Journal of Computers Vol. 33 No. 3, June 2022, pp. 225-238
doi: 10.53106/199115992022063303018

Design of Malicious Code Detection System Based on Binary Code Slicing

Mohan Liu[1*], Xiaoming Tang[2], Hanming Fei[3]

[1] China Railway Network Company Limited, Beijing, China
[2] Beijing Zhongxing Ruipu Technology Company Limited, Beijing, China
[3] Institute of Electronic Computing, Technology of China Railway Academy Group Co., Ltd., Beijing, China
liumohan@crnet.com.cn, tangxm0720@163.com, fhm727@163.com

Received 16 April, 2022; Revised 16 May 2022; Accepted 16 June 2022

Abstract: Malicious code threatens the safety of computer systems. Researching malicious code design techniques and mastering code behavior patterns are the basic work of network security prevention. With the game of network offense and defense, malicious code shows the characteristics of invisibility, polymorphism, and multi-dismutation. How to correctly and effectively understand malicious code and extract the key malicious features is the main goal of malicious code detection technology. As an important method of program understanding, program slicing is used to analyze the program code by using the idea of "decomposition", and then extract the code fragments that the analyst is interested in. In recent years, data mining and machine learning techniques have been applied to the field of malicious code detection. The reason why it has become the focus of research is that it can use data mining to dig out meaningful patterns from a large amount of existing code data. Machine learning can It helps to summarize the identification knowledge of known malicious code, so as to conduct similarity search and help find unknown malicious code. The machine learning heuristic malicious code detection method firstly needs to automatically or manually extract the structure, function and behavior characteristics of the malicious code, so we can first slice the malicious code and then perform the detection.

Through the improvement of the classic program slicing algorithm, this paper effectively improves the slicing problem between binary code processes. At the same time, it implements a malicious code detection system. The machine code byte sequence variable-length N-gram is used as the feature extraction method to further prove that the efficiency and accuracy of malicious code detection technology based on data mining and machine learning.

Keywords: binary analysis, slicing; malicious code detection, network security

1. Purpose and Meaning

In a broad sense, malicious code refers to all malicious programs designed to damage computers or network systems or consume user system resources, including network worms, computer viruses, Trojan horses, malicious web scripts, hacker attack programs, etc [1]. For example, a Trojan horse is a malicious code whose main target is to remotely control a user's computer. It sneaks into the user's computer like a spy. It is very similar to the "Trojan horse" tactics in the war, hence the name Trojan horse; the virus spreads by infecting computer files, with the main purpose of destroying or tampering with user data and affecting the normal operation of the computer system. Malicious code.

In general, malicious code may be implanted into the target computer through network security vulnerabilities, emails, storage media, or other means, and run automatically when the target computer starts [2]. The malicious code found so far includes executable programs, dynamic link libraries, help files, image files, screen saver files, connection files, WORD files, OCX controls, JAVA applets, script programs, etc [3]. Among them, the most important forms of existence are executable programs and dynamic link libraries.

On the current Internet, malicious codes are everywhere and flooded, seriously threatening network security. Since the establishment of CERT in 1988 due to the Morris worm incident, Internet security threats have increased year by year [4]. The growth trend has become particularly rapid in recent years [5]. From 1998 to 2004, the average annual growth rate reached about 50%. In Internet security incidents, the direct economic losses caused by malicious code accounted for the largest proportion [6]. The "Love Bug" virus that broke out in May 2000 caused approximately US$8.7 billion in economic losses. When the Slammer worm broke out in 2003, com-

* Corresponding Author

puter users worldwide lost more than $2 billion in just eight days. According to the survey results released by the risk management company MI2G, in 2004, malicious programs such as viruses, worms and Trojan horses caused a total of US$169 billion in economic losses worldwide [7]. This figure is equivalent to more than twice that of 2003.

In recent years, with the development of technology, there are many popular malicious codes and their variants [8]. Each type of malicious code has its own characteristics in purpose, function, form of existence, activation method, damage consequence and other methods. The schematic diagram of malicious code is shown in Fig. 1 below. The consequences of malicious code attacks are diverse. They may cause the host system to crash, may cause sensitive information leakage, may consume host resources and cause system performance degradation, or may block the network and affect normal operation [9].

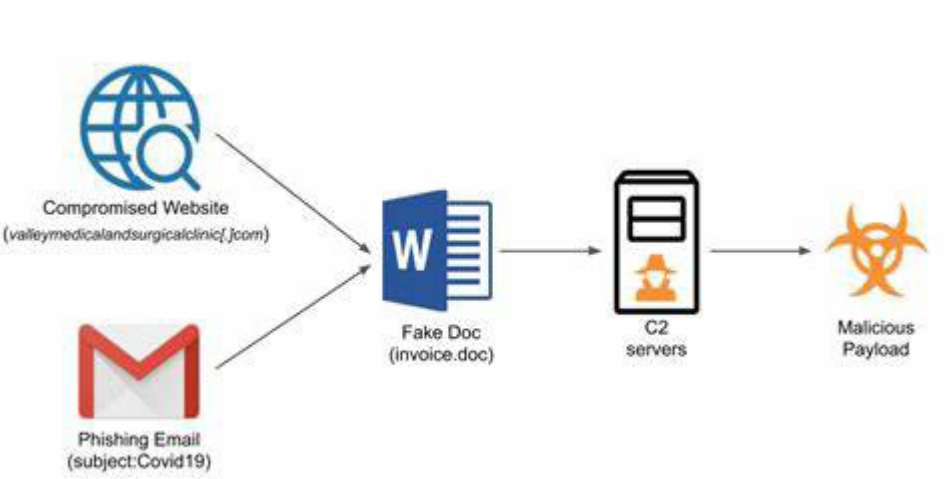

Fig. 1. Malicious code attack process

This makes people usher in new challenges, and the use of technologies such as data mining and machine learning to detect unknown malicious code has become a new direction of malicious code research [10]. Use data mining and machine learning theories to detect malicious code, mainly using classification techniques in data mining, and discover relevant patterns by learning the characteristics of malicious code and normal code [11]. These patterns can be used to perform similarity searches to find malicious code containing similar patterns. Not only can it detect unknown malicious code before it is not infected, but it also has a higher detection rate for vaguely transformed malicious code, which is the development of malicious code detection. Important direction. This method does not need to actually run the code and belongs to the category of static detection.

At the same time, various security measures can only reduce the possibility of the system being attacked, but cannot eliminate the vulnerability of the system; and the testing methods can only prove that the system is vulnerable, but cannot prove that the system is not vulnerable. Moreover, in order to meet actual needs, the scale of information systems is getting larger and larger, and the problem of security vulnerabilities will become more and more prominent. As these vulnerabilities are gradually discovered by attackers, there will continue to be new ones aimed at these vulnerabilities. Malicious code appears.

Therefore, the research of malicious code detection is a very urgent and important task, which has important theoretical and practical value. This requires detection tools not only to be able to detect malicious code that has appeared, but also to be able to detect new malicious code, not only with a low false positive rate, but also a low false negative rate.

This paper mainly focuses on the research direction of malicious code detection based on data mining and machine learning, and proposes a binary code slicing method. By removing redundant nodes of parameter transfer dependencies when the binary code procedure is called, the classic program slicing algorithm reduces the need for binary code. The complexity of generating the system dependency graph. Finally, a malicious code detection system is proposed. Finally, the test experiment proves that the malicious code detection proposed in this paper can protect users' network security to a certain extent, and the report given by the analysis platform can be used as an important basis for the judgment of security analysts.

2. Related Work

In the fight against malicious code, early detection of malicious code is very important. If found early and dealt with in time, losses can be reduced. Regarding the security issues of malicious code detection, the early detection method is to apply a signature-based method. These signatures include many different attributes, such as file names, content strings, or bytes, etc., and also discuss protecting system security from the perspective of excluding security vulnerabilities caused by these malicious codes. At that time, experts used their professional knowledge to manually analyze suspicious procedures to find relevant features. Although this analysis method is accurate but time-consuming and laborious, it may work well when there is only a small amount of malicious code. Currently, the commonly used method is to detect malicious code by comparing it with a set of known malicious codes.

Anti-program slicing is the process of analyzing program code and extracting part of the sequence of sentences (instructions) according to the specific needs of the analyst [12]. Program slicing is based on the control flow and data flow of instructions, analyzes the execution sequence of instructions and the process of data generation, copy, transfer and disappearance, and finally generates the required code fragments [13]. In terms of malicious code analysis, there is not much research done in China, but there is an increasing trend abroad. In 2001, Matthew G. Schultz and Eleazar Eskin proposed a data mining-based malicious code detection method on the basis of anti-virus software feature detection [14]. Since then, Mihai Christodorescu has been involved in the static analysis of malicious code [15], and Mihai Christodorescu has been involved in the detection of variant viruses. Mihai Christodorescu, J. ergeron [16.] In terms of code analysis based on behavioral characteristics, Tony Abou-Assaleh has done a lot of research work on the application of pattern matching method in the field of malicious code analysis [17]. The focus of these studies is mainly to determine the type of malicious code. In practical applications, there are still many challenges.

With the development of technology, traditional malicious code technologies such as viruses, worms, and Trojan horses have a trend of fusion [18], and judging the type of malicious code has lost its meaning. So far, there have been many attempts to use machine learning and data mining techniques to detect unknown malicious code [19]. The schematic diagram of machine learning is shown in Fig. 2.

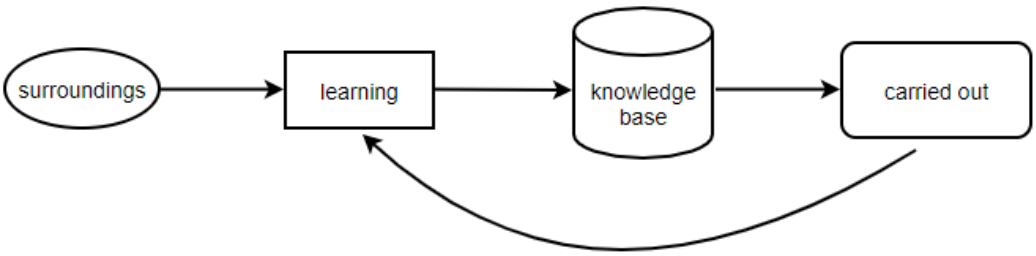

Fig. 2. Machine learning model

In 1994, Kephart et al. proposed the use of ANN (Artificial Neural Network, artificial neural network) method to detect boot sector viruses, but this classifier has great limitations, except for very small boot sector viruses, it cannot detect. For any other malicious code, these boot sector viruses only account for 5% of all malicious codes. Arnold (2000) used the same method to detect Win32 viruses. This method can also only detect Win32 viruses, and due to the limitations of the artificial neural network classifier, the gains are not great. In 2001, Schultz et al. proposed a series of data mining techniques to detect different types of viruses, extracting code features such as Win32 dll file calls, ASCII strings, byte sequences, etc., and using a variety of classification algorithms, including RIPPER, Naive Bayes and Multiple naive Bayes algorithm, among which the highest accuracy of detecting unknown malicious code is the multiple naive Bayes algorithm with byte sequence as the feature [20].

Schultz pioneered the use of data mining methods to detect malicious code, and this method has since become the focus of research in the field of malicious code detection. Abou-Assaleh (2004) found that N-gram as an effective feature can be extracted from the byte sequence, and use CNG (Common N-gram) as a feature, and k-NN (k nearest neighbor) as a classifier. To detect unknown viruses. Kolter (2004) et al. regard the byte sequence N-gram as a feature set. Because this feature set is too large, some of the most relevant features are selected to form the feature set based on the feature gain. Use a series of classification algorithms, such as IBk, Naive Bayes, support vector machines, decision trees, and bsoosted SVMs, J48. Among them, boosted J48 has the highest detection

rate. Two years later, Kolter (2006) tried to use the same method to classify malicious code into worms, viruses, Trojan horses, etc. Although the effect was not as good as malicious code detection, it also achieved a correct rate of about 90% [21].

In the military field, the key technology of malicious code is also one of the main research contents of information warfare. The United States has been studying information warfare centered on computer network offensive and defensive confrontation since the early 1980s, and regards information warfare as a key element for seizing information superiority and maintaining military superiority in the 21st century. Russia ranks information warfare as an important position second only to nuclear war. In its 1996-2000 Action Plan, it emphasizes that while properly maintaining nuclear potential, it should focus on the extensive development of information warfare means. In the Kosovo war, computer network warfare has begun to take shape.

In-depth study of the key implementation technology of malicious code and analysis technology of malicious code has very important practical significance, mainly in the following aspects:

1. The key technology of malicious code is one of the key technologies for conducting network information warfare. Using the key technology of malicious code to obtain military and commercial intelligence through the Internet is one of the key tasks of relevant state departments.

2. Malicious code is the main tool used by hackers to control the system and conduct network damage. As a network security worker, it is necessary to master the key technologies and development trends of malicious code to lay the foundation for other tasks.

3. Malicious code analysis technology is the basis of network emergency response and computer forensics. By analyzing the malicious code, we can understand the basic functions of the malicious code, grasp the possible damage activities of the malicious code to the victim system, provide information for the system recovery and system loss assessment of the victim host, and master the malicious code attack technology, self-starting technology, Deletion protection technology provides information for defense and removal of malicious code.

4. Malicious code analysis is the technical basis for malicious code detection. Existing malicious code detection technologies are all based on code features. These features include behavioral features and static data features. Through malicious code analysis, the features of malicious code can be obtained, and these features are used as the basis for detection.

5. Master new technologies and new methods of malicious code by analyzing malicious code. Hackers always show their superb computer technology in the malicious code they implement. Some of these technologies may be the latest. By analyzing the malicious code, they can master these new technologies and use them for us.

3. System Structure

Since the development of malicious code detection technology, there are mainly two detection methods:

1）Rule-based Detection:

The malicious code detection engine detects samples based on the malicious code feature rule base. The rule base mainly includes fingerprint features for malicious instructions and pattern features for malicious behavior. This detection method has high accuracy rate and short detection time, but it needs to define rules in advance, so it has better results for the detection of known samples, and its detection ability for unknown samples is relatively weak.

2）Heuristic Detection:

By monitoring the activity of the system and categorizing it as normal or abnormal, it detects whether the sample has malicious attempts. The current judgment of abnormal conditions is usually based on machine learning algorithms, which requires a period of training and modeling by the malicious code detection engine. Because this detection method is based on statistical features and probabilistic decision-making models, it usually has a certain false alarm rate and low detection performance, but it has a higher recall rate for unknown sample detection compared to regular detection.

The general model of malicious code is shown in Fig. 3 below.

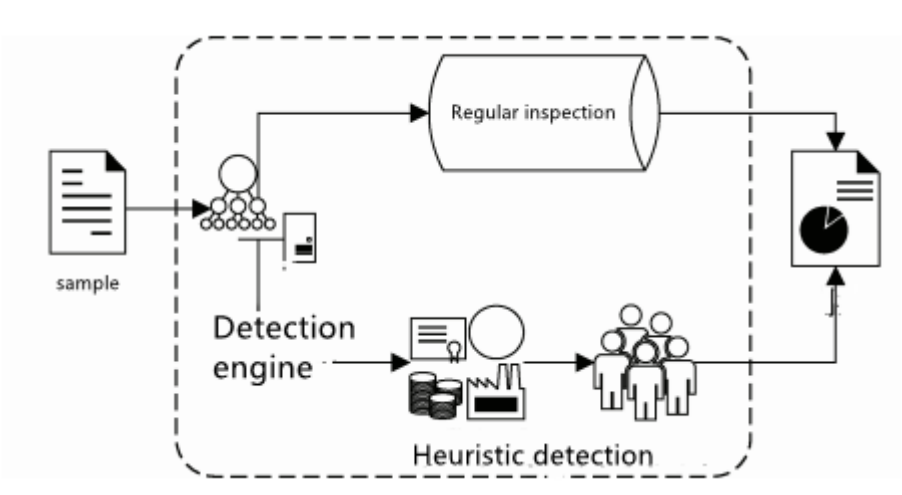

Fig. 3. General model of malicious code

3.1 Selection of Slicing Criteria

In order to clean and preprocess the original features of the malicious code and improve the accuracy and recall rate of the classification model, we manually analyzed 10 malicious code samples with a spread range of TOP 100 used by 10 cyber attack organizations, and designed a set of malicious code samples. The slicing criteria widely used by malicious code [20-22] are the basic behavior slicing criteria and the anti-analysis behavior slicing criteria. The list of slicing criteria is listed in Table 1. Among them, the basic behavior slicing guidelines include 22 file behaviors, registry behaviors, network behaviors, process/thread behaviors, etc.; anti-analysis behavior slicing guidelines include anti-virtual machine behavior, anti-sandbox behavior, and anti-debugging behavior, totaling 20 items. Extract 42 key slicing criteria.

3.2 List of Slicing Guidelines

Table 1. Slicing criterion list

	Type of program behavior	Number of program slices
Basic behavior	File behavior	4
	Registry behavior	4
	Network behavior	3
	Process behavior	5
	mutex operation	2
	Service operation	2
	DLL operation	2
Back analysis	Anti-virtual machine detection	10
	Anti-sandbox detection	8
	Anti-debugging detection	2

Based on the common characteristics of malicious behavior of malicious code, this article expands the classic slicing criterion and describes it as the following triplet: <{API, instructions}, {parameters}, {return values}>. Among them, the first element {API, instructions} is used to locate the starting position of the program slice, including key API calls or key assembly instructions; the second element {parameters} is the parameters of the API, describing the beginning of the program slice Data dependency; The third element {return values} is the return value of the API and the return value of the parameter passed in through pointers. This element describes the forward data dependency of the starting position of the program slice. Table 2 takes the anti-sandbox behavior slicing criteria as an example. It lists the eight slicing criteria used by the malicious code extracted in this article to detect sandbox behavior. The remaining slicing criteria are defined in accordance with the above-mentioned triples and extracted to be able to represent Characteristics of key behaviors of malicious code.

Table 2. Slicing criterion list of anti-sandbox behaviors

Slicing criterion number	Slicing criteria description	Call API	parameter	return
01	Check whether the cursor position change belongs to the sandbox environment	call GetCursorPos		
02	Check the user name to determine	call GetUserName		"SANDBOX" "VIRUS" "MALWARE" "\\SAMPLE" "\\VIRUS"
03	Test program running directory	call GetModuleFileName		
04	Obtain the disk size judgment (Method 1)	call CreateFile	"\\\\.\\PhysicalDrive0"	
05	Obtain the disk size judgment (Method 2)	call GetDiskFreeSpace		
06	Detection time judgment sandbox time acceleration	call GetTickCount		
07	Get the judgment of the number of CPU cores	mov eax, fs:0x18		
08	Detection of pending module judgment	call GetModuleHandle	"sbiedll.dll"	

3.3 Malicious Code Detection System Structure and Performance Indicators

3.3.1 System Architecture

This section implements a malicious code detection system based on data mining and machine learning methods, using N-grams and variable-length N-grams of machine code byte sequences as features, weighted information gain as a feature selection method, and multiple classification algorithms A system that implements malicious code detection. The model is shown in Fig. 4.

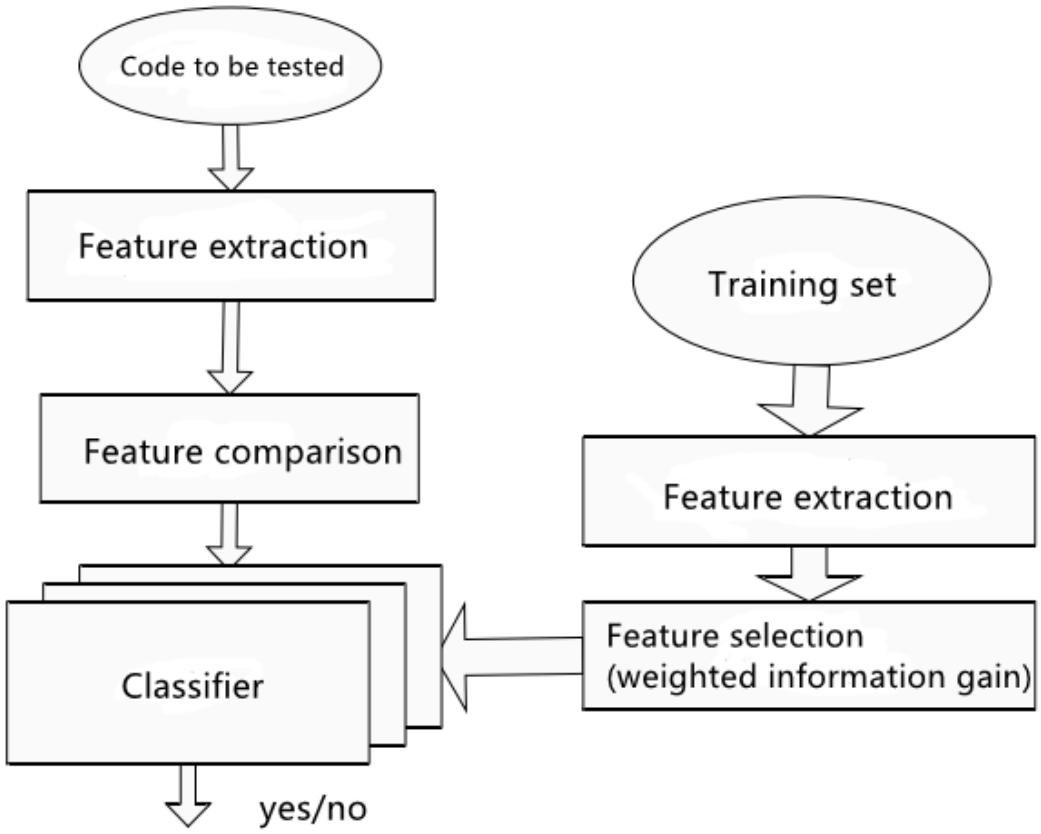

Fig. 4. Malicious code detection model

The first part is the training part: first select a certain number of malicious codes and normal codes as the training set, extract the N-gram and variable-length N-gram of the code binary sequence as features; then perform feature selection and calculate the weight corresponding to each feature The information gain IW(j) is sorted in

descending order of IW(j), and the first several are selected as effective features. According to whether each training sample contains these features, a Boolean vector space is formed for the classifier to learn.

The second part is the detection part: extract the N-gram and variable-length N-gram of the binary sequence of the code to be tested as features, and form a Boolean vector space according to whether each code to be tested contains the effective features selected by the training part. A classification algorithm analyzes the vector space to determine whether the code to be tested is malicious.

3.3.2 Malicious Code Detection Performance Indicators

Accuracy (TP+TN)/N(OA): That is, the proportion of all correctly classified codes in the set to be tested.

Detection rate TP/(TP+FN) (DTR): the proportion of the number of correctly classified malicious codes in all malicious codes in the set to be tested.

False positive rate FP/(TN+FP) (FPR): That is, the proportion of normal codes that are incorrectly classified as malicious codes in all normal codes in the set to be tested.

3.3.3 Feature Selection Method

The features extracted by the above methods contain many redundant features, and it is necessary to select features that are beneficial to distinguish code types from them. Feature selection is a process of finding the most informative feature that can accurately describe the original case.

In malicious code detection, cases refer to malicious code and normal code, and candidate features refer to byte sequences of a certain length. Regardless of whether it is a single-byte sequence or a multi-byte sequence, the number of extracted features is very large. The purpose of feature selection is to select the most relevant set of features among these features, usually this set of features is much smaller than the original feature set, so as to obtain the most satisfactory classification results. There are many kinds of feature selection methods: maximum difference (MD), maximum standard deviation (MND), maximum ratio (MR), maximum weight ratio (MWR), maximum KL distance (MKLD), class domain frequency (CF), information gain (IG), and Weighted Information Gain (WIG). Several methods are briefly introduced below.

1）Maximal Difference

Calculate the average frequency of feature F in normal code and the average frequency of F in malicious code respectively. Calculate the difference for each F:

$$P(F|B) - P(F|M)$$

The differences are sorted in ascending order. The features at the top of the list appear more frequently in malicious code than in normal code, that is, these features appear more frequently in malicious code; the features at the back of the list appear more frequently in normal code It appears more frequently than in malicious code, that is, these features appear more frequently in normal code. If we finally choose n relevant features, then we choose the n/2 features at the front of the list and the n/2 features at the back of the list.

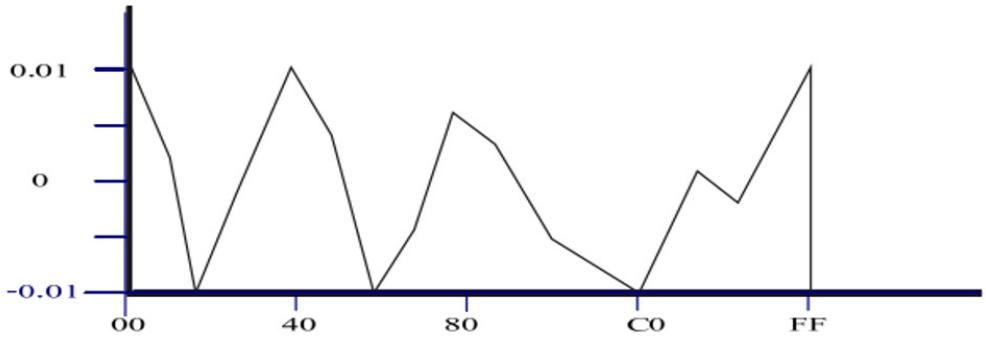

Fig. 5. The maximum difference between normal and malicious code characteristics

In Fig. 5, we depict the P(F| B) −P(F| M) values of several single-byte features. It can be seen from the figure that the maximum value of the largest difference appears at bytes 00, 40, and FF, that is, bytes 00, 40, and FF appear most frequently in normal codes; the largest difference appears at bytes 17, 5A, and C0. The minimum value, that is, bytes 17, 5A, and C0 appear most frequently in malicious code.

2）Maximal Normalized Difference

This method is similar to the maximum difference method. The difference is that the maximum standard deviation takes into account the difference in the characteristic frequency of each file, so the calculated maximum difference does not fully reflect the real situation. Let σ(F|B) be the feature F in the standard deviation of the frequency P(F) in the normal code, and the same σ(F|M) is the standard deviation of the frequency P(F) of the feature F in the malicious code. The total standard deviation is

$$\sigma^2(F)=\frac{N_B\sigma^2(F\mid B)+N_M\sigma^2(F\mid M)}{N_B+N_M}$$

The maximum standard deviation is:

$$\frac{P(F\mid B)-P(F\mid M)}{\sigma(F)}$$

3.4 Experimental Simulation and Result Analysis

3.4.1 WEKA Data Mining Platform

The full name of WEKA data mining platform is Waikato Intelligent Analysis Environment, as shown in Fig. 6 is the interface of WEKA. It integrates machine learning algorithms that can undertake data mining tasks, including data preprocessing, classification, regression, clustering, association rules, and visualization on a new interactive interface.

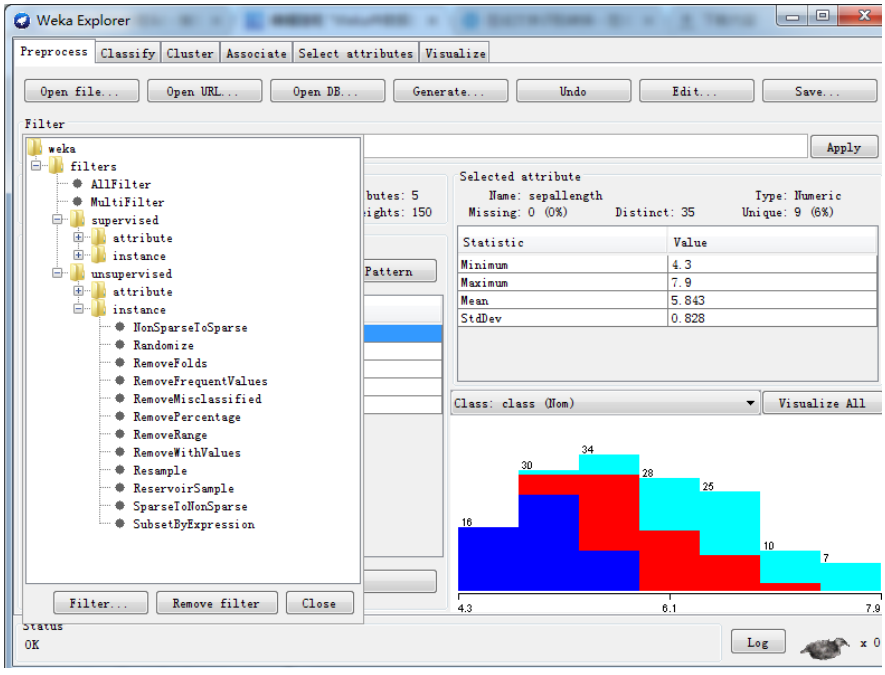

Fig. 6. WEKA interface diagram

3.4.2 Experimental Design and Result Analysis

429 normal codes and 408 malicious codes were used as the training sample set. All the malicious codes originated from websites, and the normal codes were obtained on a computer newly installed with Windows XP. The number of effective features is selected as 500, and the impact of different number of features on the classification results is compared. The classification algorithm is implemented by WEKA, and the experiment uses ten-fold cross-validation. First, use the IDA Pro disassembly tool to disassemble all files to obtain the byte sequence form of the machine code. Some instruction codes included in the malicious code are absent or very few in normal files. Similarly, there are instruction codes that are distinguishable from malicious codes in normal codes. We distinguish them by these codes. These instructions that can be used to distinguish the two types of codes are called features, and these features are used as feature vectors for analysis to form a training data sample set.

The working process of the system is divided into two phases: training phase and detection phase. In the training phase, there are two steps of machine learning and testing. First, train the classifier based on the known training data of the sample set containing malicious code and normal code, and then process the code feature data of the unknown state into a digital feature vector form, and then analyze and classify these features through the classifier, and finally the result of the discrimination is the category to which this code belongs.

When implementing N-gram and variable-length N-gram algorithms with N different times, information gain, class domain frequency, and weighted information gain are used as feature selection methods, and Naive Bayes, SVM, J48, and decision trees are selected. Malicious code detection system as a classifier. The results obtained are compared with those obtained by Kolter and Reddy, which proves the superiority of the variable-length N-grm and weighted information gain method in malicious code detection.

The following are the detection results of several combinations of feature extraction and selection methods. For each combination, the detection result with the best classification effect is selected. The results are shown in Table 3 below:

Table 3. The detection results of several feature extraction and selection method combinations

	Detection rate (%)	Correct rate (%)
3-gram (IG)	95.8	96.65
3-gram (Weighted IG)	97.9	98.21
4-gram (IG)	96.74	97.37
4-gram (Weighted IG)	98.37	98.8
Variable length N-gram (IG)	97.9	98.68
Variable length N-gram (Weighted IG)	98.83	99.16

Among them, the combination of variable length N-gram and weighted information gain has the highest detection rate. It can be seen from Table 1 that the detection results using weighted information gain are better than those of information gain, the performance of 4-gram is better than 3-gram, and the performance of variable-length N-gram is better than N-gram.

Based on the use of variable-length N-grams as features, we compare the perforormance of weighted information gain and class domain frequency.

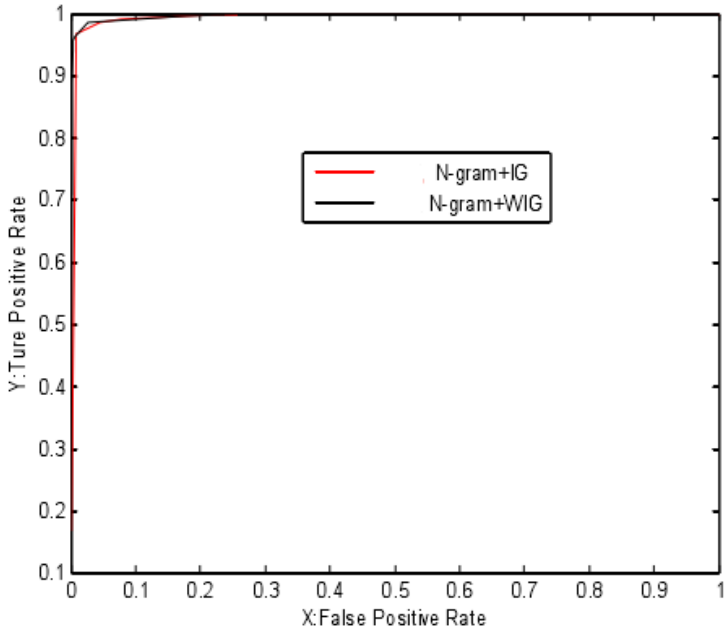

Fig. 7. ROC curve of information gain and weighted information gain when using variable-length N-grams as features

Fig. 7 is the ROC curve of information gain and weighted information gain when using variable-length N-grams as features. It can be seen from the figure that when using variable-length N-grams as features, information gain and weighted information gain are used to select features. , The detection results are not very different, the weighted information gain is slightly larger than the area under the ROC curve of the information gain, which can indicate that the feature of the weighted information gain selection is more effective than the feature of the information gain selection.

Fig. 8 shows the ROC curve of class-domain frequency and weighted information gain when using variable-length N-grams as features. Similarly, it can be clearly seen that the feature of weighted information gain selection is more effective than class domain frequency.

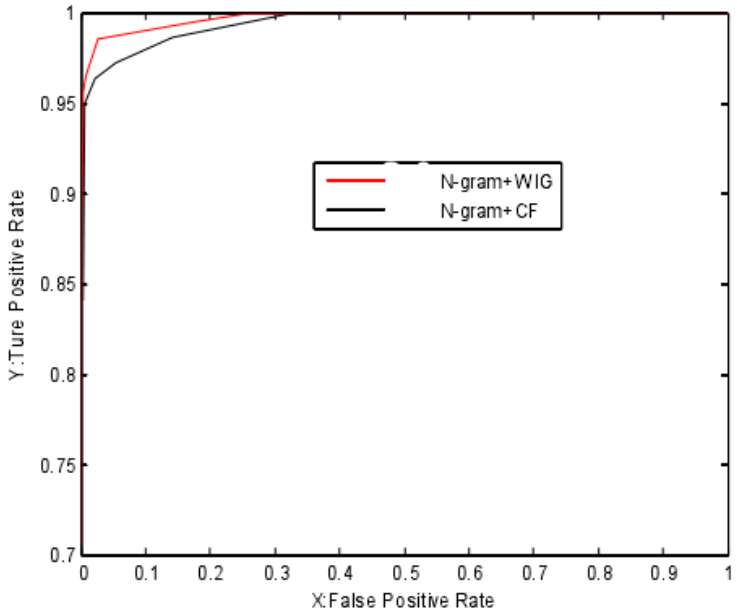

Fig. 8. ROC curve of class domain frequency and weighted information gain when using variable-length N-gram as the feature

Among them, the J48 algorithm has the highest detection rate. Fig. 9 depicts the ROC curves of the three classification algorithms.

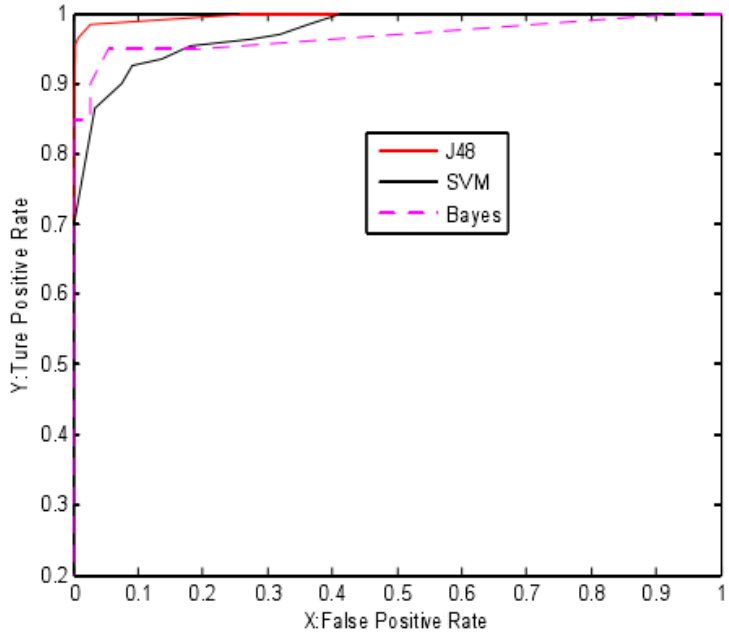

Fig. 9. Comparison of three classification algorithms

The above experimental results show that when the weighted information gain method selects effective features, the detection performance of the detection system is better than that of feature selection methods based on information gain and class domain frequency, which can more accurately select effective features and detect malicious code more effectively.

Traditional malicious code detection technology is mainly based on signature and heuristic methods. Signature-based detection method is to extract the characteristics of known virus samples, and search the virus database to find matching malicious code characteristics. This method has a high detection rate, but cannot detect emerging viruses. The heuristic algorithm uses experts to define a set of behaviors to detect unknown malicious code, which has high accuracy but low efficiency. Data mining methods can effectively detect unknown malicious code by learning the difference between malicious code and normal code.

4. Key Technology

The research on the key realization technology of malicious code and the research on the analysis technology of malicious code are two complementary tasks. By mastering the key technologies of malicious code, and being familiar with the working principle and workflow of malicious code, the efficiency of malicious code analysis can be improved; by analyzing malicious code, you can obtain the latest technical information of malicious code and improve the performance of the malicious code system designed by yourself. Including the reliability of system network attacks and the concealment of the control process.

4.1 Program Slicing Algorithm

Program slicing pays attention to the influence of instruction sequence on specific statements and variables, so slicing criteria are developed based on specific statements and variables to execute program slicing [18-19]. The slicing criterion contains two elements, namely the slicing target variable and the initial code position of the slicing. The slicing criterion of the program P can be described as a two-tuple <n, V>, where n is the number of a statement or basic block in the program, and V is the set of variables concerned by the slicing, which is a subset of the variables in P .

General steps for performing program slicing:

1) Program dependency extraction: including control dependency analysis and data dependency analysis, as well as the construction of program dependency graphs.

2) Construction of slicing criteria: Design a set of slicing criteria based on program analysis requirements, which can be variable definitions, variable references, specific parameters of procedure calls, constants, and application program interfaces in the program.

3) Program slicing generation: Select the appropriate program slicing algorithm, and execute the program slicing algorithm based on the constructed slicing criteria, as shown in Fig. 10 below, to extract the code fragments of the program.

$CFG=\{V,E,Entry,Exit\}$

$PDG=(V_p,E_p)$

$SlicingCriterion=(n,\{variable\})$

$Slice\rightarrow\phi$

```
1: ReachableGraphSlice(Node n)
2:     IF n.visited==false
3:         n.visited=true
4:         Slice=∪ {n}
5:     FOR s IN n.children
6:         Slice=∪ ReachableGraphSlice(s)
7:     RETURN Slice
```

Fig. 10. Slicing algorithm based on graph reachability

The algorithm above describes the program slicing algorithm based on graph accessibility. The input parameters include control chart CFG, program dependency graph PDG, and slicing criterion Slicing Criterion. By recursively traversing the program dependency graph, a subgraph of the program dependency graph is generated. It is a program slice for the slice criterion.

4.2 Feature Extraction Method

Some instruction codes contained in malicious code slices are not found in normal files or are very few. These instructions that can be used to distinguish the two types of codes are called features. These features are used as feature vectors for analysis to form a training data sample set. Malicious code has many characteristics. The following mainly introduces two algorithms, N-gram and variable-length N-gram sliding window feature extraction.

4.2.1 N-gram Features

The N-gram model is called a first-order Markov chain. N-gram is a series of overlapping substrings collected by a sliding window of length N, and this window slides by a unit length each time. N-gram can capture some potential features that are difficult to accurately extract by other methods, and is used in many fields, such as text classification and information retrieval. In the field of malicious code detection, N-gram is a widely used feature extraction method. In 1994, a byte N-gram-based method was used to automatically extract virus features (Kephart 1994). The main difficulty encountered when using byte N-grams as features at that time was that the number of all byte N-grams extracted from all malicious code and normal code databases was too large.

Yang (1997) proposed several feature selection methods, selecting a small number of related features from all the extracted features to form a related feature set, which solved the problem of too large feature set. N-gram extraction features have two shortcomings: First, it is difficult for N-gram to capture byte sequences of different

lengths at the same time. When a meaningful byte sequence is not a multiple of N, there will be edge mismatches, which makes it impossible to extract this. feature. The second is that the feature set generated by N-gram is very large and requires considerable storage capacity.

4.2.2 Variable-length N-gram Features

Variable-length N-grams are also called paragraphs, which are a series of meaningful continuous byte sequences. Unlike N-grams, its length is not fixed, which avoids the possibility of a meaningful sequence being split. To extract meaningful paragraphs, you first need to find breakpoints in a series of byte sequences, and the continuous sequence between adjacent breakpoints is a paragraph. There have been many algorithms for segmentation, which have been applied to communication time series, speech processing, signal processing, text classification and other fields. The segmentation objects are not the same in different environments (Firoiu 2002). In malicious code detection, the segmented object is a sequence of bytes in the form of binary code.

5. Summary and Outlook

With the popularization of computer applications and the rapid development of network interconnection, the confidentiality and integrity of electronic information are increasingly threatened. Information security has become an important research field that has attracted much attention. The emergence and increase of malicious code has caused immeasurable losses to many commercial companies, government departments and people's daily lives. Malicious code detection technology has gradually become an important research direction in the field of information security.

This article mainly studies malicious code detection technology based on data mining and machine learning, introduces the basic knowledge of malicious code, related data mining and machine learning technical theories, and analyzes the current status of malicious code detection at home and abroad, and the current feature extraction And feature selection method, on this basis, the feature extraction method of variable length N-gram and the feature selection method of weighted information gain are proposed, and an important method in the field of program understanding-program slicing is summarized, and it is targeted at malicious code detection. In actual application scenarios, a malicious code detection system is designed, which can ensure the user's network security to a certain extent.

With the further development of information technology and network technology, new malicious codes continue to emerge. Combined with the current development trend of information security of malicious codes, I think the following work needs further study:

(1) Further improve the slicing criterion library that characterizes malicious code behavior, and extract program slices of key malicious behaviors in malicious code more accurately and comprehensively.

(2) The detection and defense of malicious code is a long-term process. Since there is no universal malicious code detection method, the generalized characteristics of malicious code need to be further explored, and tools to adapt to more malicious code detection and defense need to be further developed.

(3) Any security technology research has its two sides, so the fuzzy transformation strategy can also be implemented in the security fields of information system security, software encryption, attack and deception.

Acknowledgement

Source: Scientific research project "Research on Key Technology and Typical Application Scenarios of Railway Industry Internet Data Security Exchange and Sharing Service" (Project No.: 2021YJ203).

References

[1] J. Yang, J. Fan, J. Zhou, L. Gao, Android malware detection method based on behavior pattern, Journal of Frontiers of Computer Science and Technology, DOI: 10.3778/j.issn.1673-9418.2102048. < http://fcst.ceaj.org/CN/10.3778/j.issn.1673-9418.2102048>, 2021 (accessed 21.06.08).

[2] B. Fang, J. Shi, Z. Wang, W. Yu, AI-Enabled Cyberspace Attacks: Security Risks and Countermeasures, Strategic Study of CAE 23(3)(2021) 60-66.

[3] P. Chen, B. Han, H.J. Hong, JavaScript malicious code detection system based on deep learning and blockchain, Computer

Systems & Applications 30(5)(2021) 99-106.
[4] F. Xiao, Z.W. Lin, Y. Sun, Y. Ma, Malware Detection Based on Deep Learning of Behavior Graphs, Mathematical Problems in Engineering 2019(2019) 1-10.
[5] J. Li, L.C. Sun, Q.B. Yan, Z.Q. Li, W. Srisa-an, H. Ye, Significant Permission Identification for Machine- Learning-Based Android Malware Detection, IEEE Transactions on Industrial Informatics14(7)(2018) 3216-3225.
[6] F. Nielson, H.R. Nielson, C. Hankin, Principles of Program Analysis, Springer, Berlin Heidelberg, 1999.
[7] B. Korel, The Program Dependence Graph in Static Program Testing, Information Processing Letters 24(2)(1987) 103-108.
[8] J. Ferrante, K.J. Ottenstein, J.D. Warren, The Program Dependence Graph and Its Use in Optimization, ACM Transactions on Programming Languages and Systems 9(3)(1987) 319-349.
[9] B. Wood, An insider threat model for adversary simulation, in: R.H. Anderson, T. Bozek, T. Longstaff, W. Meitzler, M. Skroch, K. Van Wyk (Eds.), SRI International, Research on Mitigating the Insider Threat to Information Systems 2, RAND Corporation, Santa Monica, CA, 2000.
[10]D.B. Parker, Fighting computer crime: A new framework for protecting information, John Wiley & Sons, Inc., New York, 1998.
[11]J.S. Park, S.M. Ho, Composite role-based monitoring (CRBM) for countering insider threats, in: Proc. Intelligence and Security Informatics, 2004.
[12]J. Shetty, J. Adibi, The Enron email dataset database schema and brief statistical report, Information sciences institute technical report, University of Southern California, 2004.
[13]M.G. Schultz, E. Eskin, F. Zadok, S.J. Stolfo, Data Mining Methods for Detection of New Malicious Executables, in: Proc. 2001 IEEE Symposium on Security and Privacy, 2001.
[14]M. Weber, M. Schmid, M. Schatz, D. Geyer, A Toolkit for Detecting and Analyzing Malicious Software, in: Proc. 18th Annual Computer Security Applications Conference, 2002.
[15]M. Christodorescu, S. Jha, Static analysis of executables to detect malicious patterns, in: Proc. 12th USENIX Security Symposium (USENIX Security 03), 2003.
[16]B. Madhusudan, J. Lockwood, Design of a System for Real-Time Worm Detection, in: Proc. 12th Annual IEEE Symposium on High Performance Interconnects, 2004.
[17]T. Abou-Assaleh, N. Cercone, V. Keselj, R. Sweidan, N-gram-based detection of new malicious code, in: Proc. 28th Annual International Computer Software and Applications Conference, 2004.
[18]A.M. Fiskiran, R.B. Lee, Runtime execution monitoring (REM) to detect and prevent malicious code execution, in: Proc. IEEE International Conference on Computer Design, 2004.
[19]B. Cornelissen, A. Zaidman, A. van Deursen, L. Moonen, R.A. Koschke, A Systematic Survey of Program Comprehension through Dynamic Analysis, IEEE Transactions on Software Engineering 35(5)(2009) 684-702.
[20]M. Yu, J. Jiang, G. Li, C. Liu, W. Huang, N. Song, A Survey of Research on Malicious Document Detection, Journal of Cyber Security 6(3)(2021) 54-76.
[21]J. Kinder, H. Veith, Jakstab: A static analysis platform for binaries, in: Proc. International Conference on Computer Aided Verification, 2008.
[22]A. Paivio, Perceptual Comparisons through the mind's eye, Memory & Cognition 3(6)(1975) 635-647.