PE-Header-Based Malware Study and Detection

Yibin Liao
Department of Computer Science
The University of Georgia, Athens, GA 30605
tigerlyb@uga.edu

***Abstract*—In this paper, I present a simple and faster apporach to distinguish between malware and legitimate .exe files by simply looking at properties of the MS Windows Portable Executable (PE) headers. We extract distinguishing features from the PE-headers using the structural information standardized by the Miscrosoft Windows operating system for executables. I use the following three methodology: (1) collect a large dataset of malware .exe and legitimate .exe from the two website, www.downloads.com and www.softpedia.com by using a Web-Spider, (2) use a PE-Header-Parser to extract the features of each header field, compare and find the most significant difference between malware and legitimate .exe files, (3) use a Icon-Extractor to extract the icons from the PE, find the most prevalent icons from the malware .exe files.**

I have evaluated our apporach on a large dataset which contains 5598 malware samples and 1237 legitimate samples respectively. The result of our experiments show that the PE-Header-Based approach achieves more than 99% detection rate with less than 0.2% false positive for distinguishing between benign and malicious executables in less than 20 minutes. We have also found 3 most prevalent icons from malware that are seldom seen in legitimate PE files, and 8 types of misleading icons from malware. My results show that it is possible to identify the malware by simply looking at some key features from PE headers.

***Index Terms*—Portable Executables; PE-Header-Based; Malware Detection;**

I. Introduction

Modern malware detectors employ a variety of detection techniques. Most of the detectors must determine the type of the file and then parse the file such as extract the contents and/or find the embedded item. Therefore the antivirus scanners need to parse a variety of formats, which make the antivirus more and more complex [1]. However, simplest malware can evade from sophisticated AVs [2]. PE is a file format which is standardized by the Microsoft Windows operating systems for executables, dynamically linked libraries (DLL), and object files [6]. Zubair et al. presented an accurate and realtime PE-Miner framework that automatically extracts distinguishing features from PE to detect malware. They completed a single-pass scan of all executables in the dataset and achieves more than 99% detection rate with less than 0.5% false alarm rate [3]. However, they scan the whole content of the PE files and cost almost one hour.

In this paper, I introduce a novel and faster approach to distinguish between malware and legitimate .exe files by simply looking at properties of the PE headers. Based on my observation, the features in the header fields show a lot of information about the file. For example, the NumberOfSections in the file header shows the number of sections in the PE file; the SizeOfInitializedData in the optional header shows the size of the initialized data section of the file; if the file embeds an icon, it will show in the .rsrc field in the section header. In order to find the most popular characteristics in malware, I extract those features and the embedded icons, then compare them to find the most significant difference between malicious files and benign files. To setup the experiment, I collect 5598 malware samples and 1237 benign files by a web-spider from two major download websites.

In summary, this paper makes the following contributions: (1). I have developed a PE-Header-Parser to extract the features from the PE headers and find the most five popular characteristics from malware. (2). I have developed a Icon-Extractor to extract the embedded icons from the PE files and find the most three popular icons and eight misleading icons from malware. (3). My experiments demonstrate that the approach detects 99.5% malware with only 0.16% false positive in only 20 minutes.

This paper is structured as follows: The next section gives an overview of PE file format. Section III presents details on the design and implementation of PE-Header-Based detection. Section IV presents the experimental results. Section V discusses the limitation and Section VI concludes the paper.

II. Overview of PE File Format

The overview of PE format is showing as the following TABLE I.

MS-DOS 2.0 Compatible EXE Header
Unused
OEM Identifier OEM Information Offset to PE Header
MS-DOS 2.0 Stub Program and Relocation Table
Unused
PE Header (Aligned on 8-byte boundary)
Section Headers
Import Pages Import information Export Information Base relocations Resourcce information

TABLE I
Typical Portable EXE File Layout

The PE file header consists of a MS-DOS stub, the PE signature, the file header, and an optional header. The MS-DOS stub is a valid application that runs under MS-MOS. After the MS-DOS stub, at the file offset specified at offset 0x3c, is a 4-byte signature that identifies the file as a PE format image file. At the beginning of an object file, or immediately after the signature of an image file, is a standard COFF file header. Every image file has an optional header that provides information to the loader. This header is optional in the sense that some files do not have it. For image files, this header is required. The optional header itself has three major parts: Standard fields, Windows-specific fields, and Data directories. Each row of the section table is, ineffect, a section header. This section table immediately follows the optional header [6].

Based on my observation, I extract features from the fields of file header, optional header and section header in my dataset.

III. PE-Header-Based Detection Approach

My PE-Header-Based detection approach consists of three main methodology: (1) develop a Web-Spider to collect a dataset of benign files, (2) develop a PE-Header-Parser to extract the features of optional header and section header fields, (3) develop a Icon-Extractor to extract the icon from the dataset of both malware and benign files. The overview of the approach is shown in Figure 1.

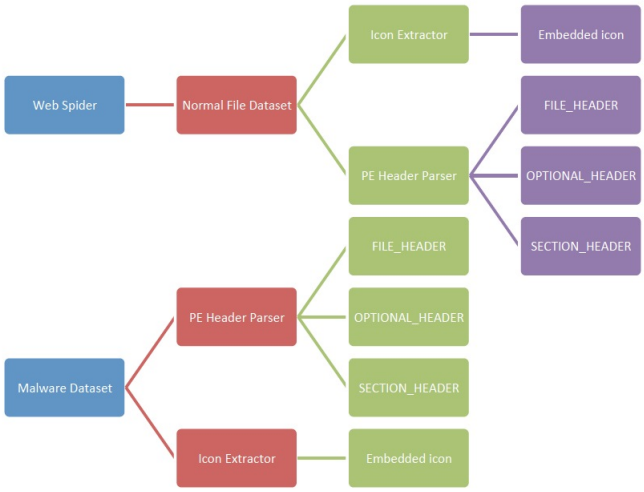

Fig. 1. Overview of the Web-Spider

A. Web-Spider

The Web-Spider is written in Python and using another two third-party python library called BeautifulSoup and Appscript. Firstly, the Web-Spider is getting the URL from the two websides: download.com [4] and softpedia.com [5]. Then the Web-Spider opens the page of the URL and parse the page using BeatuifulSoup library to find the tag contains the "href" and the keyword of the downloadable link. Lastly, the Web-Spider pass the url to the web browser and open the downloadable link using Appscript library to download the files.The overview of the Web-Spider is shown in Figure 2.

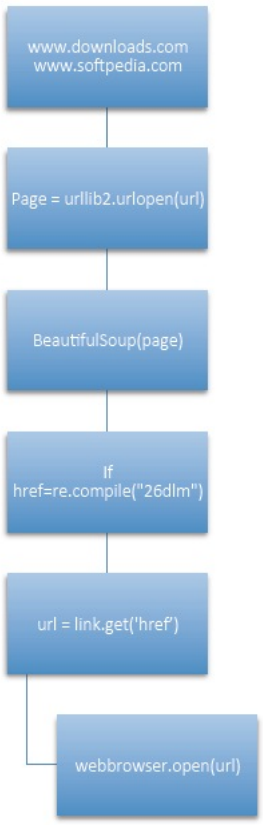

Fig. 2. Overview of the Web-Spider

B. PE-Header-Parser

The PE-Header-Parser is also written in Python and using the pefile library. As we described below, the PE headers contains important information about a file such as the number of sections, the size of the data, etc. The section header contains important information about the sections such as the section name, size, offset, etc. Firstly, the Parser import the pefile and parse all the files from the malware dataset and benign dataset respectively. Then the Parser extracts the features from file header, optional header and section header, compare the differences between malware and benign files.

1) File Header: The file header consists of the following features: Machine, NumberOfSections, TimeDateStamp, PointerToSymbolTable, NumberOfSymbols, SizeOfOptionalHeader, and Characteristics. Unfortunately, there is no big differences between malware and legitimate files in each features.

2) Optional Header: The Optional Header consists of standard fields (8 features), windows specific fields (21 features), and data directories. It is interesting to note that the SizeOfInitializedData in some of the malicious executables are equal

to zero, whereas none of them in the benign files are equal to zero. There are three other features, DLLCharacteristics, MajorImageVersion, and CheckSum are equal to zero in more than 90% malware samples. However, most benign executables contain significant higher values in such fields.

3) Section Header: The section header provides important characteristics of a section such as its name, address, size, etc. In this study, I found that many of the malware contain unknown names such as .6dnn4fh4, .Bga1m3ar, IOu15g4I, etc. The benign files offen contain the basic section name such as .text, .data, .rsrc, etc, or meaningful name like .shared, .page, .init, winzip, etc.

The Figure 3 and Figure 4 summarizes the top five features that contain than significant differences between malware and legitimate executables.

Index	Key Features	Malware (5598)	Normal (1237)	Difference
1	Size Of Initialized Data == 0	1626 (29%)	0 (0%)	29%
2	Unknown Section Name	2709 (48.4%)	16 (1.3%)	47.1%
3	DLL Characteristics == 0	5335 (95.3%)	401 (32.4%)	62.9%
4	Major Image Version == 0	5305 (94.8%)	486 (39.3%)	55.5%
5	Checksum == 0	5084 (90.8%)	474 (38.3%)	52.5%

Fig. 3. Top 5 Features

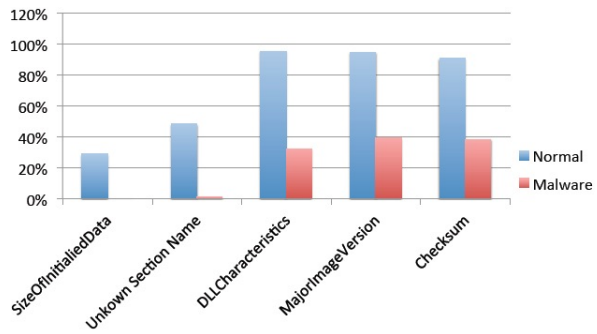

Fig. 4. Top 5 Features

C. Icon-Extractor

The Icon-Extractor is written in Python, pywin32 library, and PyQt4 library. It extracts the icons from the malware and legitimate executables respectively using ”win32gui” and save as .ico to external directories. The Icon-Extractor has found three most prevalent icons and eight misleading icons from malware.

IV. Evaluation

In order to evaluate the approach, I have collected 1237 benign PE files from the two website: downloads.com and softpedia.com. The 5598 malware samples are provided by Dr. Perdisci. All files are less than 10MB. I have done the experiments on an Intel Core i7 2.2GHz processor with 4GB Memory. The Miscrosoft Windows 7 professional is installed on this machine.

A. Header Parser Result

I have identified the features' set that consists of a number of statically computable features based on the structural information of the PE files. It is possible that some of the features might not convey useful information in a particular scenario. Therefore, it makes sense to combine them with other similar features to increase the accuracy.

Figure 5 is the evaluation result showing that combining each of the features within the index shown in TABLE II could result different detection rate and false positive rate. The following Algorithm shows the last combination of 1, 2, 3, 4, 5 fetures. If we combine all the five fetures, we can get the highest detection rate and the lowest false positive rate. The total time for the detection is less than 20 minutes.

Algorithm: Combination of 1, 2, 3, 4, 5 features for Malware Detection

```
Read the file
if SizeOfInitializedData == 0 then
    return malware
else if UnknowSectionName then
    return malware
else if (DLLCharacteristics == 0
      and MajorImageVersion == 0
      and CheckSum == 0) then
    return malware
else
    return benign
end if
```

Features combined	Malware detected	True positive	Normal detected	False positive
1,2	2811	50.2%	16	1.3%
1,2,3	5428	97.0%	10	0.8%
1,2,4	5402	96.7%	9	0.7%
1,2,5	5374	96.0%	3	0.2%
1,2,3,4	5506	98.4%	5	0.4%
1,2,3,5	5537	98.9%	3	0.2%
1,2,4,5	5482	97.9%	2	0.16%
1,2,3,4,5	5568	99.5%	2	0.16%

Fig. 5. PE-Header-Based Detection Results

B. Icon Extraction Result

The Icon-Extractor has extracted 813 out of 1237 icons from legitimate executables and 2165 out of 5598 icons from malware. The most prevalent icons which are seldom seen in legitimate PE files are shown in Figure 6 with the number of icons embedded in the malware dataset. Figure 7 shows eight misleading icons embedded in the malware

Fig. 6. Top 3 Prevalent Icons in Malware

Fig. 7. Misleading Icons

V. Limitations

The current version of this approach cannot detect all the malware from the dataset. I believe that there are some other features can be used as the key features to detect malware, such as extracting the Flags in the Characteristics fields of file header and optional header. Those limitations are left for future work.

VI. Conclusion

It's possible to identify the malware by looking at some key features from headers such as Checksum, Section Name, Initialized Data Size, DLL Characteristics and Major image Version. Looking at the PE header is much faster than scanning the whole information in the PE files. We can also identify the malware by extracting the embedded icons such as the prevalent icons or misleading icons as described below.

Acknowledgment

I would like to thank Dr. Perdisci to assign this project to me, and give me a lot of advice that greatly helped me to improve my research experience in Network and Security. I also want to thank the classmates for their helpful discussion in my presentation.

References

[1] S. Alvarez and T. Zoller. The death of AV defense in depth? - revisiting anti-virus software. http://cansecwest.com/csw08/csw08-alvarez.pdf, 2008.

[2] S. Jana and V. Shmatikov. Abusing File processing in Malware Detectors for Fun and Profit. In *Proceedings of the 33nd IEEE Symposium on Security & Privacy*, San Francisco, CA, U.S.A, May, 2012.

[3] Shafiq, M. Zubair and Tabish, S. Momina and Mirza, Fauzan and Farooq, Muddassar. PE-Miner: Mining Structural Information to Detect Malicious Executables in Realtime. In *Proceedings of the 12th International Symposium on Recent Advances in Intrusion Detection*, RAID '09, pages 121-141, Saint-Malo, France, 2009.

[4] Downloads. http://www.downloads.com/.

[5] Softpedia. http://www.softpedia.com/.

[6] Microsoft Portable Executable and Common Object File Format Specification. http://msdn.microsoft.com/library/windows/hardware/gg463125/.