Malicious Code Detection Method Based on Static Features and Ensemble Learning

Wei Li¹, Chenyi Zhang¹*, Jieying Zhou², Dan Wang³, Nuannuan Li⁴
¹School of Control and Computer Engineering, North China Electric Power University, Beijing 102206, China
²state grid, zhou-jieying@sgcc.com.cn
³state grid henan electric power company, Henan 450018
⁴state grid henan electric power research institute, Henan 450000
*Corresponding author’s e-mail: zhangchenyi@ncepu.edu.cn

Abstract. In recent years, malicious code emerges in endlessly. New types of malicious code evade the traditional malicious code detection technology through polymorphism, shelling, confusion and other ways. In order to solve the challenges brought by various technologies to the research of malicious code detection, in this paper, we propose a malicious code detection method based on static features and ensemble learning. This method extracts the information in the PE header file of malicious code samples as static features, and on this basis, builds a malicious code detection model by using stacking ensemble learning. In order to verify the effectiveness of the model, experiments are carried out on a dataset containing 5000 malicious codes and 4943 benign codes. Experiments show that the classification model based on stacking ensemble learning is the best, with 97.22% precision rate and 96.45% stable F1 score.

1. Introduction
Malicious code is a kind of harmful computer code or network script designed to create system vulnerabilities, leading to backdoors, security vulnerabilities, information and data theft and other potential damage to file and computer systems. According to the Kaspersky report, 19.8% of users' computers have been attacked by malicious software at least once in the past year. In the first quarter of 2019 alone, there were more than 150 million malware attacks, with an average of 1.6 million per day, an increase of 108% over the first quarter of 2018. The proliferation of malicious code and its number has posed a serious threat to the computer and Internet security, and presents an increasingly serious trend.

The traditional way of anti-virus engine to detect malicious code is to rely on a known malicious code recognizable fragments of the feature database, namely the virus signature library, and then through the static scanning of the file to be detected, find the specified hexadecimal string in the file, if found, the file to be tested is malicious code, otherwise it is not malicious code. In this way, the performance of anti-virus engine completely depends on the virus signature library. If the virus signature is too general, there will be a high rate of false positives. Consider that a large part of the reason why we close anti-virus software is that anti-virus software judges normal software as malicious; if the virus signature is too specific, many variants of the same kind of malicious code may not be detected, leading to false positives. In addition, the delay between the signature of malicious code and the malicious code entering
the computer system and attacking it is a fatal disaster for the computer. These are the problems of using anti-virus engine to detect malicious code. In recent years, machine learning has been successfully applied in many fields. Through research and experiments, machine learning can also achieve good results in malicious code detection.

In this paper, we propose a malicious code detection method based on static features and ensemble learning. This method is based on static analysis method, extracts the PE header file data of malicious code samples as features, and detects malicious code through ensemble learning algorithm in machine learning.

The main work of this paper is as follows:

- PE header file data is extracted from malicious code samples as features.
- Machine learning classification process is used to process and transform the features of PE header file data, and the prediction model is established.
- Malicious code is detected by ensemble learning algorithm in machine learning.
- In order to evaluate the effectiveness of the method and the influence of parameters, experiments were carried out, and the experimental results achieved 97.22% precision rate and 96.45% stable $F_1$ score.

2. Related work

At present, the commonly accepted malicious code detection methods include signature based detection, behavior based detection and heuristic detection. In practice, the signature feature library of malicious code is usually necessary to detect malware. The corresponding method can perform well only when the malicious code to be detected is fully known and documented. Signature based detection method creates a unique label for each malicious code, so that future malicious code samples can be detected correctly, and the error rate is very small, but these methods can not be well extended to detect new malicious binary files. In order to overcome the shortcomings of signature based detection, a detection method based on malicious code features is proposed to obtain better detection performance. The researchers have introduced heuristic detection method and machine learning detection method. At present, the latter has become a hot research topic in malware detection. In the malicious code detection based on machine learning, the features are extracted in binary form, which fully describes the differential representation of the program. Then, the samples are divided into malicious code and benign code by using the classification algorithm of machine learning. According to whether the program is executed when extracting feature representation, it is mainly divided into extracting static malicious features for feature analysis and dynamic malicious features for feature analysis.

In malicious code detection based on static feature analysis, such as collecting and researching graphic images, printable strings and other binary file headers, suspicious patterns can be found. Based on the sequence of opcode, a new method of malicious software is proposed in the paper [1], which can construct vector representation of executable file and detect malicious code based on machine learning algorithm. Based on n-gram and SVDD, a method sspv SVDD is proposed to detect unknown malicious code in the paper [2]. The experimental results show that the recognition accuracy of the method to a single malicious code family can reach 97%. In this paper, a method of classifying the shell algorithm of executable file is proposed in document [3], which can detect malicious code. The method first scales the entropy of a given executable file and converts the entropy of a specific storage location to a symbolic representation using symbol aggregation approximation (sax). Secondly, the supervised learning classification method, namely naive Bayesian and support vector machine, is used to detect the shell adding algorithm, and the distribution of symbols is classified. The experimental results show that the method can reach 95.35% accuracy and 95.83% recall rate. Based on the visualization method, the document [4,5] transforms malicious code into image and uses neural network to train and process, so as to realize the classification of malicious code.

In the malicious code detection based on dynamic feature analysis, the program is executed in a closed controlled environment, and the execution track of unknown code can be observed and recorded for further exploration. The document [6] builds a sub graph based on the instruction tracking record...
dynamically collected from the target executable file, and proposes a new malware detection algorithm. These graphs represent Markov chains, where the vertex is instruction, and the combination of graph cores is used to create similarity matrix between instruction tracking graphs. Finally, similarity matrix is sent to support vector machine for classification. The document [7] extracts the features based on the dynamic instruction sequence n-gram, and implements the malicious code classification based on K-means and EM clustering algorithm. The experimental results show that EM clustering is better than K-means and all of them reach more than 90% classification accuracy.

For a long time, because of the well-known defects of dynamic behavior detection methods, such as the detection time-consuming, the use of system resources, and the high false alarm rate, these make the malicious code detection based on dynamic feature analysis is especially prominent when it is applied to large-scale data. In conclusion, we select malicious code detection based on static analysis, extracts the static features of malicious code, and propose a malicious code detection method based on static features and ensemble learning by means of machine learning ensemble learning.

3. Methodology

3.1. Feature extraction
In order to detect malicious code through machine learning algorithm, it is necessary to extract the effective information of malicious code accurately, and express it in vectorization and characterization. In this paper, we focus on the detection of malicious code based on static features, mainly focus on the PE header file data of malicious code, and extracts features from it.

Portable Execute (PE) file is the general name of executable files under windows, common are DLL, EXE, OCX, SYS and so on. No matter before or now, most of malicious code exists in the form of PE file. Wikipedia defines PE file as a file format for executable file, object file and dynamic link library. It is mainly used on 32-bit and 64 bit windows operating system.

PE files have one common feature: the first two bytes are 4D 5A (MZ). If the first two bytes of a file are not 4D 5A, it is definitely not an executable file. It can be said that PE file format is a kind of data structure. PE file format encapsulates some necessary information when Windows operating system loads executable program code. These information include dynamic link library, API import and export table, resource management data and thread local storage data, etc. PE file structure: Dos header + PE header + section table + .data / .rdata / .text, in this paper, we focus on the analysis of PE header file information.

In this paper, we extract information from PE header file as feature data of malicious code detection, and some selected features and meanings are shown in Table 1.

| Features             | Meaning                                                                 |
|----------------------|-------------------------------------------------------------------------|
| AddressOfEntryPoint  | the address of program entry point                                       |
| DebugSize            | the size of the debug-directorytable                                     |
| ImageVersion         | the version of the file                                                  |
| IatRVA               | the relative-virtual address of the import-address table                 |
| ExportSize           | the size of the export table                                             |
| ResourceSize         | the size of the resource section                                         |
| NumberOfSections     | the number of sections                                                   |
| ImageBase            | the base address that the program loads by default                       |
| OSVersion            | the major version number that requires the minimum version number of the operating system |
3.2. Stacking ensemble learning

There is a kind of algorithm called ensemble learning in machine learning. In supervised learning, ensemble learning is to combine multiple weak supervised models in order to get a better and more comprehensive strong supervised model. The potential idea of ensemble learning is that even if one weak classifier gets the wrong prediction, other weak classifiers can correct the error. Ensemble learning algorithms are divided into three categories: boosting, bagging and stacking [8]. In this paper, stacking ensemble learning algorithm is selected to build the malicious code detection model. The basic idea of stacking ensemble learning algorithm is to train a model to combine other models, that is, to train multiple different models first, and then to train a model with the output of each model trained before as the input to get a final output. In the stacking ensemble learning algorithm to solve the classification problem, a single classifier is combined by classifiers. A single classifier can be called a base classifier, while a combined classifier is called a meta classifier. For stacking ensemble learning algorithm, the detection effect mainly depends on two aspects: on the one hand, the detection accuracy of each base classifier, the higher the accuracy of a single classifier, the better the learning effect of stacking ensemble learning classifier; on the other hand, the diversity of base classifiers, without affecting the error rate of base classifiers, the difference of base classifier detection will increase the accuracy of stacking ensemble learning classifier.

The specific steps of stacking ensemble learning are as follows:

The first is to use a basic model for 5-fold cross validation. For example, using xgboost as the basic model model1, 5-fold cross validation is to take 4-fold as training data and another one as testing data. Note: in stacking, this part of data will use the whole tracking set. For example, suppose that our entire training set contains 10000 rows of data and the testing set contains 2500 rows of data, then each cross validation is actually dividing the training set. In each cross validation, the training data will be 8000 rows and the testing data will be 2000 rows.

Each cross validation includes two processes: 1. Training data based training model; 2. Testing data prediction based on training model. After the first cross validation, we will get the predicted value of the current testing data, which will be a one-dimensional 2000 rows of data, denoted as A1. After this part of the operation is completed, we also need to predict the entire original testing set of the dataset. This process will generate 2500 predicted values, which will be recorded as B1 as a part of the next layer model testing data. Because we are carrying out a 50% cross validation, the above mentioned process will be carried out five times, and finally five columns and 2000 rows of data A1, A2, A3, A4, A5 will be generated for the prediction of testing set data, and five columns and 2500 rows of data B1, B2, B3, B4, B5 will be generated for the prediction of testing set data. After completing the whole steps of model 1, we can find that A1, A2, A3, A4 and A5 are actually the predicted values of the original training set. If they are pieced together, a 10000 row column matrix will be formed, which is called A1. For the data of B1, B2, B3, B4 and B5, we add all the parts and take the average value to get a matrix of 2500 rows and columns, which is recorded as B1. The above is the complete process of a model in stacking. The same layer in stacking usually contains multiple models. Hypothetically, there are model2: LR, Model3: RF, model4: GBDT, Model5: SVM. For these four models, we can repeat the above steps. After the whole process, we can get new A2, A3, A4, A5, B2, B3, B4, B5 matrices.

After that, we merge A1, A2, A3, A4, A5 side by side to get a 10000 row and five column matrix as training data. B1, B2, B3, B4, B5 side by side to get a 2500 row and five column matrix as testing data. Let the next layer of models be based on them for further training. The above is the complete steps of stacking!

In the first layer of stacking detection method, the base classifier uses three machine learning algorithms with high detection accuracy and great difference, namely Naive Bayes, XGBoost and
Logistic Regression. The second layer uses a simple Logistic Regression classifier as a meta classifier, which can reduce the probability of over fitting and control the complexity.

4. Test Results and Discussions

4.1. Experiment preparation
Microsoft’s windows operating system has more than one billion enterprise and individual customers in the world. It is the most widely used operating system at present, which makes it the most vulnerable to network attacks. Therefore, we select 9943 samples of malicious code and benign code on Windows platform, including 5000 samples of malicious code, from https://virusshare.com. There are 4943 benign code samples, all of which are PE header files of windows system after security detection.

4.2. Method validation and evaluation
In this paper, we aim to detect malicious code, which is a typical binary classification problem. For the classification problem, a very intuitive model evaluation method is confusion matrix. The confusion matrix of malicious code detection is shown in the following table:

| Actual | Predicted |                |
|--------|-----------|----------------|
|        | Benign    | Malicious      |
| Benign | TP(True Positive) | FN(False Negative) |
| Malicious | FP(False Positive) | TN(True Negative) |

In this paper, we use the Precision rate (P), Recall rate (R), and comprehensive evaluation index (F1 score) to evaluate the model. The formula is as follows:

\[
P = \frac{TP}{TP + FP} \tag{1}
\]

\[
R = \frac{TP}{TP + FN} \tag{2}
\]

\[
F_1 = \frac{2 \times P \times R}{P + R} \tag{3}
\]

4.3. Performance of classifiers
The main purpose of the experiment is to compare the performance of several models in malicious code detection tasks, including machine learning model, Naive Bayesian model, XGBoost model, Logistic Regression model and Stacking Ensemble Learning model. Firstly, the important features in the data set are extracted from the PE header file, and then the machine learning algorithm is used for training and prediction.

In this experiment, we use python and sklearn to train the model. The machine learning model includes Naive Bayesian, XGBoost, Logistic Regression and Stacking Ensemble Learning to compare and select the model with the best experimental results. The performance of each single classifier are listed in the following table:

| Model                      | P   | R     | F1   |
|----------------------------|-----|-------|------|
| Naive Bayesian             | 55.69% | 99.18% | 71.33% |
| XGBoost                    | 93.94% | 94.39% | 94.16% |
| Logistic Regression        | 85.07% | 73.72% | 78.99% |
| Stacking Ensemble Learning | 97.22% | 95.69% | 96.45% |

From Table, it can be seen that the feature extraction from PE header file and the stacking ensemble learning algorithm are better than the other three algorithms in overall performance.
5. Conclusion
In this paper, we propose a malicious code detection method based on static features and ensemble learning. In this paper, some data in PE header files are selected as the features of malicious code and benign code, and the machine learning classification process is used to process and transform the data features of PE header files, and the prediction model is established. In this paper, ensemble learning stacking algorithm in machine learning is used to train the model. Naive Bayes, XGBoost and Logistic Regression algorithm are used in the first level classifier, and simple Logistic Regression algorithm is used in the second level meta classifier to prevent over fitting. In this paper, experiments are carried out on a large number of data sets. The experimental results show that the ability of malicious code detection based on stacking ensemble learning is better than other models. It also shows that the feature extraction method based on static analysis is effective, which provides a new idea for malicious code detection.

Future work can consider how to extract more features from malicious code and design and utilize models with better performance.

Acknowledgments
This work was supported by the Science and Technology Project of the State Grid Corporation of China under Grant 5700-202024193A-0-0-00.

References
[1] Santos I., Brezo F., Ugarte-Pedrero X., et al. (2013) Opcode sequences as representation of executables for data-mining-based unknown malware detection. Information Sciences, 231: 64-82.
[2] Boujnouni, M. E., Jedra, M., Zahid, N. (2016) New malware detection framework based on N-grams and Support Vector Domain Description. In: International Conference on Information Assurance and Security. IEEE, 123-128.
[3] Bat-Erdene M., Park H., Li H., et al. (2017) Entropy analysis to classify unknown packing algorithms for malware detection. International Journal of Information Security, 16(3): 227-248.
[4] Zeng Y.Q., Zhang L.L., Zhang R.N, Yang B. (2020) Malware Family Classification Model Based on MobileNet. Computer Engineering, 46(4): 162-168.
[5] Wang B., Cai H.H., Su Y. (2019) Classification of malicious code variants based on VGG net. http://kns.cnki.net/kcms/detail/51.1307.TP.20190924.1107.006.html.
[6] Anderson B., Quist D., Neil J., et al. (2011) Graph-based malware detection using dynamic analysis. Journal in computer Virology, 7(4): 247-258.
[7] Pai S., Di Troia F., VisaggioC A., et al. (2017) Clustering for malware classification. Journal of Computer Virology and Hacking Techniques, 13(2): 95-107.
[8] YU Y., LONG J., CAI Z.P. (2017) Network Intrusion Detection through Stacking Dilated Convolutional Autoencoders. Securith and Communication Network, 2017(10): 1-11.