UNIVERSITÉ DU QUÉBEC

THESIS

SUBMITTED TO

THE UNIVERSITÉ DU QUÉBEC À CHICOUTIMI

IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE

MASTER'S DEGREE IN ADMINISTRATION

BY
PAN-MING

TITLE OF THESIS

Research on Detecting Mechanism for Trojan horse Based on PE file

MAY 2009

Bibliothèque
Paul-Émile-Boulet

UQAC
Université du Québec
à Chicoutimi

Mise en garde/Advice

Afin de rendre accessible au plus grand nombre le résultat des travaux de recherche menés par ses étudiants gradués et dans l'esprit des règles qui régissent le dépôt et la diffusion des mémoires et thèses produits dans cette Institution, **l'Université du Québec à Chicoutimi (UQAC)** est fière de rendre accessible une version complète et gratuite de cette œuvre.

Motivated by a desire to make the results of its graduate students' research accessible to all, and in accordance with the rules governing the acceptation and diffusion of dissertations and theses in this Institution, the **Université du Québec à Chicoutimi (UQAC)** is proud to make a complete version of this work available at no cost to the reader.

L'auteur conserve néanmoins la propriété du droit d'auteur qui protège ce mémoire ou cette thèse. Ni le mémoire ou la thèse ni des extraits substantiels de ceux-ci ne peuvent être imprimés ou autrement reproduits sans son autorisation.

The author retains ownership of the copyright of this dissertation or thesis. Neither the dissertation or thesis, nor substantial extracts from it, may be printed or otherwise reproduced without the author's permission.

ABSTRACT

As malicious programs, Trojan horses have become a huge threat to computer networks security. Trojan horses can easily cause loss, damage or even theft of data because they are usually disguised as something useful or desirable, and are always mistakenly activated by computer users, corporations and other organizations. Thus, it is important to adopt an effective and efficient method to detect the Trojan horses, and the exploration of a new method of detection is of greater significance.

Scientists and experts have tried many approaches to detecting Trojan horses since they realized the harms of the programs. Up to now, these methods fall mainly into two categories [2]. The first category is to detect Trojan horses through checking the port of computers since the Trojan horses send out message through computer ports [2]. However, these methods can only detect the Trojan horses that are just working when detected. The second class is to detect Trojan horses by examining the signatures of files [2][19], in the same way as people deal with computer virus. As new Trojan horses may contain unknown signatures, methods in this category may not be effective enough when new and unknown Trojan horses appear continuously, sending out unknown signatures that escape detection.

For the above-mentioned reasons, without exception, there are limitations in the existing methods if the un-awakened and unknown Trojan horses are to be detected. This thesis proposes a new method that can detect un-awakened and unknown Trojan horses- the detection by using of a file's static characteristics. This thesis takes PE file format as the object of the research, because approximately 75% of personal computers worldwide are installed the Microsoft Windows [4], and that Trojan horses usually exist as a Portable Executable (PE) file in the Windows platform. Based on the (PE) file format, my research gets all the static information of each part of PE file which is characteristic of a file. Then, this static information is analyzed by the intelligent information processing techniques. Next, a detection model is established to estimate whether a PE file is a Trojan horse. This model can detect the unknown Trojan horses by analyzing static characteristics of a file. The information that is used to verify detecting model is new and unknown to the detecting model; in other words, the information is not used during the training of the model.

The thesis is organized as follows. First, this thesis discusses the limitations of traditional detection techniques, related works of research, and a new method to detect

Trojan horse based on file's static information. Second, the thesis focuses on the research of the Trojan horse detecting models, covering the extracting of the static information from PE file, choice of intelligent information processing techniques, and setting up the Trojan horse detecting model. Lastly, the thesis discusses the direction of future research in this field.

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to my Research Supervisor Fan-Xiumei who helped and guided me towards my academic as well as my professional success. This thesis would also not be successful without my Research Associate Director Dai-Min who helped me to analyze project and solve problems. She told me how to think and how to solve the problem. She gave me many useful advices to help me complete my thesis successfully. I would also want to thank my classmates who help me to find some useful information and encourage me when I am depressed. I would also like to thank my parents for their unending support and eagerness to love.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Trojan horses have become a huge threat to the computer networks security because they can easily cause loss, damage or even theft of data. Since people realized the harm that Trojan horses bring, they have tried many ways to detect these malicious programs. This chapter begins with a brief introduction of Trojan horses and the harms they bring. The current approaches that people take to detect Trojan horses and their limitations will then be discussed in this chapter. Finally, a new method based on the static characteristics of PE file is proposed to detect Trojan horses, which can overcome the limitations of the previous ways.

1.1 Introduction of Trojan horse

The term, Trojan horse, comes from a Greek story of the Trojan War, in which the Greeks give a giant wooden horse to their foes, the Trojans, ostensibly as a peace offering. But after the Trojans drag the horse inside their city walls, Greek soldiers sneak out of the horse's hollow belly and open the city gates, allowing their compatriots to pour in and capture Troy.

1.1.1 Trojan horses and its harms

A Trojan horse program is a piece of computer software that includes hidden functionality in addition to its declared functionality. They will typically have a server that is installed on the victim computer and a client on the attacker's computer [17]. The server listens for commands sent from the client and responds by sending data back to the client. Trojan horse enables an attacker to bypass existing security measures to access a computer, thereby Trojan horse provide a "backdoor" or instill harmful pieces of software into the computer and provides any functionality that a client/server program operating in a networked environment can provide. The reason for this is that the server runs on the victim's computer and will typically run as administrator or as a highly privileged user. Listed below are the types of actions that a Trojan horse might take [17]:

- Log the victim's keystrokes (including passwords)
- Render the victim's screen on the attacker's computer
- Monitor network traffic on the victim's network
- Hijack TCP sessions involving the victim's computer
- Record conversations via the victim computer's microphone or control a web cam
- Send files from the victim's computer to the attacker
- Use the computer as a platform for attacks on other computers
- Modify data on the victim's computer

Unlike viruses and worms, a Trojan horse is not capable of propagating itself [17], but Trojan horse successfully spreads itself by new approaches. It is often delivered to a victim through an email message where it masquerades as an image or joke, or by a

malicious Web site that installs the Trojan horse on a computer through vulnerabilities in the Web browser software such as Microsoft Internet Explorer. Moreover, Trojans are the first stage of an attack and their primary purpose is to stay hidden while downloading and installing more sophisticated threats [17]. After it is installed, the Trojan horse lurks silently on the infected machine, invisibly carrying out its misdeeds such as downloading spy ware while victims continues on with their Web surfing or other normal activities.

In summary, Trojan horse is clever at hiding itself and spreading itself by various approaches and these characteristics bring difficulty to detect Trojan horse by using traditional detecting methods.

1.1.2 Traditional ways for detecting Trojan horses

As Trojan horses have become a huge security threat to computer network, it is of great importance to detect the Trojan horse efficiently for the network security. After people realized the harms of Trojan horse, they try every means to detect Trojan horse. Up to now, the approaches to detecting the Trojan horse can be grouped into two categories [2].

The first category is to detect Trojan horses through checking the port of computers since the Trojan horses send out message through computer ports [2]. However, these methods can only detect the Trojan horses that are just working when detected.

The second class is to detect Trojan horses by examining the signatures of files [2][19], in the same way as people deal with a computer virus. As new Trojan horses may contain unknown signatures, methods in this category may not be effective enough when new and unknown Trojan horses appear continuously, sending out unknown signatures that

escape detection.

For the above-mentioned reasons, without exception, there are limitations in the existing methods if the un-awakened and unknown Trojan horses are to be detected. This thesis proposes a new method that can detect un-awakened and unknown Trojan horses--the detection by using of a file's static characteristics.

1.2 Detecting Trojan horse based on file's static characteristics

One characteristic of Trojan horse is that it can hide successfully and spreads itself by various approaches, which brings troubles in detecting Trojan horses. The traditional detecting methods based on the file's dynamic characteristics cannot resolve these problems in detecting. So we propose a new method to detect Trojan horse, and this method can detect all the Trojan horse.

Now, many researchers try new detection techniques to solve the problem of network security. A group of researchers use file's static information to detect the Virus and malicious program, and finally they got satisfactory results [4] [6] [15] [16].

1.2.1 Related work

Some researches use a data mining approach based on a file's static characters to detect the Virus and malicious programs, and they got a recognition rate of over 80%. [4][6] One of the researches is "Learning to Detect New and Unknown Malicious Programs". They proposed the Bayesian method to differentiate between benign programs and

malicious programs. In their experiments, they detected 81.54% of previously unknown malicious programs with a 0.96% false-positive rate [4]. In another research "Neural Networks for Computer Virus Recognition", the researchers were able to identify 80–85% of unknown boot sector viruses successfully with a very low false positive rate (<1%) [6].

Some researchers conduct a static analysis of executables of a file. For example, the research entitled "Static Analysis of Executables to Detect Malicious Patterns" presents a unique viewpoint on malicious code detection [15]. They tested the resilience of three commercial virus scanners against code-obfuscation attacks. The results were surprising: the three commercial virus scanners could be subverted by very simple obfuscation transformations! They present an architecture for detecting malicious patterns in executables that is resilient to common obfuscation transformations. Experimental results demonstrate the capability of their prototype tool, SAFE (a Static Analyzer For Executables) [15].

Another research aims to use the Windows Portable Executable (PE) file to determine whether malicious code has been inserted into an application after compilation [16].

However, Trojan horse is characterized by its ability to transform [2][19], which differentiates it from various other malicious programs, so the above-mentioned approaches, ie. "the malicious code detection"[15], and the approach to determine whether malicious code has been inserted[16] are not effective enough to detect Trojans with numerous new Trojans keep appearing day after day.

We want to detect Trojans based on a file's static characters. Considering about 75%

of world's personal computer install the Microsoft Windows [4] and Trojan horse usually exists as a Portable Executable (PE) file in Windows platform, the PE file format is accepted as the object of our research. The aim of our research is to differentiate the Trojan horse files and normal files based on PE format. We assume if there are differences between the static characters of a Trojan file and the static characters of a normal file, all the Trojan horses can be detected with this method.

1.2.2 To detect Trojan horse by using file's static characteristics

In this thesis, a new detecting method is proposed which is based on file's static characteristics. Intelligent information processing techniques are used to analyze these static information. Then, a detection model is established to estimate whether a PE file is a Trojan horse or not, which can also detect the unknown Trojan horse by analyzing file's static characteristics. The information used to verify detecting model is new and unknown to our model, in other words, the information is not used during training the model.

1.3 Thesis organization

The rest of this thesis is organized into the following chapters: Chapter 2 introduces the PE format , and presents how to extract the static information form files according to PE

Trojan horses detecting and use the neural network as our method to analyze the static information. Chapter 4 presents all the details of the problems in setting up Trojan horse detecting models. Because the static information involves three parts, we set up a respective

model as a classifier for each part, and combine the results of these classifiers to judge whether the file is a Trojan file or not. Chapter 5 deals with the details of judging the Trojan horse. Chapters 6 summarize the thesis.

CHAPTER 2

EXTRACTS THE STATIC INFORMATION FROM PE FILE

Now that we have decided to use file's static characters to detect a Trojan horse, we need to get the static information from files. There are many ways to get the static characters of a file. Considering about 75% of world's personal computer install the Microsoft Windows [4] and Trojan horse usually exists as a Portable Executable (PE) file in Windows platform, the PE file format is accepted as the object of our research.

2.1 Introduction of the PE file

The Portable Executable (PE) file format is a new executable file format, which was introduced by the Windows NT™ (version 3.1) operating system [12] [20].

The PE file format begins with an MS-DOS header, a real-mode program stub, and a PE file signature. Immediately following is a PE file header and optional header. Beyond that, all the section headers appear, followed by all of the section bodies [20]. The structure of PE file is shown in Figure 1.

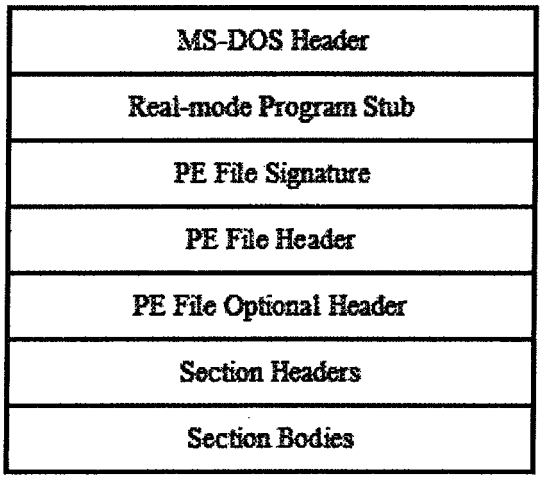

Figure 1. The structure of PE file

2.2 Information extracting and its organization

In Microsoft Portable Executable and Common Object File Format Specification, the description of the PE format is: "the entire format of PE file consists of File header followed by all of the section headers, and finally, all of the section bodies. And The File header consists of an MS-DOS MZ header, the real-mode stub program, the PE file signature, the PE file header, and the PE optional header." [14] So the PE format can be divided into three parts: File header, section header, and section body, the structure of PE format is shown in Figure2.

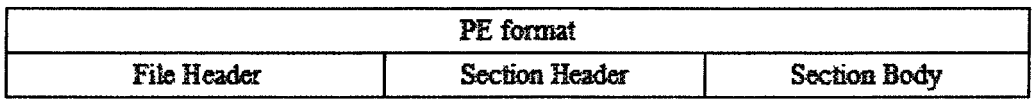

Figure 2. The structure of PE format

"The memory-mapped file is one of the coolest features of Windows NT." [20] Memory-mapped files permit the use of simple pointer dereferencing to access the data contained within the file. So the static information from each field can be extracted by using memory-mapped files for accessing data in PE files.

The information in the fields of each part of PE format will be extracted. In total, there are 56 fields in the File header and there are 10 fields in the Section header. In the section body, an application for Windows NT typically has nine predefined sections [20]. Some applications do not need all of these sections, while others may define still more sections to suit their specific needs [20]. If we extracted the static information of each section body, the workload would be very large. So we decide to extract the static information of two of the nine section bodies, because it is generally believed that if one section body can be processed, the other Section bodies can be processed likewise. The selected section bodies are import section (.idata) and export section (.edata), with 8 fields in the former and 14 in the latter one. The statistical information is shown in Table1.

Place in the PE	Trojan sample	Normal sample	Number of fields
FileHeader	TrojanHeader	NormalHeader	56
SectionHeader	TrojanSectionHeader	NormalSectionHeader	10
ImportSection	TrojanImportDirectory	NormalImportDirectory	8
ExportSection	TrojanExportDirectory	NormalExportDirectory	11
	TrojanExportFunction	NormalExportFunction	3

Table1. Statistical Information

Then we put the information in SQL server database. The connections among the tables of database are shown in Figure 3 and Figure 4.

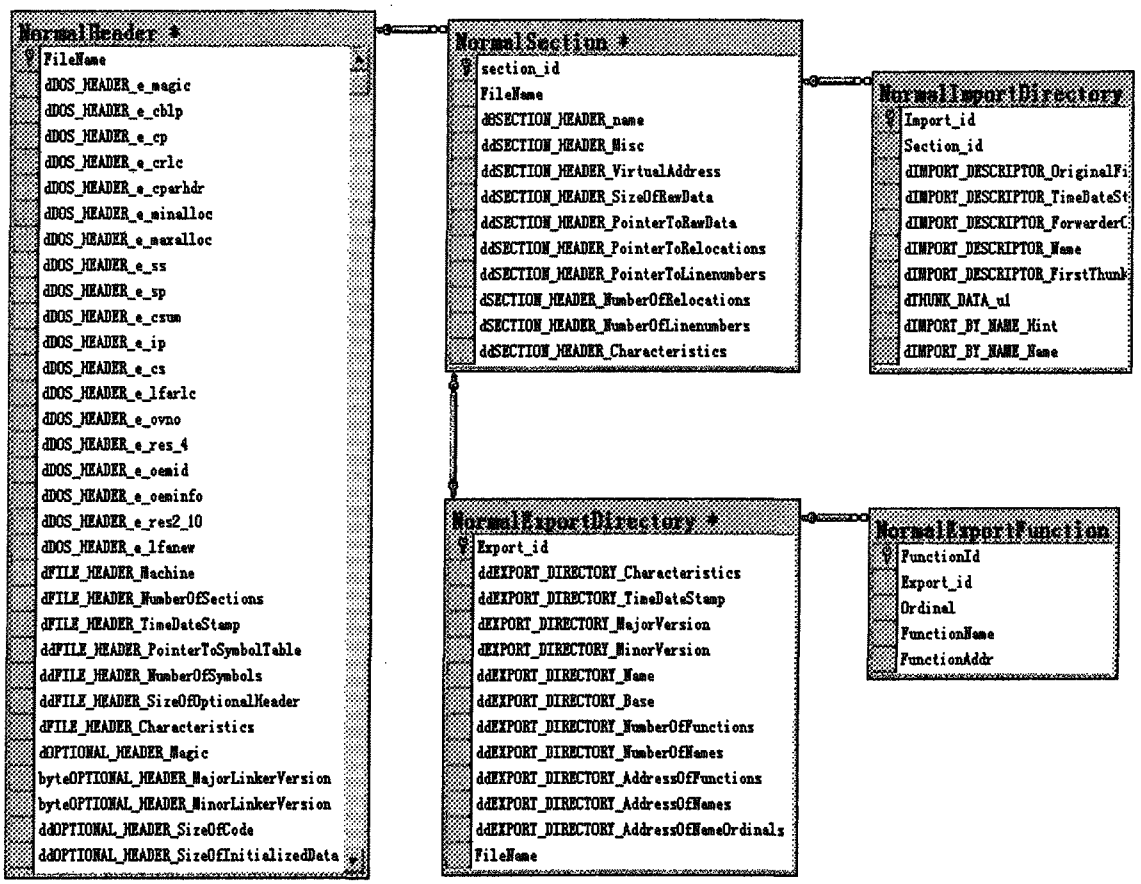

Figure 3. The connections of normal file

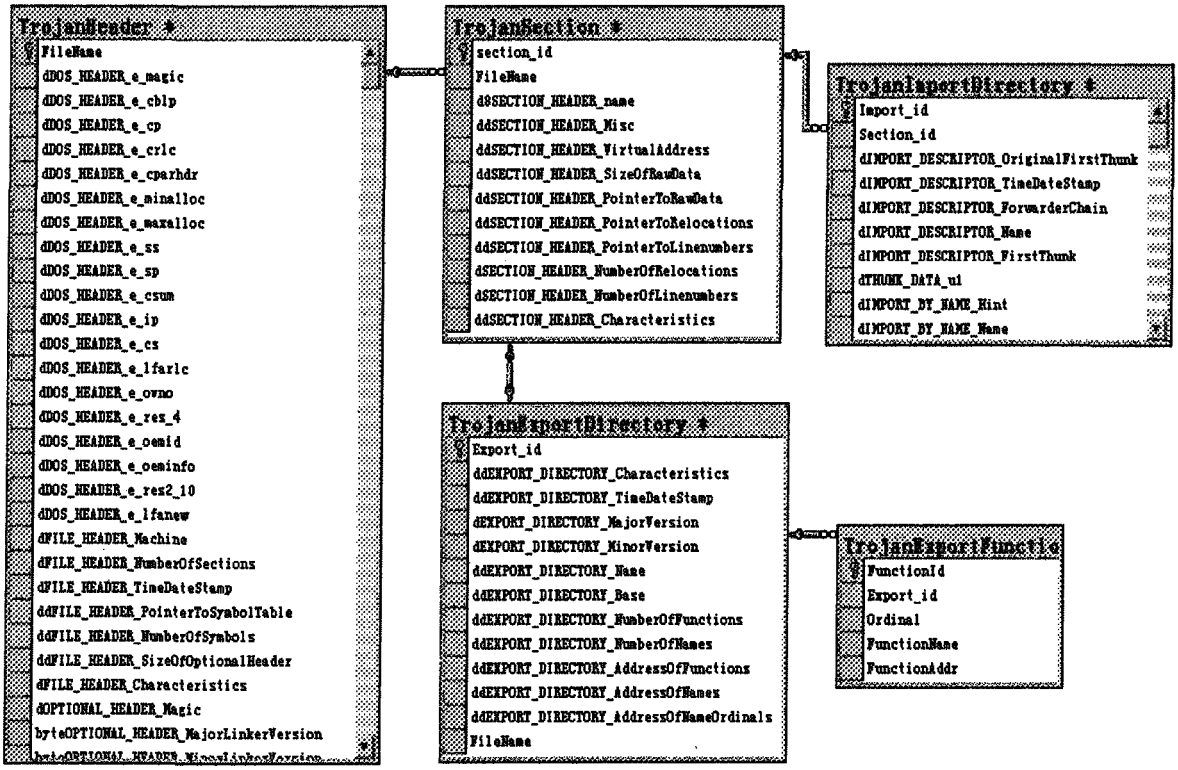

Figure 4. The connections of Trojan file

Note:

The total number of samples is 299 , with 157 Trojan samples and 142 normal samples in it.

The prefix is Normal means it is the information of Normal sample.

The prefix is Trojan means it is the information of Trojan sample.

The postfix of table is Header means it belongs to the structure of Dos Header, PE Header or Optional Header.

The postfix of table is Section means it belongs to the structure of Section Table.

The postfix of table is Import Directory means it belongs to the structure of Import Table.

The postfix of table is Export Directory means it belongs to the structure of Export Table.

The postfix of table is Export Function means it belongs to the structure of Export Function.

CHAPTER 3

WHAT IS THE TROJAN HORSES DETECTING PROBLEM

3.1 Is it a classification problem?

We use static information to judge whether the file is Trojan file or not. In other words, we want to differentiate a Trojan file from a normal file based on the file's static information. To differentiate a Trojan file from a normal file means to classify the files, so the true nature of Trojan horses detecting problem is a classification problem.

3.2 The methods for solving the problem

There are many algorithms to solve the problem of classification: Decision Tree, Neural Networks, Bayesian, Genetic Algorithms and so on [7] [24].

3.2.1 The methods of classification

1. Decision Tree

A decision-tree learning algorithm approximates a target concept using a tree representation, where each internal unit corresponds to an attribute, and every terminal unit corresponds to a class [24]. Decision trees are powerful and popular tools for classification and prediction. The strengths of decision tree methods are [24]:

- Decision trees are able to generate understandable rules.
- Decision trees perform classification without requiring much computation.
- Decision trees are able to handle both continuous and categorical variables.
- Decision trees provide a clear indication of which fields are most important for prediction or classification.

The weaknesses of decision tree methods are [24]:

- Error-Prone with Too Many Classes.
- Computationally Expensive to Train.
- Trouble with Non-Rectangular Regions.

2. Neural Networks

Strengths of Artificial Neural Networks are [24]:

- Neural Networks Are Versatile.
- Neural Networks Can Produce Good Results in Complicated Domains.
- Neural Networks Can Handle Categorical and Continuous Data Types.

Weaknesses of Artificial Neural Networks are [24]:

- All Inputs and Outputs Must Be Massaged to [0.1].
- Neural Networks Cannot Explain Results.

There are some other algorithms that can solve the problem of classification, such as Bayesian, Genetic Algorithms. Because the data of the samples is rather large, and it is impossible to find out the relations directly, we select the neural network to solve the problem as the neural network can produce good results in complicated domains [24].

Moreover, the theory of neural network is mature, and there are rich experiences summarized.

3.2.2 Trojans detecting based on BP network

The neural network with probably the closest analogy to the human brain is the Multi-layer Perceptron or MLP for short. This network has found a wide range of applications[7]. It uses supervised learning, which means that input and output data are required during the training phase. The most common training algorithm for the MLP is Backpropagation (BP)[7].

Introduction of the Backpropagation Algorithm

BACKPROPAGATION(training_examples,η, n_{in}, n_{out}, n_{hidden}) [24]

Each training example is a pair of the form $\langle \vec{x}, \vec{t} \rangle$, where $\vec{x}$ is the vector of network input values, and $\vec{t}$ is the vector of target network output values.
$\eta$is the learning rate (e.g., .05). n_{in} is the number of networks, n_{hidden} is the number of units in hidden layer, and n_{out} the number of output units.
The input from unit i into unit j is denoted x_{ji} and the weight from unit i to unit j is denoted by w_{ji}.

- Create a feed-forward network with n_{in} inputs, n_{hidden} hidden units, and n_{out} output units.
- Initialize all the weights to small random values (e.g., between -0.05 and 0.05)
- Until termination condition is met, Do
    - For each $\langle \vec{x}, \vec{t} \rangle$ in training_examples, Do

Propagate the input forward through the network:

1. Input the instance $\vec{x}$ to the network and compute the output o_u of every unit u in the network.

For example: create a feed-forward network with *2* inputs, *2* hidden units, and 1output unit. Initialize all the weights to small random values (e.g., between -0.05 and 0.05). So the output $o_u = x_{53} w_{53} + x_{54} w_{54}$ and $x_{53} = x\ w_{31} + x\ w_{32}$, $x_{54} = x\ w_{41} + x\ w_{42}$. As shown in Figure 5.

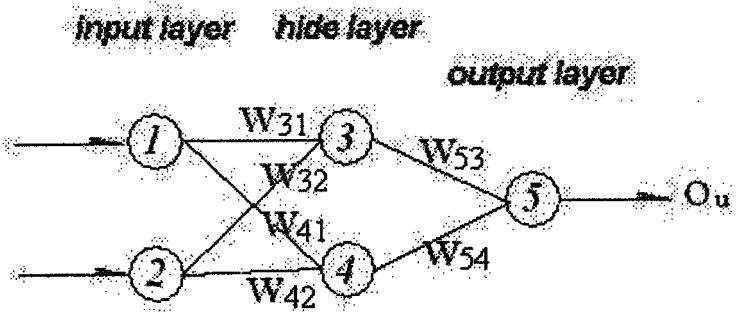

Figure 5. The example of Backpropagation Algorithm

Propagate the errors backward through the network:

2. For each output unit *k*, calculate its error term δ_k

$$\delta_k \leftarrow o_k(1-o_k)(t_k - o_k)$$

3. For each hidden unit *h*, calculate its error term δ_h

$$\delta_h \leftarrow o_h(1-o_h) \sum_{k \in outputs} w_{kh}\delta_k$$

4. Update each network weight w_{ji}

$$w_{ji} \leftarrow w_{ji} + \Delta w_{ji}$$

where $$\Delta w_{ji} = \eta \delta_j x_{ji}$$

The training process is shown in Figure 6.

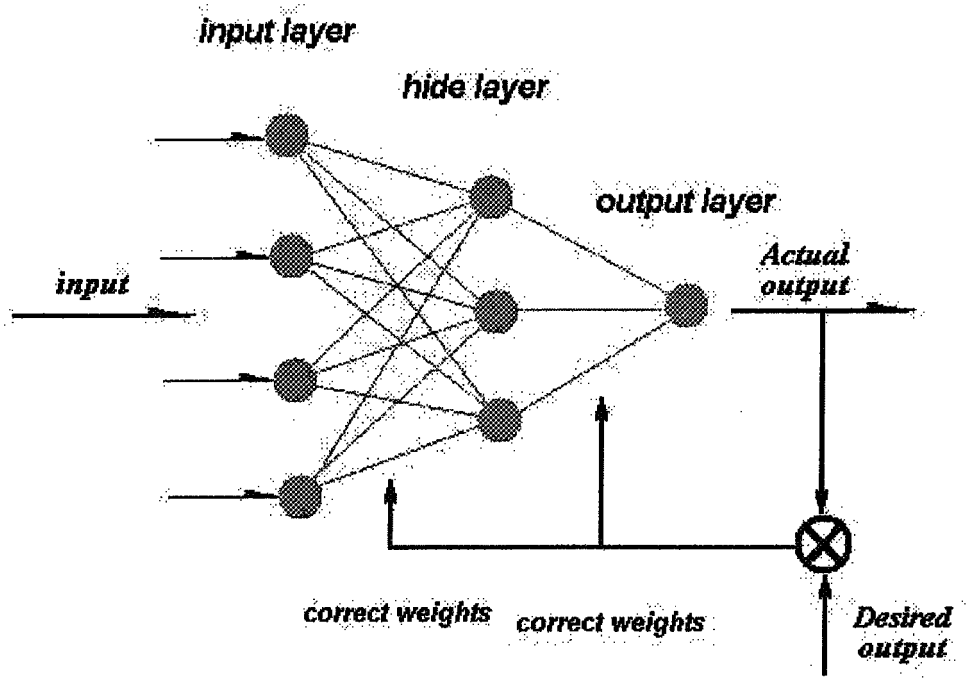

Figure 6. The Backpropagation Algorithm

CHAPTER 4

TROJAN HORSES DETECTING MODEL

The information extracted from PE file cannot be used into the model of neural network directly. The reasons are as follows:

First, there are many (more than one) sections in one PE file, and these sections are paratactic. Therefore, the structure of these multi-dimensions will bring difficulties in setting up the model of neural network.

Second, the values of all the inputs of the neural network must fall in the range of 0-1. But the types of attributes in the PE file are multiform. The type of some attributes is string and the lengths of them are inequable, thus some way must be found to solve the problem of normalization.

4.1 The difficulties in setting up models

According to chapter 2, we first organize the data in three parts, and then set up classifiers to different parts (one classifier for each part). Finally, we combine the results of classifiers to judge whether the file is a Trojan file or not. The process of our method is shown in Figure 7.

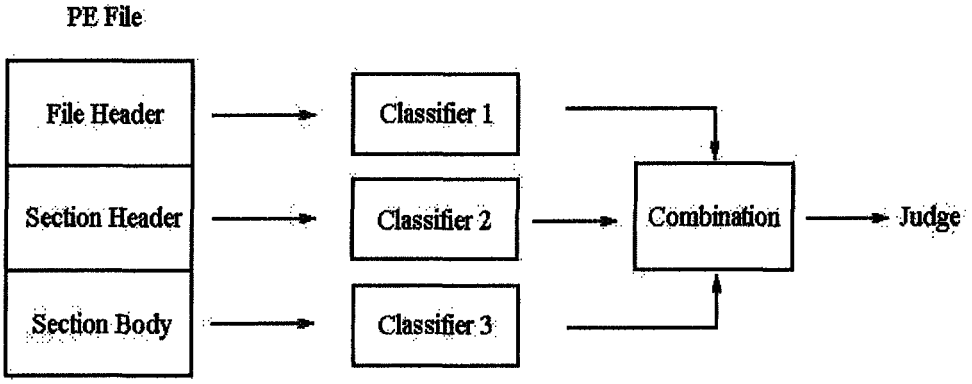

Figure7. The process of our method

The first problem mentioned at the beginning of this chapter can be explained as how to set up the classifiers for Section header. All the attributes of each Section header are the same and the numbers of Section headers in each file are different. According to "the attributes of each Section header are the same", the data of each Section header is viewed as an input data for classifier, and the desired output of the file that the Section header belongs is taken as the desired output of section header classifier. Then these data are used to train and test the classifier model. To solve the problem of "the numbers of Section headers in each file are different", the highest actual output of all the Section headers classifier in one file is used to judge whether the file is Trojan horse or not. If the classifier judges one Section header of a file to be a Trojan horse, the file is a Trojan horse. If the classifier judges all the Section headers of a file to be normal files, the file is a normal one. For example, there are three files in database, and the numbers of Section headers of file1 to file3 are 2, 3, and 4. And if the classifier judges that two Section headers of file1 are Trojan horses, one of the three Section headers of file2 is Trojan horse, and all the 4 Section headers of file3 are normal files, the file1 and file2 will be judged as Trojan horse, and file3

will be judged as normal file.

As to the second problem raised at the beginning of this chapter, the solution to it will be discussed in section 4.2.2.

4.2 The classifiers of the model

Owing to the complexity of the structure of the samples, we will do various experiments in our research. However, the workload would be too large if we set up a special model for each particular experiment and each model with a different structure. In order to simplify our workload and shorten the time of the experiment, we setup a universal training platform of neural network. And the structure of universal training platform can be changed easily to suit various experiments. The changeability includes the change of the numbers of units of input-layer and hide-layer, the usage and management of the sample space. We created the training platform of neural network in JAVA. It is based on the BP learning algorithm of neural network. There is one hide layer in the training platform. As the problem of Trojan horse detecting is a question of classification, the three layers neural network (single hide layer) can solve it [7].

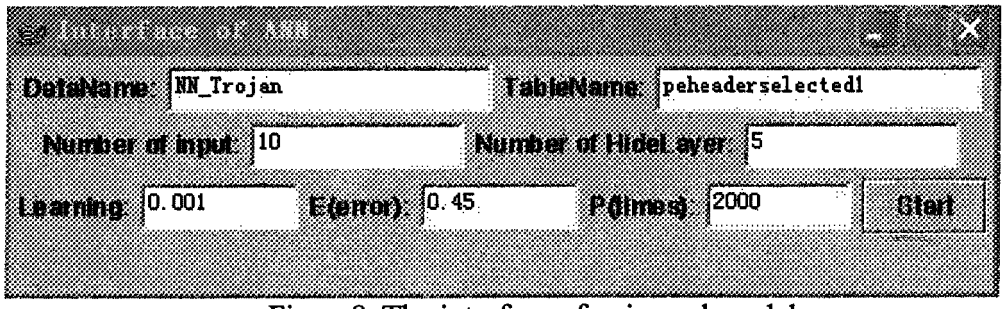

Figure8. The interface of universal model

4.2.1 The classifier for the File header

The PE header (ImageFileHeader,ImageOptionalHeader) mainly includes the three parts of DosHeader, PEHeader and OptionalHeader. There are altogether 56 fields in the File header, among which 31 will be insignificant to classifier model because they show no differences between the Trojan file and the normal file when analyzed. So the other 25 fields will be enough to set up classifier model. The selected 25 fields are shown in Table 2.

All the fields of File header must be processed before being put into classifier model. All the data in the fields (used to set up classifier model) from File header are hexadecimal numbers. According to the distributing of their values, the data will be processed in two ways: linearity and discretization.

If the distributing of the values is uniformity, the linearity way may be used to process them. For example, if the value of field is within the range of [a, b], the way of processing is:

$$\text{ToOne} = \frac{Value - a}{b - a}$$

When the distributing of the values is asymmetric, discretization way should be used to process them. Then we use user-defined function of database to implement the processing.

The ways to process the fields of File header as shown in Table 2:

Place in the PE	Field name	Normalization method
DosHeader	dDos_header_e_cblp	(dDos_header_e_cblp-60)/30
	dDos_header_e_cp	dDos_header_e_cp-2
	dDos_header_e_minalloc	dDos_header_e_minalloc/15
	dDos_header_e_ovno	dDos_header_e_ovno/26
	dDos_header_e_lfanew	(dDos_header_e_lfanew-128)/384
PEHeader	dFile_Header_NumberOfSections	(dFile_Header_NumberOfSections-1)/9
	dFile_Header_TimeDateStamp	discretization
	dFILE_HEADER_Characteristics	(dFILE_HEADER_Characteristics-258)/41100
Optional Header	byteOPTIONAL_HEADER_MajorLinkerVersion	(byteOPTIONAL_HEADER_MajorLinkerVersion-1)/6
	byteOptional_header_MinorLinkerVersion	byteOptional_header_MinorLinkerVersion/56
	ddOptional_Header_SizeOfUninitializedData	discretization
	ddOptional_Header_FileAlignment	(ddOptional_Header_FileAlignment-200)/800
	dOptional_Header_MajorOperatingSystemVersion	(dOptional_Header_MajorOperatingSystemVersion -1)/ 4
	dOptional_Header_MinorOperatingSystemVersion	dOptional_Header_MinorOperatingSystemVersion / 2
	dOptional_Header_MajorImageVersion	dOptional_Header_MajorImageVersion/7
	dOptional_Header_MinorImageVersion	dOptional_Header_MinorImageVersion/6
	dOptional_Header_MajorSubsystemVersion	(dOptional_Header_MajorSubsystemVersion-3)/2
	dOptional_Header_MinorSubsystemVersion	dOptional_Header_MinorSubsystemVersion/10
	ddOptional_Header_SizeOfHeaders	(ddOptional_Header_SizeOfHeaders-512) /3584
	ddOptional_Header_CheckSum	discretization
	dOptional_Header_SubSystem	(dOptional_Header_SubSystem-1)/2
	dOptional_Header_DllCharacteristics	dOptional_Header_DllCharacteristics/32768
	ddOptional_Header_SizeOfStackReserve	ddOptional_Header_SizeOfStackReserve/2000000
	ddOptional_Header_SizeOfStackCommit	ddOptional_Header_SizeOfStackCommit/20000
	ddOptional_Header_SizeOfHeapReserve	discretization

Table2. The ways to deal with the fields of file's PE header

We add a field "IsTrojan" into database. And the value of "IsTrojan" is the desired output of classifier model. There are three layers in classifier model: one input layer, one hide layer and one output layer. There is one unit in the output layer. The value of actual output will be corrected by error, and is taken as the output of classifier model. The value of output after correction is 0 or 1. For example, we set the error=0.45, if the range of value of

the actual output is [-0.45, 0.45], the value of output will be corrected to 0. And if the range of value of the actual output is [0.55, 1.45], the value of output will be corrected to1. If the value of output is 0, this file is to be judged as normal file, and if the value of output is 1, the file is a Trojan file.

In the input layer, the values of the 25 fields are the input of classifier model, so there are 25 units in the input layer.

In the hide layer, from the experienced formula:

Number of units in hide layer= $\sqrt{numberofunitininput * numberofunitinoutput}$ [7]

So the number of units in hide layer is about 5. Then we make experiments to establish the best structure of model.

Experiment parameters :the number of units in input layer is 25, the number of units in output layer is 1, the total number of samples is 299, and the number of samples to train the model is 200, error=0.45, learning=0.001, train times is 2000.

Parameters / Times	Number of units in hide layer	Recognition rate for train (average) (%)
1	2	47.5
2	3	45.5
3	4	42.5
4	5	45.25
5	6	47.5
6	7	48.75
7	8	47.75
8	9	47.5
9	10	47.5

Table3. Establish the best structure of model of file header

The entire model is not convergent. We make the following assumption to the main reasons.

First, the algorithm needs to be optimized.

Second, using all of the many attributes in this layer is not likely to produce a satisfactory result in setting up a model. So we should select some important attributes to set up the model in order to raise the Recognition rate of model.

Optimization of the BP learning algorithm

The back propagation (BP) algorithm is a systematic method for training multilayer neural networks. It has many drawbacks despite many successful applications of back propagation. As to many complex problems, it may require a long time to train the networks, and it may even not train at all. Long training time can be the result of the non-optimal parameters [7]. It is not easy to choose appropriate value of the parameters for a particular problem [7].

By now, there are many ways to optimize the BP learning algorithm, some of which will be selected to optimize classifier model. Now take the experiment of file's PE header as an example to introduce the improvement after optimizing the BP learning algorithm.

1. Adjust the initial weight of model

Adjust the initial weight to every nerve cell work at the place where the transfer function has the most sensitivity. The method is: firstly to adjust the weights of hide layer to an enough small value, and then to adjust the range of weights of output layer from [-1, +1]

to [-0.5, +0.5] [7].

Train the model after adjusting the initial weight. The experimental data as shown in following Table4:

Parameters / Times	Structure	Recognition rate for train (%) (Before optimize)	Recognition rate for train (%) (After optimize)
1	25-2-1	47.5	54.75
2	25-3-1	45.5	52.50
3	25-4-1	42.5	48.50
4	25-5-1	45.25	49.75
5	25-6-1	47.5	55.50
6	25-7-1	48.75	60.25
7	25-8-1	47.75	52.50
8	25-9-1	47.5	55.75
9	25-10-1	47.5	52.50

Table4. The experimental result of file's PE header after adjust the initial weight of model

2. Add the momentum

Trains the model after adding the momentum. The experimental data as shown in following Table 5:

Parameters / Times	Structure	Recognition rate for train (%) (Before optimize)	Recognition rate for train (%) (After optimize)
1	25-2-1	54.75	53.75
2	25-3-1	52.50	53.50
3	25-4-1	48.50	57.50
4	25-5-1	49.75	59.50
5	25-6-1	55.50	57.25
6	25-7-1	60.25	62.75
7	25-8-1	52.50	55.50
8	25-9-1	50.75	53.25
9	25-10-1	52.50	51.75

Table5. The experimental result of file's PE header after add momentum

From the experiment, the advanced capability of the neural network after being optimized is illustrated. But the recognition rate for training the model is lower. The highest

recognition rate for training the model is 62.75%. Then some more important attributes will be selected to set up models.

We use ID3 algorithm of decision tree to select the more important attributes. We figure out the information gain and entropy of the attributes, and select the attributes by comparing the information gain and entropy. Then we set up model with selected attributes that get the best result. After repeated experiments, 10 attributes are selected to setup classifier model.

To establish the best structure of model: Experiment parameters : the number of units in input layer is 10, number of units in output layer is 1, total number of samples is 299, number of samples to train the model is 200, error=0.45, learning=0.001, train times is 2000.

Parameters / Times	Structure	Recognition rate for train (%)
1	10-2-1	70.76
2	10-3-1	70.83
3	10-4-1	71.7
4	10-5-1	72.01
5	10-6-1	73.68
6	10-7-1	71.84
7	10-8-1	71.77
8	10-9-1	70.56

Table6. Establish the best structure of model of file's PE header

The highest recognition rate is "73.68", so the structure 10-6-1 is selected as the structure of classifier model. Then test the model.

Experiment parameters : The structure of model is 10-6-1, the total number of samples is 299, the number of samples to train the model is 200, the number of samples to

test the model is 99,error=0.45.

Parameters / Times	Learning	Train times	Recognition rate for train (%)	Recognition rate for test (%)
1	0.001	2000	81.50	42.42
2	0.001	1000	82.80	62.42
3	0.001	800	82.00	68.65
4	0.001	600	76.75	61.60
5	0.05	800	77.50.	67.64

Table7. The last results of model of file's PE header

Now the classifier of File header has been set up. The capability of model has become stronger after being optimized. Then the classifier for the other part will be set up.

4.2.2 The classifier for the Section header

There are fewer attributes in section header, and among them, only 7 values of attributes are unfixed, all the others being fixed.

In section header, the type of field named d8SECTION_HEADER_name is string. Its most length is 8. Now we come to the problem mentioned at the beginning of this chapter.

If we use the ASCII to solve the problem of normalization, then the value of 8 strings of one field is F (a1, a2, a3, a4, a5, a6, a7, a8), and the range of a_i is [0,255]. So the value of 8 strings must reach the precision of 256^8, it is very difficult for computer.

If we only select the fixed initiatory several strings, the different value of fields will become the same value after pretreatment, and the distributing cannot distribute uniformity. This may cause problem when we train the model and may even lead to failure for training model. So it is not suitable to use the ASCII to solve the problem of normalization.

Since the values of inputs must be in the area [0.1], and the value of probability

belongs to [0.1], the value of probability is counted as the value of field. And in this way, if there appear new names which are not included in our samples, its value can be set 0.5 (It is easy to understand from the point of view of probability.). Now that the second problem is solved, the next step is counting the value of probability:

At first, count:

NormalNum: the times of SectionName appear in normal file.

TrojanNum: the times of SectionName appear in Trojan file.

Then, count:

NormalTotal: the total number of normal file.

TrojanTotal: the total number of Trojan file.

Last, count the probability according to the following expressions:

$$\text{TrojanProb} = \frac{TrojanNum}{\frac{NormalNum * TrojanTotal}{NormalTotal} + TrojanNum}$$

The information of counting probability is shown in following Table 8:

SectionName	NormalNum	TrojanNum	TrojanProb	SectionName	NormalNum	TrojanNum	TrojanProb
<blank>	0	14	1	BSS	0	41	1
.adata	0	7	1	CODE	0	48	1
.aspack	0	11	1	DATA	0	43	1
.bss	0	5	1	EOOPONO	0	1	1
.edata	0	1	1	JXVWEGX	0	1	1
.idata	0	60	1	MCNILUW	0	1	1
.pec	0	1	1	OSQFOPP	0	1	1
.peco	0	5	1	pec1	0	1	1
.petite	0	2	1	rsrc	0	1	1
.rdata	2	66	0.97947	text	0	7	1
.sec	0	1	1	UPX0	0	34	1
.Shared	0	1	1	UPX1	0	34	1
.stab	0	3	1	UPX2	0	5	1
.stabstr	0	3	1	VDJCQCU	0	1	1
.data1	1	0	0	.reloc	75	55	0.51463
.tls	1	40	0.98300	.data	225	70	0.31026
.instanc	1	0	0	.text	226	65	0.29370
.orpc	1	0	0	.rsrc	227	138	0.46779
INIT	1	0	0				

Table 8. The information of counting probability of file's section's header

The ways of other field's normalization are shown in Table 9:

Field name	Normalization method
ddSECTION_HEADER_Misc	ddSECTION_HEADER_Misc /19688132.0
ddSECTION_HEADER_VirtualAddress	ddSECTION_HEADER_VirtualAddress /20004864.0
ddSECTION_HEADER_SizeOfRawData	ddSECTION_HEADER_SizeOfRawData /1339392.0
ddSECTION_HEADER_PointerToRawData	ddSECTION_HEADER_PointerToRawData /1343488.0
ddSECTION_HEADER_PointerToRelocations	ddSECTION_HEADER_PointerToRelocations /12800606.0
ddSECTION_HEADER_PointerToLinenumbers	ddSECTION_HEADER_PointerToLinenumbers /5152.0

Table 9. The ways to deal with the other fields of section's header

Now the classifier for the Section header will be set up. For the input layer: seven fields are established to setup model. The names of fields are shown in Table 5. For the hide layer: From the experienced formula:

Number of units in hide layer= $\sqrt{numberofunitininput * numberofunitinoutput}$ [7]

The number of units in hide layer is about 3. Then we make the experiments to establish the best structure of model.

Experiment parameters : the number of units in input layer is 7, the number of units in output layer is 1, the total number of samples is 1336, the number of samples to train the model is 1000, error=0.45, learning=0.001, train times is 2000.

Parameters / Times	Number of units in hide layer	Recognition rate for train (%) (average)
1	2	73.75
2	3	72.55
3	4	67.63
4	5	75.94
5	6	69.50
6	7	70.65
7	8	72.50
8	9	69.24

Table 10. Establish the best structure of model of file's section' header

From the experiment above, the highest recognition rate is 75.94%. So the structure 7-5-1 is selected as the structure of our model. Then the model will be tested.

Experiment parameters : The structure of model is 7-5-1, the total number of samples is 1336, the number of samples to train the model is 1000, the number of samples to test the model is 366, error=0.45.

Parameters / Times	Learning	Train times	Recognition rate for train (%)	Recognition rate for test (%)
1	0.001	2000	75.50	62.75
2	0.001	1000	77.25	68.67
3	0.05	1000	76.50	66.58

Table 11. The last results of model of file's section's header

4.2.3 The classifier for the Section body

As the names and numbers of attributes of each section body are different, we cannot find the common attributes of section bodies. The problem would be solved if we made experiments on each section body respectively. However, this would lead to too heavy workload. On the other hand, as it is a general opinion that if one can process one Section body, he can process the other Section bodies, we decide to conduct experiment of one section body: import section.

It is the most complex part of all the work to deal with this one field of import section: import function name.

Import Section structure features.

Import Section include 2 layers :

Import File—> Import Function, as shown in Figure 9 :

ImportFile name	ImportFunction name
kernel32.dll	GetFileSize
kernel32.dll	GetSystemTime
kernel32.dll	GetFileType
kernel32.dll	CreateFileA
kernel32.dll	CloseHandle
user32.dll	GetKeyboardType
user32.dll	LoadStringA
user32.dll	MessageBoxA
user32.dll	CharNextA

Figure9. Import Section examples

For the field of import file name, the way used in the model of file's section header can be used to calculate the probability as the value of field.

When we calculate the probability of Import Function in the same way as we used in the model of file's section header, we get the following situation:

The total number of function file used: 46652

The total number of function file (not repeated) used: 4792

$$\text{The repeated rate: } \frac{(46652-4749)}{46652} = 89.82\%$$

If there are 10 import Files in the PE file and there are about 100 import functions in every import File, then it will match 10*100*4792=4792000 times, the efficiency is too low. And there are 4792 functions which need to be calculated. It is too difficulty to calculate all the 4792 functions. And these 4792 functions are not all the functions. If there are some new functions, the probability cannot be calculated by this way. So this method cannot be

used to solve the question.

After repeated experiments, we get the most effective method – as every function consists of several words, we count these words with the same method that we count the section name. The total number of words (not repeated) is 2375. Then the probability of these words will be calculated, and then we use average value of the probability of words to calculate the probability of Import function name.

The ways of other field's normalization as shown in Table12:

Field name	Normalization method
dIMPORT_DESCRIPTOR_OriginalFirstThunk	dIMPORT_DESCRIPTOR_OriginalFirstThunk /20001512.0
dIMPORT_DESCRIPTOR_TimeDateStamp	Int(dIMPORT_DESCRIPTOR_TimeDateStamp) +1
dIMPORT_DESCRIPTOR_ForwarderChain	Int(dIMPORT_DESCRIPTOR_ForwarderChain) +1
dIMPORT_DESCRIPTOR_FirstThunk	dIMPORT_DESCRIPTOR_FirstThunk /20002288.0

Table 12. The ways to deal with the other fields of import section

Next, we setup the classifier. In the input layer, all the 6 fields are determined to set up model. They are import file name, import function and the other four fields. The names of the other four fields are shown in Table 7. In the hide layer, from the experienced formula:

Number of units in hide layer= $\sqrt{numberofunitininput * numberofunitinoutput}$ [7]

The reasonable calculation of the number of units in hide layer is about 3. Then experiments are made to establish the best structure of model.

Experiment parameters : The number of units in input layer is 6, the number of units in output layer is 1, the total number of samples is 1300, the number of samples to train the model is 1000, error=0.45, learning=0.001, train times is 2000.

Parameters / Times	Structure	Recognition rate for train (%)
1	6-2-1	53.67
2	6-3-1	58.56
3	6-4-1	62.56
4	6-5-1	65.56
5	6-6-1	59.44
6	6-7-1	61.89
7	6-8-1	62.12
8	6-9-1	52.56

Table 13. Establish the best structure of model of file's import section

From the Table above, the structure of model, 6-5-1, is established. Then the model will be tested:

Experiment parameters: structure is 6-5-1, the total number of samples is 1300, the number of samples to train the model is 1000, the number of samples to test the model is 300, error=0.45.

Parameters / Times	Learning	Train times	Recognition rate for train (%)	Recognition rate for test (%)
1	0.001	2000	66.50	58.55
2	0.001	1000	67.25	66.33
3	0.05	1000	65.75	65.50

Table 14. The last results of model of file's import section

The classifiers for each part have been set up, and then the results of classifiers will be combined to judge whether the file is Trojan horse or not. The particulars of combining the results of classifiers are presented in Chapter 5.

CHAPTER 5

COMBINING THE RESULTS OF CLASSIFIERS

5.1 The capabilities of classifiers

The classifiers set up in chapter 4 are shown in Table 15.

Num	Name	Place in File	Recognition rate for train	Recognition rate for test
1	Classifier 1	File header	82.00%	68.65%
2	Classifier 2	Section header	77.25%	68.67%
3	Classifier 3	Section body	67.25%	66.33%

Table 15. The capabilities of classifiers

Now that the classifiers for each part are set up, the results should be combined to judge whether the file is Trojan horse or not.

5.2 To judge the Trojan horse

In the experiments, the outputs of classifiers for each part are the value after correcting. The "error" is used to correct the actual outputs. But in the experiments to combine the results of classifiers, the actual outputs of classifiers are accepted as the results of classifiers to judge whether the file is Trojan horse or not.

The methods, which will be used to combine the results of classifier, are weighted

mean, unweighted mean and majority voting. As to the method of weighted mean, a model of two layers neural network (one input layer and one output layer)will be set up to establish the weights. The data used to train and test the model are the three results (actual output) of the three classifiers. And the best result is shown in following Table 16.

The classifiers	method to combine	Weighting	Recognition rate
Classifier 1	weighted mean	0.414	76.67%
Classifier 2		0.389	
Classifier 3		0.213	

Table 16. The best result of method of weighted mean

For the method of unweighted mean, the average of three results (actual output) of classifiers is calculated directly, and the average is used to judge whether a file is Trojan horse. In this experiment , the key factor affecting the Recognition rate is the "dividing point" that is used to judge. The "dividing point" is a value which is used to distinguish the Trojan horses from a normal file. If the average is higher than the "dividing point", then the file is Trojan horse, and if the average is less than "dividing point", the file is a normal file. And when we set the value of "dividing point" in the range of [5.5, 6], it will get better Recognition rate. In the experiment, the best Recognition rate is gotten when the value of "dividing point" is 5.67. And the best result is shown in following Table 17.

The classifiers	method to combine	Recognition rate
Classifier 1	unweighted mean	74.66%
Classifier 2		
Classifier 3		

Table 17. The best result of method of unweighted mean

With the method of majority voting, a file can be judged as Trojan horse if more than two classifiers judge it to be a Trojan. In the same way, a file will be a normal file if more than two classifiers judge it to be a normal one. The result is shown in the following Table 18.

The classifiers	method to combine	Recognition rate
Classifier 1	majority voting	71.54%
Classifier 2		
Classifier 3		

Table 18. The result of method of majority voting

The results of different way to combine the classifiers are shown in Table 19:

Num	Way to combine	Recognition rate
1	weighted mean	76.67%
2	unweighted mean	74.66%
3	majority voting	71.54%

Table 19. The results of different way to combine the classifiers

The highest recognition rate is 76.67%. Though we try hard and the recognition rate has been raised greatly, but the last result is not good enough. We think the main reason is that the capabilities of classifiers for each part are not good enough. All the Recognition rates of classifiers are below 70%.

In Table 19, it is obvious that the result of the method of weighted mean is better than the result of the method of unweighted mean, while the result of method of unweighted mean is better than the result of the method of majority voting. It indicates that the effects (to judge the Trojan horse) of different part of FE file are different, and in our model the File header is the more important part to judge the Trojan horse.

CHAPTER 6
CONCLUSION

6.1 Summary

In conclusion, as traditional detection techniques have limitations in detecting Trojan horse [2] [19], new ways of detecting Trojan horse must be established to solve the problem which have posed a huge security threat to computer network. In this thesis, we present a new method to detect Trojan horse by analyzing a file's static characteristics. This new method can detect all the Trojan horses. Considering about 75% of world's personal computer install the Microsoft Windows [4] and Trojan horse usually exists as a Portable Executable (PE) file in Windows platform, we take PE format as our subject, then we extract the static information of files according to PE format, and finally we use the neural network to analyze the static information. We setup models as classifiers to judge whether a PE file is a Trojan horse or not. By dividing the data of PE file in three groups, we solve the problem caused by the structure of PE file in setting up the model of neural network. As we divide the data in three groups, we setup classifier for each group. At last we combine the results of classifiers to judge whether the file is Trojan file or not. And with the model, we can detect the unknown Trojan horse by analyzing static characteristics of file. We verify our results by the information that is not used during training (i.e., the files to which the

information belongs is new and unknown to our model).The final results are as follows:

The classifiers for different parts of PE file are shown in following Table.

Num	Name	Place in File	Recognition rate for train (%)	Recognition rate for test (%)
1	Classifier 1	File header	82.00	68.65
2	Classifier 2	Section header	77.25	68.67
3	Classifier 3	Section body	67.25	66.33

Table 15. The classifier of three parts

The results of different methods to combine the classifiers are shown in following Table.

Num	The classifiers	Way to combine	Weighting	Recognition rate
1	Classifier 1	weighted mean	0.414	76.67%
	Classifier 2		0.389	
	Classifier 3		0.213	
2	Classifier 1	unweighted mean	0.333	74.66%
	Classifier 2		0.333	
	Classifier 3		0.333	
3	Classifier 1	majority voting		71.54%
	Classifier 2			
	Classifier 3			

Table 20. The results of different method to combine the classifiers (particular)

The highest recognition rate is 76.67%. Though we try hard and the recognition rate has been raised greatly, the last result is not good enough. We think the main reason is that the capabilities of classifiers for each part are not good enough. The Recognition rates of classifiers are less than 70% all. So we should try to raise the recognition rate of classifiers of each part.

6.2 The future works

As time is limited and the workload very large, the work we did are far from enough. In the future we should endeavor in the following aspects:

- ✧ Try other methods of intelligent information processing technique. There are many methods which can solve the problem of classification. Try other methods or use other methods to assist neural network in more links of experiments. It will get better results.
- ✧ Try to optimize the BP learning algorithm. We only use some basic ways to optimize the algorithm, which may partly explain why the results of our classifiers are not good enough. Try more ways to optimize the algorithm in order to get better results.
- ✧ Try more ways of pretreatment of input data. The pretreatment of input data can decide whether the experiment can be successful, and it is a difficult problem in our works. Trying more solving ways will help our work greatly.
- ✧ Increase the number of samples. The number of samples of models is small, such as the models of file header which have many units of input layer, Sufficient samples avail us to find more important attributes and parts of the PE file for detecting Trojan horse.

References

[1] Bishop, C.M (1996). *Neural networks for pattern recognition.*

[2] Ccidnet. *The overall analysis of Trojan horse.* Date posted in ITsecurity.com: 10 sep 2004 (in Chinese)

[3] Chellappa, R., Fukishima, K., Katsaggelos, A., Kung, S-Y., LeCun, Y., Nasrabadi, N. M. and Poggio, T. A., "*Applications of artificial neural networks to image processing.*" *IEEE Transactions on Image Processing*, vol. 7, no. 8, Aug. 1998.

[4] Eleazar Eskin, Matthew G. Schultz, Erez Zadok, and Salvatore J. Stolfo "*Learning to Detect New and Unknown Malicious Programs,*" *2004 Computer Science Department, Columbia University.*

[5] Fred Cohen. *A Short Course on Computer Viruses.* ASP Press, 1990.

[6] Gerald Tesauro, Jeffrey O. Kephart, Gregory B. Sorkin, "*Neural Networks for Computer Virus Recognition.*" High Integrity Computing Laboratory IBM Thomas J. Watson Research Center IEEE Expert, vol. 11, no. 4, August 1996

[7] Han Liqun 《*The theory , design and application of artificial neural network*》 ISBN: 7-5025-3354-0 2002.1 (in Chinese)

[8] J.E. Dickerson, J. Juslin*, O. Koukousoula*, J.A. Dickerson, "*Fuzzy intrusion detection,*" *IFSA World Congress and 20th North American Fuzzy Information Processing Society (NAFIPS) International Conference, Vancouver, British Columbia, Volume 3, 1506-1510, July, 2001.*

[9] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. and Jackel, L. D., "*Backpropagation Applied to Handwritten Zip Code Recognition.*" *Neural Computation*, vol. 1, no. 4, 1989.

[10] LeCun, Y. and Bengio, Y., "*Pattern Recognition and Neural Networks*" in *The Handbook of Brain Theory and Neural Networks*, Arbib, M. A. (Ed.), MIT Press, 1995.

[11] Matt Pietrek. *An In-Depth Look into the Win32 Portable Executable File Format.* February 2002 MSDN

[12] Matt Pietrek. *Peering Inside the PE: A Tour of the Win32 Portable Executable File Format.* March 1994 msdn.microsopft.com

[13] Michael (Micha) Shafir. *Are WEB applications Trojan horses?* 9 July 2002

[14] Microsoft. *Microsoft Portable Executable and Common Object File Format Specification Microsoft Corporation Revision 6.0.* February 1999

[15] Mihai Christodorescu, Somesh Jha. *Static Analysis of Executables to Detect Malicious Patterns. Computer Sciences Department University of Wisconsin, Madison, 2003.*

[16] Michael Weber, Matthew Schmid & Michael Schatz Cigital, Inc. *A toolkit for detecting and analyzing malicious software. Dulles, VA 20166. 2002*

[17] National Infrastructure Security Co-Ordination Centre *Trojan Horse Programs and Rootkits.* NISCC Technical Note 08/03

[18] Prasad Dabak, Milind Borate, Sandeep Phadke. *Portable Executable File Format.* October 1999

[19] Rising. *Introduce of Trojan horse*, Date posted in rising.com: Dec 2002 (in Chinese)

[20] Randy Kath. *The Portable Executable File Format from Top to Bottom.* Microsoft Developer Network Technology Group (MSDN)

[21] Sackinger, E., Boser, B., Bromley, J., LeCun, Y. and Jackel, L. D., "*Application of the ANN Neural Network Chip to High-Speed Character Recognition.*" *IEEE Transaction on Neural Networks*, vol. 3, no. 2, March, 1992.

[22] Solla, S. and LeCun, Y., "*Constrained Neural Networks for Pattern Recognition,*" *Neural Networks: Concepts, Applications and Implementations Vol IV*, Antognetti, P. and Milutinovic, V. (Ed.), Prentice Hall, 1991.

[23] Steven M. Beattie, Andrew P. Black, Crispin Cowan, Calton Pu, Lateef P. Yang. *Guardhouse: Locking the Stable door ahead of the Trojan horse.* Department of Computer Science & Engineering, Oregon Graduate Institute of Science & Technology. 1999

[24] Tom M.Mitchell. 《*Machine Learning*》 ISBN: 0070428077 Publisher: McGraw-Hill Science/Engineering/Math; (March 1, 1997)