DOI:10.3969/j.issn.1673-4785.201209059
Network Publishing Address:http://www.cnki.net/kcms/detail/23.1538.TP.20130205.1834.001.html

# Immune based computer virus detection approaches

TAN Ying[1,2], ZHANG Pengtao[1,2]
(1. Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China; 2. Key Laboratory of Machine Perception, Ministry of Education, Peking University, Beijing 100871, China)

**Abstract**: The computer virus is considered one of the most horrifying threats to the security of computer systems worldwide. The rapid development of evasion techniques used in virus causes the signature based computer virus detection techniques to be ineffective. Many novel computer virus detection approaches have been proposed in the past to cope with the ineffectiveness, mainly classified into three categories: static, dynamic and heuristics techniques. As the natural similarities between the biological immune system (BIS), computer security system (CSS), and the artificial immune system (AIS) were all developed as a new prototype in the community of anti-virus research. The immune mechanisms in the BIS provide the opportunities to construct computer virus detection models that are robust and adaptive with the ability to detect unseen viruses. In this paper, a variety of classic computer virus detection approaches were introduced and reviewed based on the background knowledge of the computer virus history. Next, a variety of immune based computer virus detection approaches were also discussed in detail. Promising experimental results suggest that the immune based computer virus detection approaches were able to detect new variants and unseen viruses at lower false positive rates, which have paved a new way for the anti-virus research.

**Keywords**: computer virus detection; artificial immune system; immune algorithms; hierarchical model; negative selection algorithm with penalty factor
**CLC Number**: TP309.5 **Document Code**: A **Article ID**: 1673-4785(2013)01-0080-15

Due to the rapid development of computer technology and the Internet, the computer has become a part of daily life in the 21st century. Meanwhile, the computer security systems are getting more and more notice. The computer viruses, new variants and unseen viruses in particular, have been one of the most dreadful threats to the computers worldwide. Today viruses are becoming more complex with faster propagation speed and stronger ability for latency, destruction and infection. At present a virus is able to spread all over the world in a matter of minutes and results in huge economic losses. The mission of how to protect computers from these various types of viruses has become priority number one.

**Received Date**: 2012-09-27. **Network Publishing Date**: 2013-02-05.
**Foundation Item**: National Natural Science Foundation of China (No. 61170057, 60875080).
**Corresponding Author**: TAN Ying. E-mail: ytan@pku.edu.cn.

Currently, there are several companies that produce various anti-virus products, most of which are based on signatures. These products are usually able to detect known viruses effectively with lower false positive rates and overheads. Unfortunately, these same products fail to detect new variants and unseen viruses. Based on the metamorphic and polymorphous techniques, even a layman can develop new variants of known viruses easily using virus automatons. For example, the Agobot has observed more than 580 variants from its initial release, which makes use of polymorphism to evade detection and disassembly[1]. Thus, traditional signature based computer virus detection approaches are no longer suitable for the new environments; dynamic and heuristics techniques as well have started to emerge.

Dynamic techniques, such as virtual machine,

keep watch over the execution of every program during run-time and stop the program once it tries to harm the system. Most of these techniques monitor the behaviors of a program by analyzing the application programming interface (API) call sequences generated at runtime. As the huge overheads of monitoring API calls, it is practically impossible to deploy the dynamic techniques on personal computers at this time.

Data mining approaches, one of the most popular heuristics, try to mine frequent patterns or association rules to detect viruses by using classic classifiers. These approaches have led to some success. However, data mining approaches lose the semantic information of the code and cannot easily recognize unseen viruses in the long term.

The computer virus is named after the biological virus, due to the similarity between them, such as parasitism, propagation and infection. The biological immune system (BIS) protects organisms from antigens for a long time, resolving the problem to detect unseen antigens successfully[2]. Inspired from the BIS, applying immune mechanisms to detect computer viruses has developed into a new anti-virus field in the past few years, attracting many researchers. Forrest et al. applied the immune theory to computer anomaly detection for the first time in 1994[3]. Since then, many researchers have proposed various kinds of virus detection approaches and achieved some success. Some of them are mainly derived from ARTIS[4-6].

As time goes on, more and more immune mechanisms become clear which lay a good foundation for the development of the AIS. On this basis, many immune based computer virus detection approaches have been proposed, in which more and more immune mechanisms are involved. The simulations of the AIS to the BIS keep going on and the immune based computer virus detection approaches have paved a new way for the anti-virus research.

The researchers of this paper have done some related works in the anti-virus field and achieved some success[7-9]. They have tried to make full use of the relativity among different features in a virus sample by constructing an immune based hierarchical model[7]. On the basis of the traditional negative selection algorithms (NSA), a novel negative selection algorithm with penalty factor (NSAPF) was proposed to overcome the drawback of the traditional NSA in defining the harmfulness of "self" and "nonself". It focuses on the danger of the code and greatly improves the effectiveness of the NSAPF based virus detection model.

The rest of this paper was organized as follows: In Section 1, the background knowledge of computer viruses is introduced. Section 2 presents a variety of classic computer virus detection approaches. In Section 3, the artificial immune system and immune based computer virus detection approaches are briefly described. Our previous works and conclusions are proposed in detail in Sections 4 and 5, respectively.

# 1 Computer virus

## 1.1 Definition and features

In a narrow sense, a computer virus is a program that can infect other programs by modifying them to include a possibly evolved copy of it[10]. In a broad sense, a computer virus indicates all the malicious code that is a program designed to harm or secretly access a computer system without the owners' informed consent; such as viruses in the narrow sense, worms, backdoors and Trojans[11]. Through the development of the computer virus, the lines have become blurred between the different types of viruses and are not clear. In this paper, all the programs that are not authorized by users and perform harmful operations in the background are referred to as viruses.

The features of the computer viruses are listed below.

1) Infectivity: Infectivity is the fundamental and essential feature of the computer virus in the narrow sense, which is the foundation to detect a virus. When a virus intrudes into a computer system, it starts to scan the programs and computers on the Internet that can be infected. Next, through self-duplicating, it spreads to the other programs and computers.

2) Destruction: According to the extent of destruction, the virus is divided into "benign" virus and malignant virus. "Benign" viruses merely occupy system resources, such as GENP, W-BOOT, while malignant viruses usually have clear purposes. They can destroy data; delete files, even format diskettes.

3) Concealment: Computer viruses often attach

themselves to benign programs and start up with the host programs. They perform harmful operations in the background hiding from users.

4) Latency: After intruding into a computer system, the viruses usually hide themselves from users instead of attacking the system immediately. This feature makes the viruses have longer lives. They spread themselves and infect other programs in this period.

5) Trigger: Most viruses have one or more trigger conditions. When these conditions are satisfied, the viruses begin to destroy the system. Other features of the viruses include illegality, expressiveness, and unpredictability.

### 1.2 Development phases of the viruses

The viruses are evolved with the computer technology all the time. The development of the viruses approximately goes through several phases which are described below.

1) DOS boot phase: Fig. 1 and Fig. 2 illustrate the boot procedures of DOS without and with boot sector virus, respectively. Before the computer system obtains the control right, the virus starts up, modifies interrupt vector and copies it to infect the diskette. These are the original infection procedures of the viruses. What is more, the similar infection procedures can be found in the viruses nowadays.

2) DOS executable phase: The viruses exist in a computer system in the form of executable files in this phase. They would control the system when the users run applications infected by the viruses. Most viruses now are executable files.

3) Virus generator phase: Virus generators, also called virus automatons, can generate new variants of known viruses with different signatures. Metamorphic techniques are used here to obfuscate virus scanners which are based on virus signatures, including instruction reordering, code expansion, code shrinking and garbage code insertion[12].

4) Macro virus phase: Before the emerging of macro viruses, all the viruses merely infect executable files as it is almost the only way for the viruses to obtain the right of execution. When users run the host of a virus, the virus starts up and controls the system. Infecting data files cannot help the virus to run itself. The emerging of macro viruses changed this situation and their punching bags are data files, mainly Microsoft Office files.

5) Virus techniques merging with hacker techniques: Nowadays merging of virus techniques and hacker techniques has been a tendency. It makes the viruses have much stronger concealment, latency and much faster propagation speed than ever before.

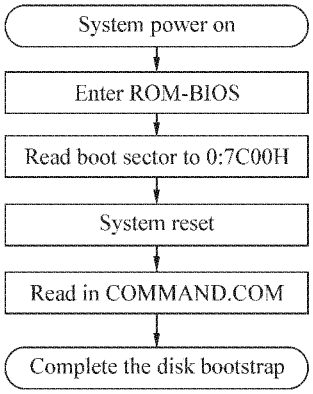



**Fig. 1 Normal boot procedure of DOS**

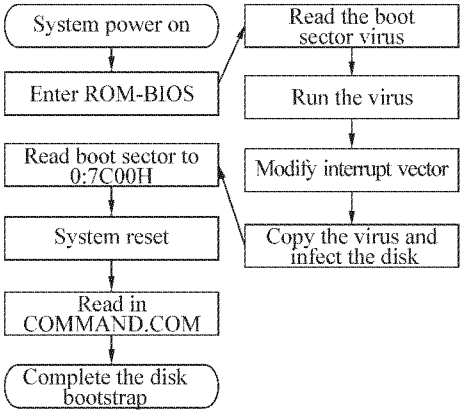



**Fig. 2 Boot procedure of DOS with boot sector virus**

## 2 Classic virus detection approaches

The computer virus has become a major threat to the security of computers and the Internet worldwide. A wide range of host-based anti-virus solutions have been proposed by many researchers and companies[13-46]. These anti-virus techniques could be broadly classified into three categories: static techniques, dynamic techniques and heuristics.

The fight between the viruses and the anti-virus techniques is more violent now than ever before. The viruses disguise themselves by using various kinds of evasion techniques, such as metamorphic and polymorphous techniques, packer and encryption techniques. Coping with the new situations, the anti-virus techniques unpack the suspicious programs, decrypt them

and try to be robust to those evasion techniques. Nevertheless, the viruses evolve to anti-unpack anti-decrypt and develop into obfuscating the anti-virus techniques again. The fight will never stop and the virus techniques will always be ahead of the anti-virus techniques. What can we do is to increase the difficulty of intrusion, decrease the losses caused by the viruses and react to them as soon as possible.

### 2.1 Static techniques

Static techniques usually work on program 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.

The signature based virus detection technique is the mainstream anti-virus approach and most of the commercial anti-virus products are based on this technique. A signature usually is a bit string which is divided from a virus sample and it is able to identify a virus uniquely. The signature based anti-virus products are referred to as scanners in this paper.

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

This technique is able to detect known virus very quickly with lower false positive and high true positive rates. It is one of the simplest approaches with minimal overheads. Nevertheless, since a signature of a new virus can be only extracted after the break out of the virus by experts, it would take a long time to detect the new virus effectively. The losses caused by the virus already cannot be recovered. Furthermore, with the development of virus techniques, there are many evasion techniques which are used to help the virus evade from the signature based scanners, such as metamorphic and polymorphous techniques, packer, and encryption techniques. The signature based anti-virus techniques are easily defeated by these techniques. For example, simple program entry point modifications consisting of two extra jump instructions effectively defeat most signature based scanners[13].

To conclude, the signature based anti-virus techniques are vulnerable to unseen viruses and the evasion techniques of viruses. As a result, a variety of dynamic and heuristic anti-virus approaches are developed to cope with these situations.

### 2.2 Dynamic techniques

Computer viruses often show some special behaviors when they harm the computer systems. For example, writing operation to executable files, dangerous operations (e. g., formatting a diskette), and switching between a virus and its host. These behaviors give us an opportunity to recognize the viruses. Based on the above idea, the dynamic techniques keep watch over the execution of every program during run-time and observe the behaviors of the program. They would stop the program once it tries to harm the computer system. 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[14].

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

Based on the assumption that the viruses have some special behaviors that can identify themselves and would never emerge in benign programs, the behavior monitors keep watch over every behavior of a virus and wish to prevent destruction from the dangerous operations effectively.

This approach is considered to be able to detect known viruses, new variants and unseen viruses, whereas it is very dangerous to run viruses in a real computer by using this approach. If a behavior monitor fails to kill a virus, the virus would take control of the computer. Moreover, the overheads brought in by a behavior monitor are too huge to personal computers. The false positive rate of this approach is high inevitably and the approach cannot recognize the type and name of a virus, thus it cannot eliminate the virus from a computer. Furthermore, it is very hard to implement a relative perfect behavior monitor.

In order to separate the running program from the real computer, the virtual machine approach creates a virtual machine (VM) and runs the programs in the VM. The execution environment of a program here is the VM which is software, instead of the physical ma-

chine. Hence the computer is safe, even when the VM is crashed by a virus. It is very easy to collect all the information while a program is running in a VM. If the VM captures any dangerous operation, it would give the users a tip. When it confirms that the running program is a virus, it will kill the virus.

The virtual machine approach is very safe and can recognize almost all the viruses, including encrypted and packed viruses. Now the VM approach has become one of the most amazing virus detection approaches. However, the VM brings comparable overheads to the host computers. How to implement a relative perfect VM is a new research study. In addition, the VM only simulates a part of the computer's functions which provides opportunities for anti-VM techniques to evade from the VM approach.

Anti-VM techniques have been used in many viruses recently. For example, inserting some special instructions into a virus may cause the crash of a VM. The entry point obscuring is also involved by the viruses to evade from the VM approach.

Ref. [15-20] proposed some new dynamic techniques based virus detection models. Although these models have shown promising results, they can produce high false positive rates, an issue which has yet to be resolved[21].

### 2.3 Heuristics

Schultz et al., who are pioneers to apply the techniques of machine learning and data mining to the anti-virus field, proposed a data mining framework to detect unseen virus effectively and automatically[22]. Three approaches are taken to the feature extraction procedure. The first one makes use of Bin-Utils[23] of GNU to extract resource information of a program. String sequences are extracted by using GNU strings program in the second approach. The third approach is called hex dump[24] which transforms binary files into byte sequences. However, DLL and function names are too unstable to detect virus. This work lays a good foundation for the application of the techniques of machine learning and data mining in the anti-virus field.

Kolter et al. proposed a technique to detect virus in the field based on the relevant N-Grams selected by using the information gain (IG)[25]. They clearly identified that using the techniques from machine learning and data mining to detect virus is feasible. N-Gram is a concept from text categorization, which means $N$ continuous words or phrases. In the anti-virus field, an N-Gram is usually defined as a binary string of length $N$ bytes. The experimental results revealed that the boosted decision trees outperformed other classifiers with an area under the receiver operating characteristic curve (AUC), 0.996. Later they extended this technique to classify virus according to the functions of their payloads[26].

A new feature selection criterion, class-wise document frequency (CDF), was proposed by Reddy et al. and applied to the procedure of N-Gram selection[27]. Their experimental results suggested that the CDF outperformed the IG in the feature selection process. They guessed the reason might be most of the relevant N-Grams selected by using the IG came from benign programs. What is more, since the CDF tries to select the features with the highest frequencies in a specific class, it has a bias to the information of the class. As a result, it could not select the discriminating features for the class effectively.

Stolfo et al. made use of N-Grams to identify file types[28] and later to detect stealthy virus[29-30]. Their experimental results showed that the method was able to detect embed virus. However, this method was not a general virus detection method.

Sulaiman et al. proposed a static analysis framework for detecting variants of viruses which was called disassembled code analyzer for virus (DCAM)[31]. Different from the traditional static code analysis which usually works on the binary string of a program, the authors extracted virus features from disassembled codes. The programs which got through three steps of matching were considered as benign programs; otherwise the DCAM classified the programs as viruses. The experimental results suggested that the DCAM worked very well and could prevent breakouts of previous identified viruses.

Henchiri and Japkowicz adopted a data mining approach to extract the frequent patterns (FPs) to detect virus[32]. They filtered FPs twice and tried to obtain general FPs based on the intra-family support and inter-family support. Several classifiers were involved in this work, such as the J48 decision tree and naive

Bayes. They verified the effectiveness of their model using 5-fold cross validation, showing good results.

A virus detection model using cosine similarity analysis to detect obfuscated viruses was proposed in Ref. [33]. This work was based on the premise that given a variant of a virus, they can detect any obfuscated version of the virus with high probability. Actually this model was only worked on code transposition technique. The biggest issue in this model was that how to extract functions within a program cannot be completed in real time.

Ye et al. made use of associative classification and post-processing techniques to detect virus[34]. Firstly, they extracted the API calls from Windows PE files as the features of the samples, and stored them in a feature database. Then they extended a modified FP-Growth algorithm proposed in Ref. [35-36] to generate the association rules. Finally, the authors reduced the number of the rules and got a concise classifier by using post-processing techniques. Promising results demonstrated that this model outperformed popular anti-virus software as well as previous data mining based virus detection systems.

Tabish et al. proposed a virus detection model using statistical analysis of the byte-level file content[37]. This model worked on 1-, 2-, 3-, 4-Gram. And 13 statistical features were computed on the basis of N-Grams. This model was not based on signatures. It neither memorized specific strings appearing in the file content nor depended on prior knowledge of file types. However, the false positive rates were relatively high. And how to set the block size was not introduced.

Very recently, many new virus detection methods have been proposed, for details, please refer to Ref. [38-46].

# 3 Immune based computer virus detection approaches

## 3.1 Artificial immune system

### 3.1.1 Features and applications of AIS

The AIS is a computational system inspired by the BIS, which are referred to as the second brain. The AIS is a dynamic, adaptive, robust and distributed learning system. As it has the ability of fault tolerant and noise resistant, it is suitable for the applications in time-varying unknown environment.

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. The steps of a general immune algorithm are shown in Algorithm 1.

**Algorithm** 1 A general immune algorithm

Step 1): Input antigens;

Step 2): Initialize antibody population;

Step 3): Calculate the affinities of the antibodies;

Step 4): Lifecycle event and update the antibodies - creation and destruction;

Step 5): If the terminate criteria are satisfied, go to Step 6); otherwise, go to Step 3);

Step 6): Output the antibodies.

There are four main algorithms in the AIS field: negative selection algorithm (NSA), clonal selection algorithm, immune network model, and danger theory based immune algorithms. The principle of the NSA is shown in Fig. 3.

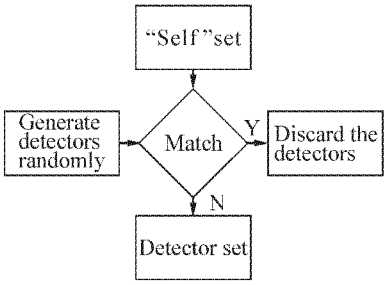



**Fig. 3 The principle of the negative selection algorithm**

Let us take the computer virus detection problem as an example to introduce the NSA. Firstly, the NSA generates virus detectors randomly which are referred to as "nonself", while the benign programs are taken as "self". Secondly, matches between "self" and "nonself" are done. If a detector matches a "self", it would be considered as "self" and discarded; otherwise, it is included in a detector set. Finally, the NSA obtains a detector set in which none of the detectors matches any "self", and which is then used to detect viruses.

The "nonself" prior knowledge is not needed in the procedure of extracting the detector set by using the NSA, so the NSA based approaches are able to classify "self" and "nonself" without the knowledge of "nonself". This feature makes the NSA based approaches a-

ble to recognize unseen "nonself". Now the NSA is mainly used in the computer security and fault diagnosis fields.

3.1.2 Motivations of applying immune mechanisms to detect virus

As we know, the computer virus is named after biological virus because of the similarity between them, such as parasitism, propagation, infection, and destruction. The BIS has protected body from antigens from the very beginning of life successfully, resolving the problem of defeating unseen antigens[2]. 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 (CAVS). To sum up, applying immune mechanisms to computer security system, especially the CAVS is reasonable and has developed into a new field in the past few years, attracting many researchers. The relationship of the BIS and CAVS is listed in Table 1.

**Table 1 The relationship of the BIS and CAVS**

| BIS | CAVS |
|---|---|
| Antigens | Computer viruses |
| Antibodies | Detectors for the viruses |
| Binding of an antigen and an antibody | Pattern matching of the viruses and detectors |

Applying immune mechanisms to detect virus enables the CAVS to recognize new variants and unseen viruses by using existing knowledge. The CAVS with immune mechanisms would own many finer features, such as dynamic, adaptive and robust. It is considered to be able to make up the fault 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 few years.

### 3.2 Related works

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[47]. 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 Ref.[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 proposed by Kim and Bentley[48].

Matzinger proposed the "danger theory" in 2002[49]. The danger theory (DT) believes that the immune system is more concerned with danger than nonself. It explains a lot of new findings successfully and corrects the fault of traditional "self" and "nonself" model in defining of harmfulness of "self" and "nonself". Many researchers have tried to introduce this new theory into AIS which has developed into a new branch of AIS.

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 computer virus detection approaches have been proposed[50-65].

Edge et al. introduced a new artificial immune system based on retrovirus algorithm (REALGO) which was inspired by reverse transcription ribonucleic acid (RNA) as found in the biological systems[51]. In the learning phase, positive selection generated new antibodies using genetic algorithm based on known virus signatures and negative selection, ensuring 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. With 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. proposed an immune based virus detection approach with process call arguments and user feedback[52]. 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[53]. In the test phase, 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 unseen viruses. However, let users distinguish a virus from normal files and give feedback was very difficult.

Li Tao proposed a dynamic detection model for computer viruses based on an immune system[55]. 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".

A DT inspired artificial immune algorithm for online supervised two-class classification problem was proposed[63]. The size of the danger zone in this algorithm is decreased with the increasing of the accumulated intensity of the antibody. The better antibodies will proliferate and live longer by using the clonal selection algorithm, while a suppression mechanism is utilized to control the antibody population. Experimental results suggested that this algorithm performed well with good generalization capability.

Zhu and Tan proposed a DT based learning model for combing classifier and applied it to spam detection[64]. There are three components in this model: binary-valued signal 1, signal 2 and danger zone. If the signal 1 and signal 2 make the same classification for a sample, the sample is classified directly. Otherwise, a self-trigger process has to be done to solve the signal conflict problem. The classifiers used to emit immune signals are supposed to be conditionally independent, in order to get different trained classifiers from the same data source.

The immune based computer virus detection approaches are able to detect new variants and unseen viruses. These approaches have developed into a new field for computer virus detection and attracted more and more researchers. However, there is a lack of rigorous theoretical principle of mathematics. In addition, the simulations of the AIS to the BIS are still very simple. Combining with the characteristics of computer virus detection and the studies of immune algorithms are needed. There is still a long way to apply immune based computer virus detection approaches in the real world.

# 4 Our work

## 4.1 An immune based virus detection system using affinity vectors

### 4.1.1 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[8].

Firstly, the IVDS generates a detector set from a training set by using negative selection and clone selection. The negative selection eliminates autoimmunity detectors and ensures that any detector in the detector set will not match any "self", while the clone selection increases the diversity of the detectors in the detector set which helps the IVDS obtain a stronger ability to recognize new variants and unseen viruses. Secondly, two novel hybrid distances named hamming-max and shift r bit-continuous distance are proposed to calculate the affinity vector of a program. Finally, based on the affinity vector, three classic classifiers, support vector machine (SVM), radial basis function (RBF) neural network and K-nearest neighbor (KNN), are involved to estimate the performance of the proposed IVDS.

### 4.1.2 Experiments and discussions

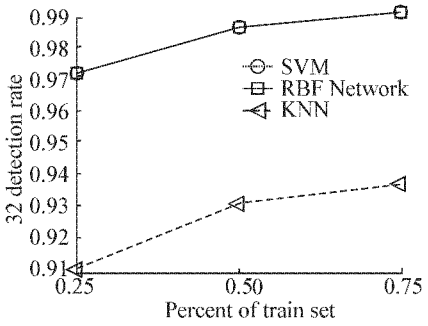
The experiments were conducted in the CILPKU08 dataset[66] which are collected by the Computational Intelligence Laboratory at Peking University in 2008. There are 3 547 malware in this dataset. Three test datasets used here are obtained by randomly dividing the CILPKU08 dataset, the details of which are shown in Table 2.

**Table 2 The test datasets used in the experiments**

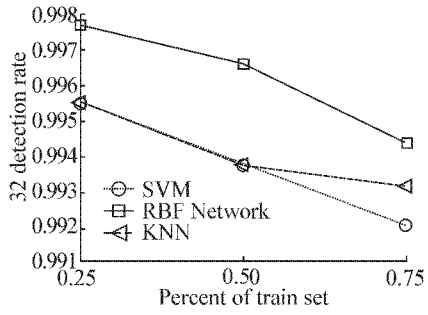
| Datasets | Training set | | Test set | | The percentage of training set |
|---|---|---|---|---|---|
| | Benign programs | Viruses | Benign programs | Viruses | |
| Dataset 1 | 71 | 885 | 213 | 2 662 | 0.25 |
| Dataset 2 | 142 | 1 773 | 142 | 1 773 | 0.50 |
| Dataset 3 | 213 | 2 662 | 71 | 885 | 0.75 |

The percentage of training set given in Table 2 is set as NTS / (NTS + NDS), where the NTS and NDS denote the number of the programs in a training set and a test set, respectively. There is no overlap between a

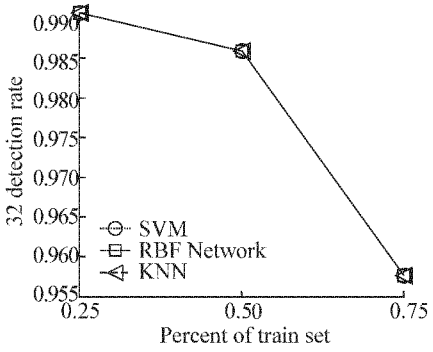
test set and its corresponding training set. This setting makes the experiments believable. The experimental results are shown in Fig. 4.



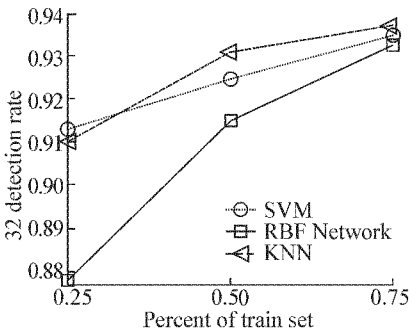

(a) Detect benign files in train set




(b) Detect virus files in train set




(c) Detect benign files in test set




(d) Detect virus files in test set

**Fig. 4 The accuracies of the SVM, RBF network and KNN**

Fig. 4 illustrates that the IVDS achieves the optimal accuracy when the percentage of the training set is 25%. The RBF network outperformed the SVM and KNN in all the training sets, while it got worse accuracies in the test sets. It is easy to see that the RBF network has weaker generalization ability in this context, whereas the SVM and KNN have stable performances in all the training sets and test sets with different percentages of the training sets. The experimental results suggested that the proposed IVDS has a strong detection ability and good generalization performance.

## 4.2 A hierarchical artificial immune model for virus detection

### 4.2.1 Overview

A hierarchical artificial immune model for virus detection (HAIM) was proposed in Ref. [7]. The motivation of the HAIM is to make full use of the relativity among the different signatures in a virus sample. Generally speaking, a virus usually contains several heuristic signatures and a heuristic signature may appear in various viruses. It is reasonable to believe that there is some relativity among these heuristic signatures and a specific combination of some signatures makes up a virus. The HAIM, taking a virus as a unit, detects viruses by making full use of the simple relativity among 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 in a training set, while 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 Fig. 5 and Fig. 6, respectively. The virus gene library generating module extracts a virus instruction library based on the statistics collected in a training set. Here an instruction is defined as a binary string of length 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 by using a sliding window, respectively. Finally, according to the negative selection mechanism, the candidate virus library is upgraded to the detecting virus gene library.

The virus gene library generating module extracts a virus instruction library based on the statistics collected in a training set. Here an instruction is defined as a binary string of length 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 by using a sliding window, respectively. Finally, according to the negative selection mechanism, the candidate virus library is upgraded to the detecting virus gene library.

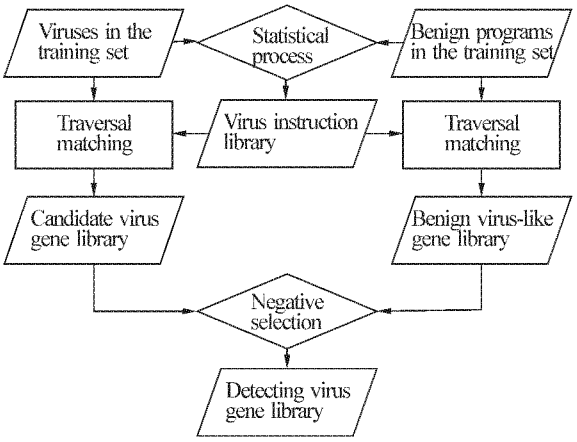



**Fig. 5　Virus gene library generating process**

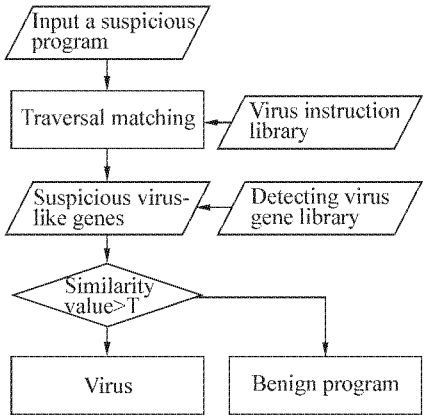



**Fig. 6 Self-nonself classification process**

The virus gene library generating module extracts a virus instruction library based on the statistics collected in a training set. Here an instruction has been defined as a binary string of length 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 by using a sliding window, respectively. Finally, according to the negative selection mechanism, the candidate virus library is upgraded to the detecting virus gene library.

In the self-nonself classification module, the suspicious virus-like genes are extracted from a suspicious program by using the method to generate the candidate virus gene library. Then the virus-like genes in the suspicious program are matched with the detectors in the detecting virus gene library to get a matching value, which is taken as the affinity of the program. If it is larger than a chosen threshold, the program is regarded as a virus; otherwise it is a legal program.

The hierarchical matching method to calculate the affinity of a suspicious program is illustrated in Fig. 7. In the gene level, T-successive consistency matching is used to make a fuzzy matching. In the individual level, a suspicious program is compared to a virus sample in the detecting virus gene library on the individual level. As the interrelated information of instructions is saved as much as possible, the HAIM takes full advantage of the potential relevance between different signatures to detect viruses. Due to the similarity among different viruses, it is able to detect new variants and unseen viruses effectively. Finally, a classification decision is made by the detecting virus gene library. By getting through the matching processes with three levels, a wise decision was made to successfully classify the program.

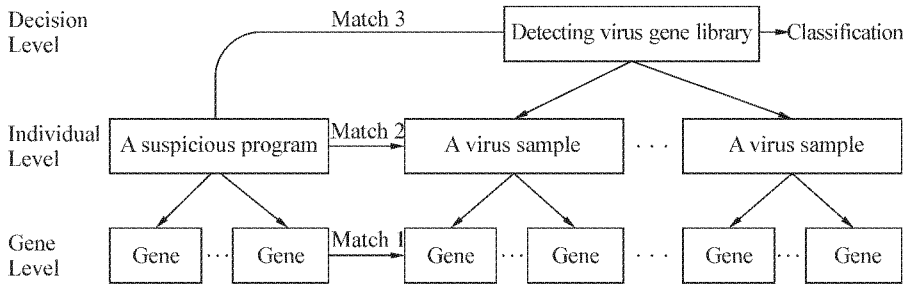



**Fig. 7 The hierarchical matching method**

4.2.2　Experiments and discussions

The experiments in this work are taken on the CILPKU08 dataset[66]. Through randomly dividing the dataset nine times, nine test datasets are obtained and nine tests are carried out, respectively. The experimental results are listed in Table 3.

**Table 3 Experimental results of the HAIM** %

| Testset | 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 |

Table 3 illustrated that the proposed HAIM is a very stable model with excellent performance. It achieves high true positive rates in detecting unseen viruses in the test sets with all the false positive rates lower than 2%. The average true positive rate achieves 93.27% in detecting unseen viruses in the test sets which is relatively high.

### 4.3 A virus detection model based on a negative selection algorithm with penalty factor

#### 4.3.1 Overview

The negative selection algorithm (NSA) 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 detectors matches "self", and it is then used to detect virus. A traditional NSA assumes that the entire "self" are harmless and all the "nonself" are harmful. However, this is not always the case in organisms. Taking cancer cells as an example, not all the "self" are harmless; and similarly, not all the "nonself" is harmful, for example, food. A computer security system, therefore, only has to identify dangerous virus instead of reacting to all the "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 a virus. This kind of dangerous signatures provides some useful information. In fact, the operation of formatting a disk can be included in both viruses and benign programs. Deleting such dangerous code snippets, as is done by the traditional NSA, destroys useful information, which is obviously a disadvantage for a virus detection model.

Theoretically, every program, regardless of whether it is a benign program or a virus, can use almost all the instructions and functions in a computer system. Moreover, almost all the functions used in a virus are also able to be used in 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 deleting too many detectors.

Based on the above analysis, a virus detection model based on a negative selection algorithm with penalty factor (MDM-NSAPF) was proposed to overcome the drawback of the traditional NSA in defining the harmfulness of "self" and "nonself"[67]. The MDM-NSAPF consists of a virus signature extraction module (MSEM) and a suspicious program detection module (SPDM). A flowchart of the MSEM is shown in Fig. 8.

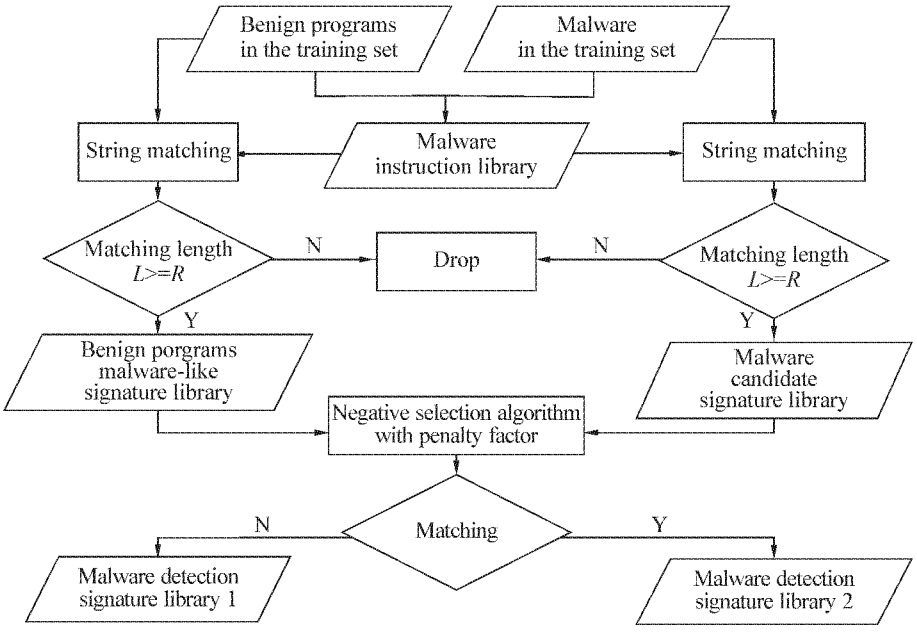



**Fig. 8 The flowchart of the MSEM**

In the MSEM, a virus candidate signature library (MCSL) and a benign program virus-like signature library (BPMSL) are extracted, respectively, from the virus and benign programs in the training set after generating the virus instruction library (MIL). Taking the MCSL as "nonself" and the BPMSL as "self", a NSAPF is introduced in to extract the virus detection signature library (MDSL) which consists of the MDSL1 and MDSL2. The signatures in the MDSL1 are characteristic signatures of "nonself", whereas the signatures in the MDSL2 are dangerous ones belonging to both "self" and "nonself" which should be penalized by using a penalty factor $C$.

In the SPDM, the signatures of suspicious programs are extracted by using the MIL. Then r-continuous bit matching is computed between the signatures of the suspicious program and the MDSL. If the matching value exceeds a given classification threshold, we classify the program as a virus; otherwise it is considered to be a benign program.

4.3.2 Experiments and discussions

The experiments in this work were conducted in the three datasets: Henchiri dataset, CILPKU08 dataset, and VX Heavens dataset[66]. The experimental results on the Henchiri dataset and CILPKU08 dataset are shown in Fig. 9 and Fig. 10, respectively. Fig. 9 indicates that 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 the 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 Fig. 10, the overall accuracy of the MDM-NSAPF is about 1% to 3% higher than that of HAIM.

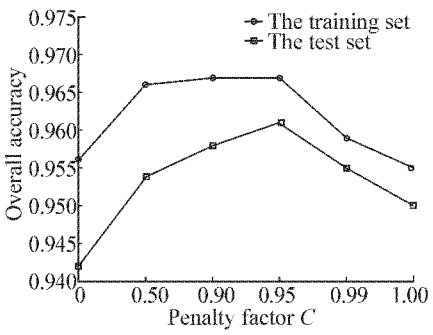



**Fig. 9 Results for Henchiri dataset**

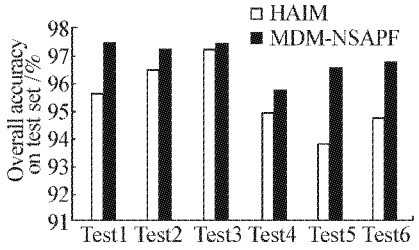



**Fig. 10 Results for CILPKU08 dataset**

The area under the receiver operating characteristic curve (AUC) was taken as the measure of the effectiveness of the MDM-NSAPF on the VX Heavens dataset. The results are shown in Fig. 11. Comparing to the results of Tabish[37], the optimal AUC of the MDM-NSAPF is about 0.04 higher on average.

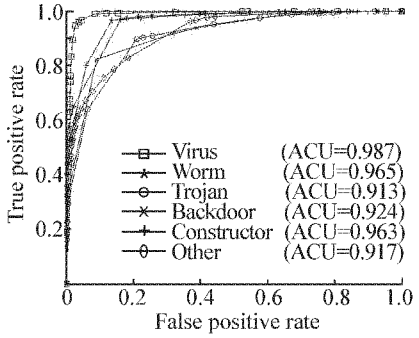



**Fig. 11 Experimental results for VX Heavens dataset**

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 MDM-NSAPF 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 MDM-NSAPF model is effective to detect unseen virus with lower false positive rates.

## 5 Conclusions

It was discovered that the classic virus detection approaches cannot detect new variants and unseen viruses efficiently, and thus, novel virus detection approaches are required urgently. Immune based computer virus detection approaches, due to their strong capability to detect unseen viruses, have been developed as a new field in the community of the anti-virus research. Many researchers have proposed a variety of virus detection models based on immune mechanisms and achieved some major success. As it turns out, with the promising experimental results of the immune based computer virus detection approaches; they are able to detect new variants and unseen viruses at lower false

positive rates and the overheads are limited. However, there is a lack of rigorous theoretical principles of mathematics applied in the study. The simulations of the AIS to the BIS are still very simple. There is still a long way to go for us to apply the immune based computer virus detection approaches in the real world.

## References:


[1] BAILEY M, OBERHEIDE J, ANDERSEN J, et al. Automated classification and analysis of internet malware[C]//The 10th Symposium on Recent Advances in Intrusion Detection. Gold Coast, Australia, 2007: 178-197.

[2] PERELSON A S, WEISBUCH G. Immunology for physicists [J]. Reviews of Modern Physics, 1997, 69(4): 1219-1268.

[3] FORREST S, PERELSON A S, ALLEN L, et al. Self-nonself discrimination in a computer[C]//IEEE Computer Society Symposium on Research in Security and Privacy. Oakland, USA, 1994: 202-212.

[4] KEPHART J O, ARNOLD W C. Automatic extraction of computer virus signatures[C]//The 4th Virus Bulletin International Conference. Jersey Islands, UK, 1994: 178-184.

[5] KEPHART J O, SORKIN G B, SWIMMER M, et al. Blueprint for a computer immune system[C]//Proceedings of the Seventh International Virus Bulletin Conference. San Francisco, USA, 1997: 159-173.

[6] OKAMOTO T, ISHIDA Y. Distributed approach against computer viruses inspired by the immune system[J]. IEICE Transactions on Communications, 2000, 83(5): 908-915.

[7] WANG Wei, ZHANG Pengtao, TAN Ying, et al. A hierarchical artificial immune model for virus detection[C]//International Conference on Computational Intelligence and Security. Beijing, China, 2009: 1-5.

[8] CHAO Rui, TAN Ying. A virus detection system based on artificial immune system[C]//International Conference on Computational Intelligence and Security. Beijing, China, 2009: 6-10.

[9] WANG Wei, ZHANG Pengtao, TAN Ying. An immune concentration based virus detection approach using particle swarm optimization [C]//International Conference on Swarm Intelligence. Beijing, China, 2010: 347-354.

[10] COHEN F. Computer viruses: theory and experiments[J]. Computers and Security, 1987, 6(1): 22-35.

[11] FU Jianming, PENG Guojun, ZHANG Huanguo. Computer virus analysis and confronting[M]. Wuhan, China: Wuhan University Press, 2009.

[12] DAOUD E A. Metamorphic viruses detection using artificial immune system [C]//International Conference on Communication Software and Networks. Macau, China, 2009: 168-172.

[13] XU J Y, SUNG A H, MUKKAMALA S, et al. Obfuscated malicious executable scanner[J]. Journal of Research and Practice in Information Technology, 2007, 39: 181-197.

[14] KERCHEN P, LO R, CROSSLEY J, et al. Static analysis virus detection tools for unix systems[C]//13th National Computer Security Conference. Washington, DC, USA, 1990: 4-9.

[15] CHRISTODORESCU M, JHA S, SESHIA S A, et al. Semantics-aware malware detection[C]//IEEE Symposium on Security and Privacy. Berkeley/Oakland, USA, 2005: 32-46.

[16] CARPENTER M, LISTON T, SKOUDIS E. Hiding virtualization from attackers and malware[J]. Security & Privacy, 2007, 5(3): 62-65.

[17] WILLEMS C, HOLZ T, FREILING F. Toward automated dynamic malware analysis using CW Sandbox[J]. Security & Privacy, 2007, 5(2): 32-39.

[18] YAN Wei, ZHANG Zheng, ANSARI N. Revealing packed malware[J]. Security & Privacy, 2008, 6(5): 65-69.

[19] ZHANG Xiaosong, PAN Xiaohui, LONG Xiaoshu. Analysis of virtual machine applied to malware detection system[C]// International Symposium on Information Engineering and Electronic Commerce. Ternopil, Ukraine, 2009: 290-294.

[20] WANG Cheng, PANG Jianmin, ZHAO Rongcai, et al. Malware detection based on suspicious behavior identification [C]//Proceedings of the 2009 First International Workshop on Education Technology and Computer Science. Washington, DC, USA: IEEE Computer Society, 2009, 2: 198-202.

[21] HOFMEYR S A, FORREST S, SOMAYAJI A. Intrusion detection using sequences of system calls[J]. Journal of Computer Security, 1998, 6(3): 151-180.

[22] SCHULTZ M G, ESKIN E, ZADOK E, et al. Data mining methods for detection of new malicious executables[C]// Proceedings of the IEEE Symposium on Security and Privacy. Oakland, USA, 2001: 38-49.

[23] Cygnus. GNU Binutils Cygwin[EB/OL]. [2012-09-16]. http://sourceware.cygnus.com/cygwin.

[24] MILLER P. Hexdump[EB/OL]. [2012-09-16]. http://miller.emu.id.au/ pmiller/software/hexdump/.

[25] KOLTER J Z, MALOOF M A. Learning to detect malicious executables in the wild[C]//Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Seattle, USA, 2004: 470-478.

[26] KOLTER J Z, MALOOF M A. Learning to detect and classify malicious executables in the wild[J]. Journal of Machine Learning Research, 2006, 7: 2721-2744.

[27] REDDY D K S, PUJARI A K. N-gram analysis for computer virus detection[J]. Journal of Computer Virol, 2006, 2(3): 231-239.

[28] LI W J, WANG K, STOLFO S J, et al. Fileprints: identifying filetypes by N-gram analysis[C]//Proceedings of the 6th IEEE Systems, Man, and Cybernetics Information Assurance Workshop. Piscataway, USA: IEEE Press, 2005: 64-71.

[29] STOLFO S J, WANG K, LI W J. Towards stealthy malware detection [M]//CHRISTODORESCU M, JHA S, MAUGHAN D. Advances in Information Security. [S. l.]: Springer, 2007: 231-249.

[30] LI W J, STOLFO S J, STAVROU A, et al. A study of malcode-bearing documents [C]//International Conference on Detection of Intrusions & Virus, and Vulnerability Assessment (DIMVA). Lucerne, Switzerland, 2007: 231-250.

[31] SULAIMAN A, RAMAMOORTHY K, MUKKAMALA S, et al. Disassembled code analyzer for malware (DCAM) [C]//Proceedings of the IEEE International Conference on Information Reuse and Integration. Las Vegas, USA, 2005: 398-403.

[32] HENCHIRI O, JAPKOWICZ N. A feature selection and evaluation scheme for computer virus detection[C]//Sixth International Conference on Data Mining. Hong Kong, China, 2006: 891-895.

[33] KARNIK A, GOSWAMI S, GUHA P. Detecting obfuscated viruses using cosine similarity analysis[C]//Proceedings of the First Asia International Conference on Modeling & Simulation. Phuket, Thailand, 2007: 165-170.

[34] YE Yanfang, JIANG Qingshan, ZHUANG Weiwei. Associative classification and post-processing techniques used for malware detection[C]//2nd International Conference on Anti-Counterfeiting, Security and Identification. Guiyang, China, 2008: 276-279.

[35] YE Yanfang, WANG Dingding, LI Tao, et al. IMDS: intelligent malware detection system[C]//Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Jose, USA, 2007: 1043-1047.

[36] YE Yanfang, WANG Dingding, LI Tao, et al. An intelligent PE-malware detection system based on association mining[J]. Journal in Computer Virology, 2008, 4(4): 323-334.

[37] TABISH S M, SHAFIQ M Z, FAROOQ M. Malware detection using statistical analysis of byte-level file content [C]//Proceedings of the ACM SIGKDD Workshop on Cyber Security and Intelligence Informatics. Paris, France, 2009: 23-31.

[38] TREADWELL S, ZHOU M. A heuristic approach for detection of obfuscated malware [C]//IEEE International Conference on Intelligence and Security Informatics. Dallas, USA, 2009: 291-299.

[39] YE Yanfang, LI Tao, JIANG Qingshan, et al. CIMDS: adapting post-processing techniques of associative classification for virus detection [J]. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 2010, 40(3): 298-307.

[40] ZOLKIPLI M F, JANTAN A. A framework for malware detection using combination technique and signature generation [C]//Proceedings of the 2010 Second International Conference on Computer Research and Development. Kuala Lumpur, Malaysia, 2010: 196-199.

[41] KOMASHINSKIY D, KOTENKO I. Malware detection by data mining techniques based on positionally dependent features[C]//18th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP). Pisa, Italy, 2010: 617-623.

[42] FENG Shaorong, HAN Zhixue. An incremental associative classification algorithm used for malware detection[C]//2nd International Conference on Future Computer and Communication (ICFCC). Wuhan, China, 2010, 1: 757-760.

[43] MUHAYA F B, KHAN M K, XIANG Y. Polymorphic malware detection using hierarchical hidden Markov model [C]//IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing (DASC). Sydney, Australia, 2011: 151-155.

[44] SHANKARAPANI M K, RAMAMOORTHY S, MOVVA R S, et al. Malware detection using assembly and API call sequences[J]. Journal in Computer Virology, 2011, 7 (2): 107-119.

[45] RAVI C, MANOHARAN R. Malware detection using windows API sequence and machine learning[J]. International Journal of Computer Applications, 2012, 43(17): 12-16.

[46] HAN K S, KIM I K, IM E G. Detection methods for malware variant using API call related graphs[C]//International Conference on IT Convergence and Security. Suwon, Korea, 2012: 607-611.

[47] FORREST S, HOFMEYR S A, SOMAYAJI A, et al. A sense of self for uix processes[C]//Proceedings of IEEE Symposium on Security and Privacy. Oakland, USA, 1996: 120-128.

[48] KIM J, BENTLEY P. Towards an artificial immune system for network intrusion detection: an investigation of clonal selection with a negative selection operator [C]//2001 IEEE Congress on Evolutionary Computation. Seoul, Korea, 2001: 1244-1252.

[49] MATZINGER P. The danger model: a renewed sense of self[J]. Science, 2002, 296(5566): 301-305.

[50] LEE H, KIM W, HONG M. Artificial immune system against viral attack[C]//International Conference on Computational Science 2004. Krakow, Poland, 2004: 499-506.

[51] EDGE K S, LAMONT G B, RAINES R A. A retrovirus inspired algorithm for virus detection & optimization[C]// Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation. Seattle, USA, 2006: 103-110.

[52] LI Zhou, LIANG Yiwen, WU Zejun, et al. Immunity based virus detection with process call arguments and user feedback[C]//International Conference on Bio-Inspired Models of Network, Information and Computing Systems. Budapest, Hungary, 2007: 57-64.

[53] GONZALEZ F, DASGUPTA D. Anomaly detection using real-valued negative selection[J]. Journal of Genetic Programming and Evolvable Machines, 2003, 4(4): 383-403.

[54] BALACHANDRAN S, DASGUPTA D, NINO F, et al. A general framework for evolving multi-shaped detectors in

negative selection[C]//Proceedings of the IEEE Symposium Series on Computational Intelligence. Honolulu, USA, 2007: 401-408.

[55] LI Tao. Dynamic detection for computer virus based on immune system [J]. Science China Series F: Information Sciences, 2009, 39(4): 422-430.

[56] HARMER P K, WILLIAMS P D, GUNSCH G H, et al. An artificial immune system architecture for computer security applications [J]. IEEE Transactions on Evolutionary Computation, 2002, 6(3): 252-280.

[57] MARHUSIN M F, CORNFORTH D, LARKIN H. Malicious code detection architecture inspired by human immune system [C]//Proceedings of the 2008 Ninth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing. Phuket, Thailand, 2008: 312-317.

[58] GONG Tao. Unknown non-self detection & robustness of distributed artificial immune system with normal model [C]//7th World Congress on Intelligent Control and Automation. Chongqing, China, 2008: 1444-1448.

[59] ZHANG Yu, LI Tao, QIN Renchao. A dynamic immunity-based model for computer virus detection[C]//2008 International Symposiums on Information Processing (ISIP). Moscow, Russia, 2008: 515-519.

[60] QIN Renchao, LI Tao, ZHANG Yu. An immune inspired model for obfuscated virus detection [C]//International Conference on Industrial Mechatronics and Automation. Chengdu, China, 2009: 228-231.

[61] ZENG Jie, LI Tao. A novel computer virus detection method from ideas of immunology [C]//International Conference on Multimedia Information Networking and Security. Wuhan, China, 2009: 412-416.

[62] AL D E. Metamorphic viruses detection using artificial immune system[C]//International Conference on Communication Software and Networks. Macau, China, 2009: 168-172.

[63] ZHANG Chenggong, YI Zhang. A danger theory inspired artificial immune algorithm for on-line supervised two-class classification problem [J]. Neurocomputing, 2010, 73(7): 1244-1255.

[64] ZHU Yuanchun, TAN Ying. A danger theory inspired learning model and its application to spam detection [C]//International Conference on Swarm Intelligence. Chongqing, China, 2011: 382-389.

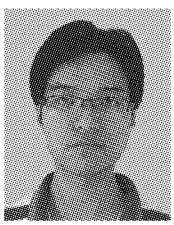
[65] ZHANG Pengtao, TAN Ying. A danger feature based negative selection algorithm[C]//International Conference on Swarm Intelligence. Shenzhen, China, 2012: 291-299.

[66] Computational Intelligence Laboratory of Peking University. CILPKU08 Dateset[EB/OL]. [2012-09-16]. http://www.cil.pku.edu.cn/resources/.

[67] ZHANG Pengtao, WANG Wei, TAN Ying. A malware detection model based on a negative selection algorithm with penalty factor[J]. Scientia Sinica Informationis, 2010, 53(12): 2461-2471.

**About the authors:**

TAN Ying (M'98, SM'02), male, born in 1964, received his BS in 1985, MS in 1988, and his PhD in signal and information processing from Southeast University, Nanjing, China, in 1997. Since then, he became a postdoctoral fellow then an associate professor at University of Science and Technology of China. He worked with the Chinese University of Hong Kong in 1999 and in 2004—2005. He was elected out of 100 talent program of the Chinese Academy of Science in 2005. Now, he is a full professor, advisor for Ph. D. candidates at the Key Laboratory of Machine Perception (Ministry of Education), Peking University, and Department of Machine Intelligence, EECS, Peking University, and he is also the head of Computational Intelligence Laboratory (CIL) of Peking University. He has authored or coauthored more than 200 academic papers in refereed journals and conferences and several books and book chapters. His current research interests include computational intelligence, artificial immune system, swarm intelligence and data mining, signal and information processing, pattern recognition, and their applications. He is an associate editor of IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics and an associate editor of the International Journal of Swarm Intelligence Research and the IES Journal B: Intelligent Devices and Systems. He is a member of the advisory board of the International Journal of Knowledge-Based and Intelligent Engineering Systems and of the editorial board of the Journal of Computer Science and Systems Biology and Applied Mathematical and Computational Sciences. He is also the editor of Springer Lecture Notes on Computer Science, LNCS 5263, 5264, 6145, 6146, 7331 and 7332, and the guest editor of several referred journals, including Information Science, Soft Computing, Neurocomputing, and the International Journal of Artificial Intelligence, etc. He was the recipient of a number of academic and research achievement awards from his country and universities due to his outstanding contributions and distinguished works, including the 2009 National Natural Science Prize of China. He was the general chairs of International Journal on Swarm Intelligence (ICSI 2010, ICSI 2011, ICSI 2012, and ICSI 2013) and the program committee chairs of ISNN2008, ICACI2013, ICIST2013, etc.



ZHANG Pengtao, male, born in 1986, received a Bachelor of Science in Computer Science from Dalian University of Technology, Liaoning, China, in 2008. He is currently majoring in Computer Science and working towards the PhD degree at the Key Laboratory of Machine Perception (Ministry of Education) and Department of Machine Intelligence, EECS, Peking University, Beijing. His research interests include artificial immune system, intelligent information processing algorithm, computer information security, pattern recognition, machine learning and data mining.