Expert Systems With Applications 52 (2016) 16–25

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Malicious sequential pattern mining for automatic malware detection

Yujie Fan[a], Yanfang Ye[b], Lifei Chen[a,c,*]

[a] School of Mathematics and Computer Science, Fujian Normal University, Fuzhou, China
[b] Department of Computer Science and Electrical Engineering, West Virginia University, Morgantown, USA
[c] Department of Computer Science, University of Sherbrooke, Sherbrooke, Canada

ARTICLE INFO

Keywords:
Malware detection
Instruction sequence
Sequential pattern mining
Classification

ABSTRACT

Due to its damage to Internet security, malware (e.g., virus, worm, trojan) and its detection has caught the attention of both anti-malware industry and researchers for decades. To protect legitimate users from the attacks, the most significant line of defense against malware is anti-malware software products, which mainly use signature-based method for detection. However, this method fails to recognize new, unseen malicious executables. To solve this problem, in this paper, based on the instruction sequences extracted from the file sample set, we propose an effective sequence mining algorithm to discover malicious sequential patterns, and then All-Nearest-Neighbor (ANN) classifier is constructed for malware detection based on the discovered patterns. The developed data mining framework composed of the proposed sequential pattern mining method and ANN classifier can well characterize the malicious patterns from the collected file sample set to effectively detect newly unseen malware samples. A comprehensive experimental study on a real data collection is performed to evaluate our detection framework. Promising experimental results show that our framework outperforms other alternate data mining based detection methods in identifying new malicious executables.

1. Introduction

Malware, short for ***mal***icious soft***ware***, is software that design to damage or destruct computers without owners' permission (Schultz, Eskin, Zadok, & Stolfo, 2001). Due to the rapid development of information technology, malware has posed a serious threat to networks as well as computer systems. For instance, worm has increasingly threaten the hosts and services by exploiting the vulnerabilities of the largely homogeneous deployed software base (Sun & Chen, 2009). In addition, in the application of the online transaction, trojan horses often steal sensitive information from online users through website phishing (Abdelhamid, Ayesh, & Thabtah, 2014). Due to the enormous loss and adverse effect cause by malware, malware detection is one of the cyber security topics that are of great interests.

To protect legitimate users from the attacks, the most significant line of defense against malware is anti-malware software products, which mainly use signature-based method for detection (Griffin, Schneider, Hu, & Chiueh, 2009; Kephart & Arnold, 1994). In these scanning tools, unique signatures (a set of short and unique strings) are extracted from already known malicious files. Then, an executable file is identified as a malicious code if its signature matches with the list of available signatures. Such simple approach is fast to identify known malware with small error rate. However, extracting signature is a tough work which requires a great deal of time, funds and more importantly, the expertise. This is the main disadvantage of this method. The second issue is that signature-based method is restricted to recognize already known malware, and thus it is unreliable and ineffective against the new, unseen malicious codes. In fact, simple obfuscation techniques can easily bypass such signatures-based detection. Besides, driven by the economic benefits, today's malware samples are created at a high speed (thousands per day). For example, Symantec reported that 21.7 million new pieces of malware were created in October 2015 (Symantec, 2015); according to McAfee Labs threat report, there were more than 400 million total malware samples in the first quarter of 2015 (McAfee Labs, 2015).

In order to solve the above-mentioned problems, heuristic-based detection method, which utilizes data mining as well as machine learning techniques, is developed to conduct intelligent malware detection. This approach aims to learn special patterns that capture the characteristics of malware. Generally, its detection process can be divided into two phases: feature extraction and classification. In the first phase, various features are extracted from malware samples via static analysis or dynamic analysis to

* Corresponding author at: School of Mathematics and Computer Science, Fujian Normal University, China. Tel.:+8659122868128.
E-mail addresses: kobefyj@126.com (Y. Fan), yanfang.ye@mail.wvu.edu (Y. Ye), clfei@fjnu.edu.cn (L. Chen).

http://dx.doi.org/10.1016/j.eswa.2016.01.002
0957-4174/

represent the file; based on the extracted features, classification techniques are applied to identify the malware automatically. For instance, Schultz et al. (2001) extracted three different types of features (i.e., system resource information, printable strings and byte sequences) from the files, then used as inputs for Ripper, Naive Bayes and Multi-Naive Bayes to classify malware and benign files.

Since Application Programming Interface (API) calls can well represent the actions of an executable, it is one of the most effective features used by the heuristic-based methods. Many researches have been done based on API calls, including Hofmeyr, Forrest, and Somayaji (1998), Ye, Wang, Li, Ye, and Jiang (2008) and so forth. There are some other researchers applying another meaningful feature (i.e., the machine instructions) to detect malware, such as Santos et al. (2010), Shabtai, Moskovitch, Feher, Dolev, and Elovici (2012) and Runwal, Low, and Stamp (2012). Although these works demonstrate desirable detection results, they did not take the order of the features into consideration and thus fail to mine patterns with notable difference between malicious files and benign files.

In this paper, we propose a new sequence mining algorithm to discover malicious sequential patterns based on the machine instruction sequences extracted from the Windows Portable Executable (PE) files, then use it to construct a data mining framework, called MSPMD (short for **M**alicious **S**equential **P**attern based **M**alware **D**etection), to detect new malware samples. The main contributions of this paper can be summarized as follows:

- *Well represented feature for malware detection*: Instruction sequences are extracted from the PE (Portable Executable) files as the preliminary features, based on which the malicious sequential patterns are mined in the next step. The extracted instruction sequences can well indicate the potential malicious patterns at the micro level. In addition, such kind of features can be easily extracted and used to generate signatures for the traditional malware detection systems.
- *Effective malicious sequential pattern mining algorithm*: We propose an effective sequential pattern mining algorithm, called MSPE (**M**alicious **S**equential **P**attern **E**xtraction), to discover malicious sequential patterns from instruction sequence. MSPE introduces the concept of objective-oriented to learn patterns with strong abilities to distinguish malware from benign files. Moreover, we design a filtering criterion in MSPE to filter the redundant patterns in the mining process in order to reduce the costs of processing time and search space. This strategy greatly enhances the efficiency of our algorithm.
- *All-Nearest-Neighbor (ANN) classifier for malware detection*: We propose ANN classifier as detection module to identify malware. Different from the traditional k-nearest-neighbor method, ANN chooses k automatically during the algorithm process. More importantly, the ANN classifier is well-matched with the discovered sequential patterns, and is able to obtain better results than other classifiers in malware detection.
- *Comprehensive experimental studies*: We conduct a series of experiments to evaluate each part of our framework and the whole system based on real sample collection, containing both malicious and benign PE files. The results show that MSPMD is an effective and efficient solution in detecting new malware samples.

The remainder of this paper is organized as follows: Section 2 introduces the related work. In Section 3, an overview of MSPMD is presented. Section 4 describes the method for instruction sequence feature extraction. Section 5 presents the proposed algorithm for malicious sequential pattern mining. Section 6 describes the ANN classifier for malware prediction based on the mined malicious sequential patterns. Experimental results are presented in Section 7. Finally, Section 8 concludes.

2. Related work

Signature-based method is widely used in anti-malware industry for malware detection (Griffin et al., 2009). However, this classic method always fails to detect variants of known malware or previously unseen malware. The problem lies in the signature extraction and generation process, and in fact these signatures can be easily bypassed (Ye et al., 2008). For example, to evade the widely-used signature-based detection, malware developers can employ techniques such as polymorphism and metamorphism (Jain & Bajaj, 2014). Not only the diversity and sophistication of malware have significantly increased in recent years, driven by economic benefits, today's malware samples are also created at a rate of thousands per day (McAfee Labs, 2015; Symantec, 2015). In order to remain effective, anti-malware industry calls for intelligent malware systems which can automatically detect newly unseen malware from the collected file samples. Many research efforts have already been conducted on developing intelligent malware detection systems applying data mining techniques. Such data-mining-based detection methods require a feature extraction process to mine some features. Actually, the performance of the detection method mainly depends on what the features are extracted from the executables. More specifically, if the extracted features are well representative, it is expected to obtain better result when using these features to detect malware. Over the past few years, API calls and machine instructions are two of the most widely used features (Bazrafshan, Hashemi, Fard, & Hamzeh, 2013). Besides these, there also exists many researches relying on other features for malware detection, such as byte code (Nissim, Moskovitch, Rokach, & Elovici, 2014), data flow graph (Wchner, Ochoa, & Pretschner, 2014), Dynamic Link Libraries (DLLs) (Narouei, Ahmadi, Giacinto, Takabi, & Sami, 2015).

API calls represent the requests of windows executables on operate system. Due to their effectiveness to reflect the actions of executable, API calls are considered to be one of the most attractive features for detecting malware. The first attempt to use API as a feature of program was Hofmeyr et al. (1998). They presented a method for anomaly intrusion detection based on sequences of system calls. In their work, normal behavior was defined in short sequences of system calls executed by program. Then, three measures were used to detect abnormal behavior as deviations from the normal behavior. The representative research on API calls has been done by Ye et al. (2008). They developed an intelligence malware detection system (IMDS): it first extracted the API calls from each sample; then an objective-oriented association (OOA) mining algorithm was employed to generate OOA rules; finally it applied Classification Based on Association (CBA) (Bing, Wynne, & Ma, 1998) to build the classifier for malware detection. The experimental results showed that IMDS outperformed the signature-based methods and other data-mining-based methods in terms of detection rate and classification accuracy. Another interesting work using API calls for malware detection was Ahmadi, Sami, Rahimi, and Yadegari (2013), which was a dynamic malware detection system. They employed the iterative pattern mining method (Lo, Cheng, Han, Khoo, & Sun, 2009) to extract frequent iterative patterns and used Fisher score to conduct feature selection. The experiment results showed that high detection rate with low false alarm can be achieved when applying an iterative pattern mining approach. In very recent, Uppal, Sinha, Mehra, and Jain (2014) utilized the call grams and odds ratio selection to extract the distinct API sequences, then used as inputs to the classifiers to categorize malware and benign samples. Qiao, Yang, He, Tang, and Liu (2014) created a new representation method to transform API call sequences into byte-based sequential data for further