HCFE) approach is presented by extracting the hybrid concentration (HC) of malware in both the global resolution and the local resolution.

9.1
Introduction

Immune concentration based malware detection approaches is mainly divided into three parts[1, 2, 3, 4, 5, 6].

- Generate 'self' and 'nonself' detector libraries from the randomly selected training set;
- Extract the immune concentration for each training sample to construct a feature vector;
- Three trained classifiers including KNN , RBF neural networks and SVM are used to detect the testing sample characterized by the ordered concentration vector.

The overview of the proposed algorithm is outlined in Algorithm 12.

The approach computes a statistical and information-theoretic feature in a manner of immune concentration on the byte-level file content . The generated feature vector of a program is then given as an input to standard data mining classification algorithms which classify the file as malware or not.

9.2
Generation of Detector Libraries

The operating principle of generating 'self' detector library and 'nonself' detector library is shown in Figure 9.2. The concrete step is to divide all detectors into two

Algorithm 12 Algorithm for Malware Detection

Generate *'self'* and *'nonself' detector libraries* from training set
The sizes of the libraries are decided by parameter m which corresponds to proportional selection of the *potential detectors*

for each the sample in training set **do**
Extract the *immune concentration based feature vector* from each training sample through the two detector libraries
end for

Use these *feature vectors* to train a certain classifier

while Algorithm is running **do**
if a program is detected **then**
Characterize the sample by *immune concentration based feature vector* through trained *'self'* and *'nonself' detector libraries*
Use trained classifier to predict the *label* of the program
end if
end while

sets by their tendency value and calculate these detectors' importance, with important detectors retained.

'Self' detector library are composed of detectors with utmost representative of benign files and 'nonself' detector library are composed of those detectors with utmost representative of malware. Intuitively, the fragment that appears most in malware programs while seldom in benign programs is a good representative of malware.

The detectors in the library are a set of fixed-length fragments. Here a fixed length L-bit fragment of binary data which is considered containing appropriate information of functional behaviors is taken as the detector to discriminate malware from benign program. The length L is set not too short to discriminate 'self' and 'nonself' and not too long to make malware-special data hidden in the binary data of files. Considering that one meaningful computer instruction is 8 or 16 bits normally, it is reasonable to set 'L' as 16, 32 or 64. A sliding window (shown in Figure 9.1, the overlap of sliding window is [L/2] bits) is used to count the document frequency of a detector in the malware programs and benign programs. The difference of its document frequency in the malware programs and benign programs can reflect its tendency to be a malware or a benign file.

After counting the document frequency of each fragment, the tendency $T(X)$ of fragment X is defined in formula 9.1.

$$T(X) = P(X = 1|C_v) - P(X = 1|C_s) \tag{9.1}$$

$P(X = 1|C_v)$ means document frequency of fragment X appears in malware samples of training set;
$P(X = 1|C_s)$ means document frequency of fragment X appears in benign samples

Index