ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 5, May 2015

Malware detection using data mining techniques

Abhay pratap singh[1], Dr.S.S handa[2]

M.Tech Student (CSE) from Manav Rachana International University, Faridabad[1]

Professor, Computer Science from Manav Rachana International University, Faridabad[2]

Abstract: Nowadays, malicious software attacks and threats against data and information security has become a complex process. The variety and number of these attacks and threats has resulted in providing various type of defending ways against them, but unfortunately current detection technologies are ineffective to cope with new techniques of malware designers which use them to escape from anti-malwares. In current research, we present a combination of static and dynamic methods to accelerate and improve malware detection process and to enable malware detection systems to detect malware with high precision, in less time and help network security experts to react well since time detection of security threats has a high importance in dealing with attacks.

Keywords: Malware, Malware Detection, Escape Techniques, Data Mining

1. INTRODUCTION

The Continues growth of malwares, has resulted in creating enormous threats in information and security points so that cyber defense centers have high importance in many countries. Like country boundaries which could be attacked from different aspects such as contraband and thieves, virtual space also suffer from these attacks [1].

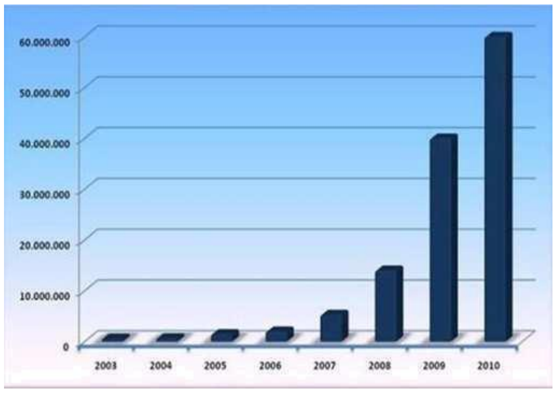

Figure 1. *Ncreased volume of malware from 2003 to 2010.*

Experiences have shown that most of these attacks are from malwares. On time detection of virtual space security attacks has a significant importance in protecting resources. In order to detect such malwares, before the advent of malicious effects, we should employ methods for detecting good and bad software behaviors to be able to detect which software is problematic and which ones are not. For this means, we should investigate both type of software in order to not face with a problem in detection process [2].

Figure 1 indicates increased volume of malware from 2003 to 2010 which has reported by Panda laboratory and it is predicted that this increasing trend of attack would continue in the next few years with a much faster speed so that the mean number of new threats per day exceeds from 55000 attacks per day. These attack are usually done to computer networks of sensitive agencies such as security entities, banks, economic centers, information storage centers, computer networks and etc.

2. MALWARE DEFINITION AND ANALYSIS

Computer applications which have a destructive content and apply to system from invader, are called malware and the systems which apply on it is called victim system [3]. The malware word is assigned to virus, worm, Trojan and any other program which is created for distractive goals and abusing of users' privacy.

But what is the difference between a virus and a worm? What is the difference between these two and Trojan? Do antivirus programs apply against worms and Trojans or only against the viruses? All of these questions originate from one source and it's the complex and complicated world of destructive codes [1].

Enormous numbers of available destructive codes have made their classification difficult. Generally, malwares are classified into several kind based on behavior, attack method: For example, some kind of malware classification is as follow: virus, worm, spyware, rootkit, each one has a special behavior which are described below:

2.1. Virus

A code which includes itself to other programs such as operating systems and needs to run within the host program [4].

2.2. Worm

Malwares which transform themselves from one system to other using self-publishing in a network which include some connected computers. Generally, viruses try to publish themselves via a program, while worms unlike viruses put themselves only in one computer, and try to pollute a computer network [1].

2.3. Trojan Horse

A type of malware that appears in the form of pieces of software code and are intended for useful purposes. It runs up desired functions for users but hiddenly runs a series of actions beside it. It even can destroy the integration of a system [3].

2.4. Logic Bomb

A Logic bomb does not publish itself, but is installed on a system and waits until an external event such as data input, reaches to a special date, creating, deleting or even modify a special file leading to damaging the system [2].

2.5. Backdoors

Backdoor is a kind of software which enters the computer system without authorization and achieves its goals without normal entering to system [1].

2.6. Spy

A term for a collection of software that collects user personal information such as most visited pages, email addresses, keys pressed by the user [5].

2.7. Rootkit

Rootkit is a malware that has the ability to hide itself and its activities on the target system. Owner of rootkit is capable to run file and settings on the victim system without the owner of system being aware of it. It usually attaches itself to original files of operating system core and run with it.

Rootkits try targeting original structures and programs of the operating system and the integrity of their contents in order to change performance trend and the result of their running. Rootkits can hide themselves from users through the following methods:

a) Rootkit integrate its codes with operating system codes which are at low-levels and accordingly can access all system requests such as reading files, running processes and etc.
b) Rootkit transfers its malicious codes into healthy processes and by doing so, it can use the memory that and do its malicious programs [6].

The base of traditional and usual methods to detect malware is using signature in which part of malware code is hold as the signature in the database and malware detection is carried out using signatures available in the database. Due to the failure of old methods in detecting new and unrecognized malwares or polymorphic malwares in recent years, researchers have tried to present more reliable methods for malware detection using unchanging characteristics of the malwares [6].

Nowadays, signature for antiviruses is a tool which is created manually. Before writing a signature, the analyst should identify how to deal with the unknown sample as a threat for users.

The process of searching malware is called analyzing. The more analysis tool and techniques, the more attackers try in using hidden making techniques and generating dynamic hidden codes from user's perspective. Analysts use two type of analysis to detect malware: static analysis and Dynamic analysis.

2.8. Static Analysis

Software analysis without execution, is called static analysis which without running the program, investigates the code and can detect malicious code and put it in one of the available groups based on different learning methods [7].

Since such methods deal with real codes, they can be used in the conditions in which there are polymorphic malwares. One of the problems of static analysis is that source code of the program isn't usually available which this reduces using of static analysis techniques that results in analyzing their binary codes which in turn is very complicated.

In the static method, binary codes are checked and viruses are detected based on binary codes. In fact this is the key part of static method. It is worth mentioning that extracting binary codes is a relatively complex work [5].

3. DYNAMIC ANALYSIS

To overcome these shortcomings, several dynamic detection methods have been proposed. Unlike the static method which relies on malware binary codes, there is a completely different method without using the codes but according to the runtime behavior [3].

Although promising, but unfortunately this method is too slow as real time detectors on the end host and often need virtual machine technology [1]. In fact, program analyzing, while it is running, is called dynamic analysis which also referred to as behaviors analyzing and include software running and watching its behavior, system interaction and its effects on host system [6]. Dynamic analysis method need to run polluted files in a virtual environment like a virtual machine, a simulator, sand box, etc to analyze it in virtual environment [2].

To analyze programs by dynamic methods, different techniques have been applied.

So far which the most common method and techniques include [8]:

Checking recalled functions.
Following the flow of
information.
Following the order of running functions.

4. MALWARE DETECTION TECHNIQUES

There are different methods to detect malwares but considering that malware have become more complicated using hidden techniques; we need more advanced methods to detect them.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 5, May 2015

Generally, common malware detection techniques are divided into two categories:

Detection methods based on
signature Detection methods based
on behavior

4.1. Signature- Based Detection

The main goal of this method is to extract the unique bytes sequence of codes as the signature. Searching for a signature in the suspicious files is a part of the task [8].

Most of today's commercial anti-malwares use a set of signatures to detect malicious programs which these suspicious codes are compared with a unique sequence of structures of programs or bytes [7].

If the signature is not available in the dataset, it means that the file is begin other than malicious [9].

The main problem of such approaches is that the anti-malwares experts should wait until new malware harm several computers, order to define a signature for it [8].

Usage of polymorphic model in cryptography has led to neutralize the signature based method which makes these polymorphic malwares undetectable through this method.

In order to overcome these problems, the behavior based method is used.

4.2. Behavior-Based Detection

Behavioral parameters include many factors such as source or destination of malware, kinds of attachments and other statistical properties [8]. Dynamic behaviors are directly used in evaluating the damage to the system and also help us to detect and classify new malwares. Malware clustering based on dynamic analysis is based on running the malware in a real controlled environment [7].

4.3. Comparison between Detection Methods

Given the polymorphism and transformation techniques which currently are used by malware designers, the signature based methods are inherently prone to errors [9].

Signature based methods are unable to detect more complex malwares and can hardly detect malwares which use polymorphism and transformation methods. In addition, one of the limitations of signature-based detection methods is that they require human knowledge to update the signature database by new signatures [8].

Furthermore, a number of research studies have shown that some of polymorphic software's writers can easily defeat signature based method by obfuscation methods [9].

Given the mentioned problems, it is better to use analysis method at runtime. However, the behavior based methods also have a major problem since this method is to slow as the real-time detectors on the final host and they often need virtual machine technologies.

5. METHODS USED FOR ESCAPING FROM ANTI MALWARES

Since signature-based antivirus systems try to find viral codes by searching for a character sequence string in the executive file, virus programmers apply various techniques to hide malwares and such sequences some of which are described below.

5.1. Cryptography

Virus code encryption by different encryption key would result in creating different texts.

As a result, it could be ensured that signature based scanners can't detect this virus. To run the virus, these texts should initially be decoded.

Detailed analysis of decoding algorithm is only possible if we know these keys [10].

5.2. Polymorphic Generator

Malwares use a polymorphic generator to change codes while the original algorithm remains intact. However, we should know that, at the end, all samples generated from a malware do the same work.

This is performed by combining many commands that have no impact on the execution mode and its effects. For example, each copy of the virus may be neutral group of commands such as increasing and then decreasing over the same operand or left ship and then right shift or push a value and pop it again.

All these methods will effectively hide virus codes from the signature based anti viruses [10].

5.3. Obfuscation

In malwares there are different evasion approaches to evade the malcodes from external anti malware scanners such as Code obfuscation, decrypting encryption and etc.

In code obfuscation the main goal is to hide the underlying logic of the program so as to prevent the others from having any related knowledge of the code[8].

The malicious code remains incomprehensible and all its harmful functionality whenever activated. When we apply some obfuscation transformations to a code, then it results in a kind of self-decrypting encryption.

But Packing refers to encrypt or compress the executable file. In Packing, original code remains hidden till the runtime or the unpacking process of executable codes which results in the immunity of code for static analysis [7].

Packed malware codes can be treated as subset of obfuscated codes which are compressed and cannot be analyzed so, consequently unpacking phase is necessary to reveal the overall semantic of the code [9].

6. PROBLEM DEFINITION

One the most important and most serious problems which the internet world is faced with is the existence of malwares like.

According to studies conducted in this field, we have concluded that 80 percent of damages to systems have been from malwares and only 20 percent of it has been from other factors [9].

However, unfortunately, most of the works has been on the 20% and the malwares have received less attention and thus we're facing many security problems every day [5].

In the early days of virus emergence, there were only static and simple viruses in the world [3].

Therefore, simple signature based methods were able to overcome them. But these methods were only useful as long as there weren't so many variations in the types of malwares and malwares writers didn't use obfuscation techniques to sophisticate them [5].

However, rapid developments in malwares activities convinced researchers to explore new methods, so that after some time, researchers were forced to use data-mining methods to detect malwares by employing data mining, they could add a lot of malware to anti-malware and hence they didn't have to investigate all malwares, because checking all of them require enormous time and cost [2].

One of such works was a method called n-grams. At that time, Geraldn et al. [3] developed n-grams analysis method to detect boot sector viruses using neural networks.

The base of n-grams detection method was the occurrence frequencies in the benign and malicious programs [3].

After that, Hofmeyr [10] used a simple sequence of system calls as a guide to evaluate malicious codes. This API CALLs sequence showed the hidden dependencies between code sequences.

Thereafter, Shultz, al. [7] tried to use the name of DLLs as a useful feature in the file categorization. However, in the recent work by Ye [7], a system (IMDS) was generated in which the system calls pattern has been used. Then data mining process has been applied on these patterns. The study includes 12214 healthy files and 17366 malicious files which they have only used 200 files to test the system [7].

Although the accuracy and learning rate of this method is relatively good, but there is a fundamental problem that is Unbalancing of the test data versus the balancing of learning data.

What we do in this study consists of a very large data set which involve various types of bengin and malicious softwares which generally, the number of extracted calls is about 5000 different features of 420 different files from 890 libraries which includes different types of malwares such as Trojan, Backdoor, Worm, Exploit, Flooder, Sniffer, Spoofer and viruses.

7. RESEARCH METHODOLOGY

This research has been performed by some basic steps:

data collection
data processing
analysis of results

In the following, we will discuss each of these steps.

7.1. Data Collection

In order to collect data related to malwares. We downloaded some malicious files form offensive computing [11].

Each sample of this set provides us its executive's code. These codes are used to learn the proposed model. In order to evaluate and test, a set of 3131 collected malware were tested which more than 90% of them include rootkits.

We selected this malwares set because in this study, our goal is detection rates of malwares especially rootkits.

7.2. Data Processing and Preparation

In this section, we deal with data processing using 3 reverse engineering tools namely: HDasm [12], Ida pro [13] and W32dsm89 [14] as well as Peid anti-packing tool [15].

First we process the Peid tool (which is the malware executive file) but with the understanding that the file has been packed by Packing tool. Otherwise, there is no need to apply this tool on it.

In fact by unpacking task, the packing task will be removed if it has been applied on it because otherwise, the file isn't executable by reverse engineering tool and thus we can't see the called system functions in it.

Afterwards, we give the file as input to three above-mentioned disassembler and they get the assembly code of these fields and return the called system functions list from these assembly codes. Then we save the list as an Xml file. Later, we apply our algorithm on this stored file to detect whether it is a malware or not and finally we obtain our success rate in detecting malwares using Weka data-minig tool.

7.3. Analysis of Results

Malwares of the same category usually have the samegeneral patterns, for example a number of system functions names are common in all members of this family.

We aim to analyze and detect malwares by examining the shared pattern using machine learning techniques among malwares.

In fact, we want to use so called Api calls in malware to

In fact by unpacking task, the packing task will be removed if it has been applied on it because otherwise, the file isn't executable by reverse engineering tool and thus we can't see

the called system functions in it.

Afterwards, we give the file as input to three above-mentioned disassembler and they get the assembly code of these fields and return the called system functions list from these assembly codes. Then we save the list as an Xml file. Later, we apply our algorithm on this stored file to detect whether it is a malware or not and finally we obtain our success rate in detecting malwares using Weka data-minig tool.

7.3. Analysis of Results

Malwares of the same category usually have the samegeneral patterns, for example a number of system functions names are common in all members of this family.

We aim to analyze and detect malwares by examining the shared pattern using machine learning techniques among malwares.

In fact, we want to use so called Api calls in malware to overcome the limitations of traditional signature based methods and to cope with techniques used by malwares writers as well as to increase malware detection rate.

This method, which is based on called system functions in malware executive code, uses reverse engineering tool and monitoring tool for static and dynamic analysis, respectively. This means, that we obtain their assembly code by disassembling them and then extract called system function in it and obtain the API CALLs list of malware executive file by monitoring the file using monitoring tool.

Finally, with respect to the shared sequence of maleware which is common among them and could be used to detect and identify them as the signature, we deal with the detection of malwares.

The advantages of this method include its high success rate in malwares detection because it is directly in contact with malware binary codes and also there is no need to run them and we can understand whether it is a malware or not only using their code and obtaining the shared sequence of called system functions.

Furthermore, we apply the prepared algorithm on the log file of each file to obtain our database.

After that, we transform the information of this database to a data mining tool (here we used Weka tool) to obtain the success rate of detection task.

Figure 2 shows a graph of data mining operation results using Weka tool on database. As shown above, the success rate of this method in rootkit detection is over than 97% which is a remarkable rate.

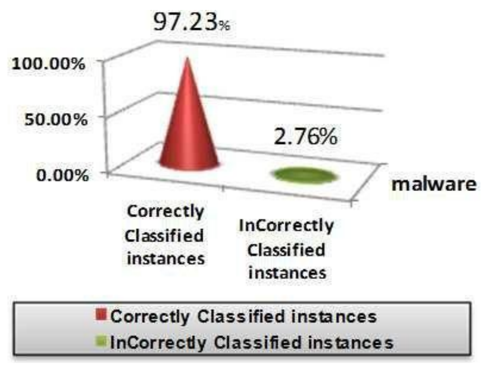

Figure 2. *Success rate of our method in rootkit detection*.

8. DISCUSSION AND CONCLUSION

Malwares are becoming widespread and more complex every day. As examples of their complexity, we can note the need of using polymorphism techniques, transformation and encryption, The traditional methods such as matching some code string of malwares signatures do not have enough efficiency.

However, there are also some problems in dynamic methods which their slowness is the most important one.

This is why we need a more intelligent detection method.

This type of detection (which is based on static method) is based on called system functions in each executive code of the malware and its goal is to detect versions of malware which haven't seen yet or are a new version of old malware families.

REFERENCES

[1] Ravi, C & Manoharan, R. Malware Detection using Windows Api Sequence and Machine Learning. International Journal of Computer Application, Vol.43, No.17, 2012.
[2] Ravi, C & Chetia, G. Malware Threats And Mitigation Strategies: A Survey, Journal of Theoretical and Applied Information Technology, Vol. 29, No. 2, pp. 69-73, 2011.
[3] Egele, M. S, A Survey on Automated Dynamic Malware-Analysis. ACM Computing Surveys, Vol. 44, No. 2, 2012.
[4] Herath, H. M. P. S., & Wijayanayake, W. M. J. I. Computer Misuse in the Workplace. Journal of Business Continuity & Emergency Planning, Vol.3, No.3, P.P 259–270, 2009.
[5] Mathur, K., and Saroj H. A Survey on Techniques in Detection and Analyzing Malware Executables. International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 44, No. 2, 2012.
[6] Doherty, N. F., Anastasakis, L., & Fulford, H, The Information Security Policy Unpacked: A Critical Study of the Content of University Policies. International Journal of Information Management, Vol.29, No.6, pp. 449–457, 2009.
[7] G. Tahan, L.R.Y. Automatic Malware Detection Using Common Segment Analysis and Meta-Features. Journal of Machine Learning Research, 13l, pp. 949-979, 2012.
[8] I. Gurrutxaga , Evaluation of Malware clustering based on its dynamic behaviour. Seventh Australasian Data Mining conference, Australia, pp. 163–170, 2008.
[9] Rieck. K, Willems.T, D¨ussel. P and Laskov. p, Learning and classification of malware behavior, 5th international conference on Detection of Intrusions and Malware, and Vulnerability Assessment. Berlin, Heidelberg: Springer-Verlag, pp. 108–125, 2008.
[10] Patel, S. C., Graham, J. H., & Ralston, P. A, Qualitatively Assessing the Vulnerability of Critical Information Systems: A New Method for Evaluating Security Eenhancements.

International Journal of Information Management, Vol.28, pp. 483–491, 2008.
[11] http:// *www.offensivecomputing.com*
[12] *http://hdasm.software.informer.com*
[13] www.hex-rays.com
[14] processchecker.com/file/W32dsm89.exe.html
[15] https://boveda.banamex.com.mx/englishdir/ayudas/masinfoahnlab.htm