Volume 3, Issue 5, May 2013 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Malware Classification based on Clustering and classification

Dr.A.Kumaravel
Prof& Dean (Department of Computer Science and Engineering)
Bharath University
Chennai, India

A.Aarthi
B.Tech Student (Department of Computer Science and Engineering)
Bharath University
Chennai, India

***Abstract:** Malware, short for malicious software, means a variety of forms of hostile, intrusive, or annoying software or program code. Malware is a pervasive problem in distributed computer and network systems. Malware variants often have distinct byte level representations while in principal belong to the same family of malware. The byte level content is different because small changes to the malware source code can result in significantly different compiled object code. Entropy analysis initially determines if the binary has undergone a code packing transformation. If packed, dynamic analysis employing application level emulation reveals the hidden code using entropy analysis to detect when unpacking is complete. A similarity search is performed on the malware database to find similar objects to the query. Additionally, a more effective approximate flow graph matching algorithm is proposed that uses the de compilation technique of structuring to generate string based signatures amenable to the string edit distance. We use real and synthetic malware to demon strate the effectiveness and efficiency of Malwise.*

***Keyword:** polymorphic,malware,Malwise,object code.*

I. INTRODUCTION

To classify the packed and polymorphic malware, we proposes a novel system,Named Malwise, for malware classification using a fast application level emulator to reverse the code packing transformation, and two flowgraph matching algorithms to perform classification. Malware is a pervasive problem in distributed computer and network systems. Malware variants often have distinct byte level representations while in principal belong to the same family of malware. The byte level content is different because smallchanges to the malware source code can result in significantlydifferent compiled object code. In this project wedescribe malware variants with the umbrella term ofpolymorphism

II. RELATED WORK

Our approach employs both dynamic and static analysis to classify malware. Entropy analysis initially determines if the binary has undergone a code packing transformation. If packed, dynamic analysis employing application level emulation reveals the hidden code using entropy analysis to detect when unpacking is complete. If not, then Static analysis then identifies characteristics, building signatures for control flow graphs in each procedure. A similarity search is performed on the malware database to find similar objects to the query. Additionally, a more effective approximate flow graph matching algorithm is proposed that uses the decompilation technique of structuring to generate string based signatures amenable to the string edit distance. We use real and synthetic malware to demonstrate the effectiveness and efficiency of Malwise.

III. CONCEPT

In this project, we will develop a spam zombie detection system, named SPOT, by monitoring outgoing messages. SPOT is designed based on a statistical method called Sequential Probability Ratio Test (SPRT) which has bounded false positive and false negative error rates.

IV. ALGORITHM

NAÏVE BAYESIAN CLASSIFIER

Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem (from Bayesian statistics) with strong (naive) independence assumptions. A more descriptive term for the underlying probability model would be "independent feature model".

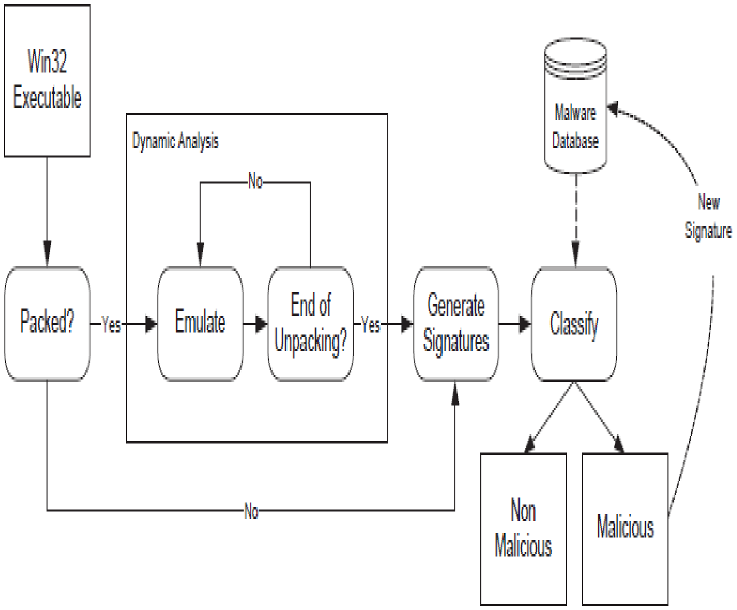
In simple terms, a naive Bayes classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 4" in diameter. Even if these features depend on each other or upon the existence of the other features, a naive Bayes classifier considers all of these properties to independently contribute to the probability that this fruit is an apple.

Depending on the precise nature of the probability model, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without believing in Bayesian probability or using any Bayesian methods.

RANDOM FOREST ALGORITHM

It is unexcelled in accuracy among current algorithms. For many data sets, it produces a highly accurate classifier.It runs efficiently on large data bases. It can handlethousands of input variables without variable deletion. It gives estimates of what variables are important in the classification. It generates an internal unbiased estimate of the generalization error as the forest building progresses. It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data are missing. It has methods for balancing error in class population unbalanced data sets. Generated forests can be saved for future use on other data. Prototypes are computed that give information about the relation between the variables and the classification.It computes proximities between pairs of cases that can be used in clustering, locating outliers, or (by scaling) give interesting views of the data. The capabilities of the above can be extended to unlabeled data, leading to unsupervised clustering, data views and outlier detection. It offers an experimental method for detecting variable interactions. Learning is fast

SYSTEM ARCHITECTURE



MODULE DESCRIPTION

1. *Data Collection*

Our data set consists of 100 binaries out of which 90 are benign and 10 are spyware binaries. The benign files were collected from Download.com, which certifies the files to be free from spyware. The spyware files were downloaded from the links provided by SpywareGuide.com. This hosts information about different types of spyware and other types of malicious software.

2. *Byte Sequence Generation*

We have opted to use byte sequences as data set features in our experiment. These byte sequences represent fragments of machine code from an executable file. We use xxd, which is a UNIX-based utility for generating hexadecimal dumps of the binary files. From these hexadecimal dumps we may then extract byte sequences, in terms of *n*-grams of different sizes.

3. *Feature Extraction*

The output from the parsing is further subjected to feature extraction. We extract the features by using following approaches, the Common Feature-based Extraction (CFBE) and Frequency-based Feature Extraction. The occurrence of a feature and the frequency of a feature. Both methods are used to obtain Reduced Feature Sets (RFSs) which are then used to generate the ARFF files.

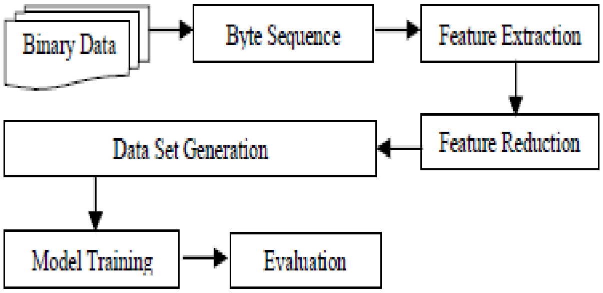
4. Dataset Generation

Two ARFF databases based on frequency and common features were generated. All input attributes in the data set are represented by Booleans. These ranges are represented by either 1 or 0.

5. Classification

A Naive Bayes classifier is a probabilistic classifier based on Bayes theorem with independence assumptions, i.e., the different features in the data set are assumed not to be dependent of each other. This of course, is seldom true for real-life applications. Nevertheless, the algorithm has shown good performance for a wide variety of complex problems. J48 is a decision tree-based learning algorithm. During classification, it adopts a top-down approach and traverses a tree for classification of any instance. Moreover, Random Forest is an ensemble learner. In this ensemble, a collection of decision trees are generated to obtain a model that may give better predictions than a single decision tree.

Process



EXPERIMENTAL AND RESULT

To detect the completion of unpacking, we proposed and evaluated the use of entropy analysis. It was shown that our system can effectively identify variants of malware in samples of real malware. It was also shown that there is a high probability that new malware is a variant of existing malware. Finally, it was demonstrated the efficiency of unpacking and malware classification warrants Malwise as suitable for potential applications including desktop and Internet gateway and Antivirus systems.

VII. CONCLUSION

Malware can be classified according to similarity in its flow graphs. This analysis is made more challenging by packed malware. In this paper we proposed different algorithms to unpack malware using application level emulation. We also proposed performing malware classification using either the edit distance between structured control flow graphs, or the estimation of isomorphism between control flow graphs. We implemented and evaluated these approaches in a fully functionally system, named Malwise. The automated unpacking was demonstrated to work against a promising number of synthetic samples using known packing tools, with high speed

REFERENCES

[1] Symantec, "Symantec internet security threat report: Volume XII," Symantec2008.

[2] F-Secure. (2007, 19 August 2009). F-Secure Reports Amount of Malware Grew by 100% during 2007. Available: http://www.fsecure.com/en_EMEA/aboutus/pressroom/news/2007/fs_news_20071204_1_eng.html

[3] K. Griffin, S. Schneider, X. Hu, and T. Chiueh, "Automatic Generation of String Signatures for Malware Detection," in *Recent Advances in Intrusion Detection: 12th International Symposium, RAID 2009, Saint*-Malo, France, 2009.

[4] J. O. Kephart and W. C. Arnold, "Automatic extraction of computer virus signatures," in *4th Virus Bulletin International Conference, 1994, pp.* 178-184.