PE-Probe: Leveraging Packer Detection and Structural Information to Detect Malicious Portable Executables

M. Zubair Shafiq, S. Momina Tabish, Muddassar Farooq

Next Generation Intelligent Networks Research Center (nexGIN RC)
National University of Computer & Emerging Sciences (NUCES-FAST)
Islamabad, 44000, Pakistan.
{zubair.shafiq,momina.tabish,muddassar.farooq}@nexginrc.org

June 2009

Abstract

The number of executable malware and the sophistication of their destructive ability has exponentially increased in past couple of years. Malware writers use sophisticated code obfuscation and encryption (a.k.a. packing) techniques to circumvent signatures – derived from the code of the malware for detection – stored in the signatures' database of commercial off-the-shelf anti-virus software. In fact, it is claimed that more than half of new malware are created by simply re-packing the existing malware. Malware packing can undoubtedly be considered as the most challenging problem faced by anti-virus vendors nowadays.

In this paper we present a novel scheme – PE-Probe – which has the ability to detect packed files and uses structural information of portable executables to detect zero-day (i.e. previously unseen) malicious executables. As a result, our proposed scheme is fully robust to code obfuscation and packing techniques. PE-Probe functions in two phases: (1) it classifies a given executable as packed or non-packed by employing well-studied heuristics, and (2) it invokes specialized structural models – separately developed for packed and non-packed executables – for malware detection on the basis of the outcome of the previous step. PE-Probe is real-time deployable as its scanning time is, on the average, less than quarter of a second per executable.

We have carefully designed experiments – keeping in view the stringent testing scenarios – to analyze the reliability and robustness of our scheme to packing and obfuscation techniques. We report our experiments on a recently obtained malware dataset from `OffensiveComputing.net`, which contains more than half a million malicious executables.

1 Introduction

Computer malware is becoming an increasingly significant threat to the computer systems and networks world-wide. In recent years, security experts have observed an explosive increase in the number and sophistication of new malware. According to a recent threat report by Symantec, in 2008 alone, $5,491$ new software vulnerabilities were reported, 1.6 million new malware signatures were created, 245 million new attacks were reported, and the financial losses caused by malware soared to more than 1 trillion dollars [11].

A majority of anti-virus (AV) vendors deploy signature based malware detection techniques that utilize a pre-defined signatures' set. The talk of detecting zero-day malware still eludes commercial AV software. A malware can cause significant damage in the "vulnerability window" which starts from the launch of new malware and finishes when its signature update is downloaded by users from the vendors' web site.

`Chernobyl`, `Z0mbie`, and `CodeRed` are classic examples of malware which exploited the vulnerability window to cause significant damage. The extent of penetration recently achieved by `Conficker` is another reminder to the AV industry that its products are susceptible to zero-day attacks. Moreover, the number of new malware signatures created in 2008 alone are more than half of malware signatures cumulatively created before this year. The manifold increase in the number of new malware signatures will definitely increase the complexity of scanning process. But owing to Moore's Law – the number of transistors on a chip double every 18 months – the effect of increasing processing overheads will be limited.

Another important issue to reckon with is the fact that malware writers often use sophisticated code-obfuscation techniques to evade signatures. In fact, simply repacking existing malware – in most cases – would evade signature based techniques. Malware authors launch new versions of a given malware by simply re-packing it using another packer. It is pertinent to note that 50% new malware are re-packed versions of existing known malware. Moreover, 92% malware use some kind of packing techniques. It is claimed that malware packing is the most challenging threat currently faced by AV vendors [12].

A number of non-signature based malware detection techniques have been proposed recently. These techniques analyze several features of malware, such as machine-level code, disassembled code, static calls from the disassembled code, and run-time API calls. The common bottlenecks of such techniques are their high false alarm rates and run-time performance overheads.

To this end, we have proposed a novel non-signature based malware detection technique – called PE-Miner – which overcomes the above-mentioned limitations [9], [10]. PE-Miner extracts approximately two hundred features using structural information of portable executable (PE) files. These features are extracted from headers of all major portions of PE files. It has been shown that decision tree (J48) is the quickest and most accurate classification scheme. However, we have also shown that the proposed structural features are not fully robust to executable packing. The detection accuracy of PE-Miner degrades – in a worst case scenario – by approximately 10%, when it is trained using only packed executables and tested using only non-packed executables. This highlights the extent of bias shown by structural features for packed/non-packed executables.

In this paper, we propose an improved version of PE-Miner – called PE-Probe – which removes the above-mentioned bias of structural features. PE-Probe first classifies a given executable as packed or non-packed based on well-studied heuristics. Based on previous outcome, the test executable is compared with either of the two specialized structural models – separate models are developed for packed and non-packed executables – for malware detection. We evaluate the accuracy of our proposed technique using an extensive malware collection obtained from `OffensiveComputing.net`. We also show the robustness of our scheme to executable packing.

2 Architecture of PE-Probe

In this section, we present the detailed architecture of our PE-Probe malware detection system. Figure 1 shows the detailed architecture of PE-Probe. To remove the bias of structural features towards packed/non-packed executables, we have utilized two modules:

1. A non-signature based packer detector (M-1) to classify packed/non-packed executables and then process them separately.

2. We build separate customized learning models for packed/non-packed executables. To this end, we identify a subset of features (M-2(np)) for non-packed files which have the most information. Similarly, for packed files, we select features (M-2(p)) which are least perturbed by packing.

In the following subsections, we provide the design details of each module.

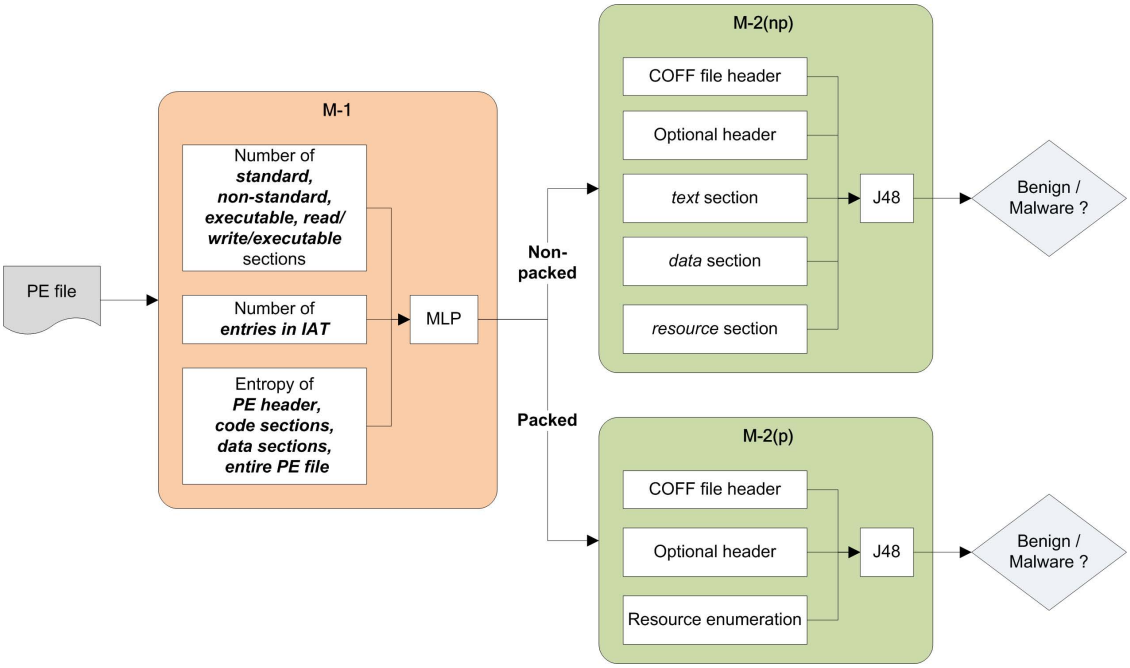

Figure 1: Architecture of PE-Probe

2.1 M-1

The task of M-1 module is to classify a given executable as packed/non-packed. A number of signature based packer detectors are available such as PEiD [5] and Protection ID [6]. However, signature based packer detectors are known to have notoriously high rate of false negatives, i.e. packed files detected as non-packed. In comparison, universal un-packers detect and extract the presence of hidden code using sand-box environments [2], [8]. As a result, they not only have significantly large performance overheads but are also prone to errors such as halt, crash and evasion. Therefore, in our PE-Probe, we utilize an efficient and accurate heuristics based packer detector (M-1) [7]. These heuristics are fed to a multi-layer perceptron (MLP) for eventual classification. The heuristics deployed by M-1 can be categorized into three categories, which we will discuss in the following subsections.

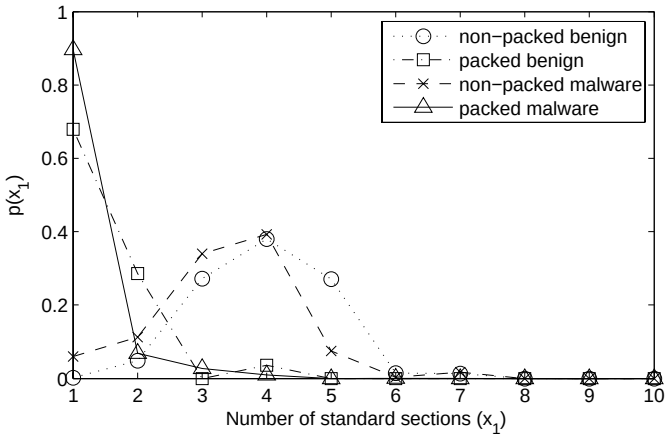

Figure 2: Distribution of the number of standard sections.

2.1.1 The name, number and type of sections

The name of sections used in benign executables are fairly standard as they are mostly compiled using Microsoft Visual C/C++/Basic compilers. On the other hand, packers mostly change the standard names of sections. Figure 2 shows this trend for packed/non-packed and benign/malware PE files. The packed executables generally have lesser number of executable sections compared with non-packed executables. Packers mostly attach a stub section to the packed executables in order to decompress or decrypt the remaining code. Therefore, such sections are enabled for read/write/executable simultaneously. Such type of sections are not frequently seen in benign PE files.

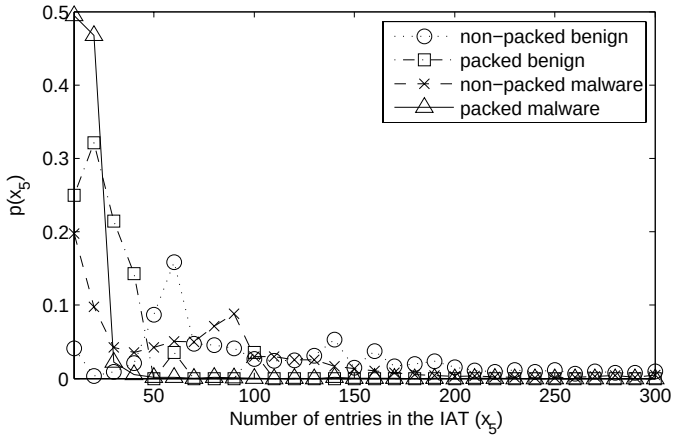

Figure 3: Distribution of the number of entries in import address table.

2.1.2 The number of entries in the import address table

The import address table (IAT) contains the addresses of external functions. Most benign executables have a large number of entries because they import different functions from Windows API. In comparison, packers hide these functionalities into compressed data and only an unpacker stub section is made visible. Consequently, the number of entries in import table are significantly small. The unpacker stub decompresses at run-time and adds the address entries in the import address table. Figure 3 highlights these patterns for packed/non-packed and benign/malware files.

2.1.3 The entropies of various portions of an executable

Packers tend to compress or encrypt various portions of an executable file. It is a well-known fact that "randomness" of a compressed portion is higher than well-structured headers and code/data sections [3]. Consequently, entropy[1] of the compressed portions is higher compared with that of non-compressed portions. In our study, we calculate entropies of PE header, code sections, data sections and entire PE file. Figure 4 shows the distribution of entropy of PE header.

2.2 M-2

The second module (M-2) consists of customized models for packed and non-packed executables compared with a single model of PE-Miner. An interested reader may find the details of structural features used by

[1]Entropy quantifies the amount of information/randomness contained in the given data.

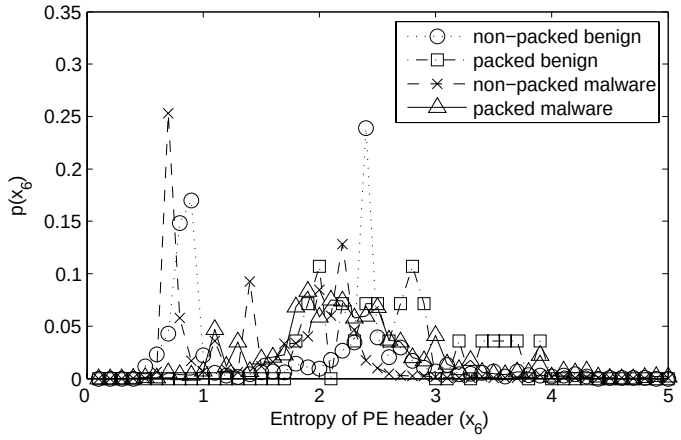

Figure 4: Distribution of the entropy of PE header.

PE-Miner in [9], [10]. For brevity we skip them and only explain the details of sub-modules added to the original PE-Miner framework.

2.2.1 M-2(np)

M-2(np) contains the customized model for non-packed executables only. For this module, we only select the features with the highest information. To quantify the information contained in features, we use a well-known information-theoretic measure called information gain. We calculate the information gain values of all features for non-packed executables. We subsequently select only the group of features, belonging to a particular portion of an executable, with the highest values of information gain. For M-2(np), we only use the top quartile of features. Figure 5 shows the distribution of information gain values of all features. In this figure, the top quartile is represented by '+' marks on the right hand side of the solid line. The top quartile includes structural features extracted from COFF file header, optional header, and text, data, resource sections (see Figure 1). We also plot the distribution of 'major linker version' feature in Figure 6.

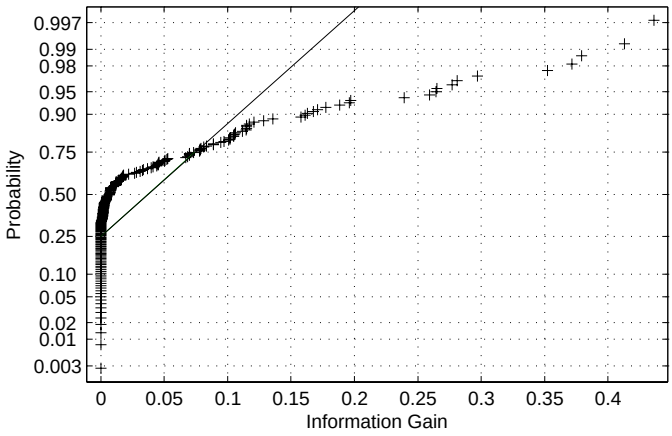

Figure 5: Normal distribution plot for information of structural features for non-packed PE files.

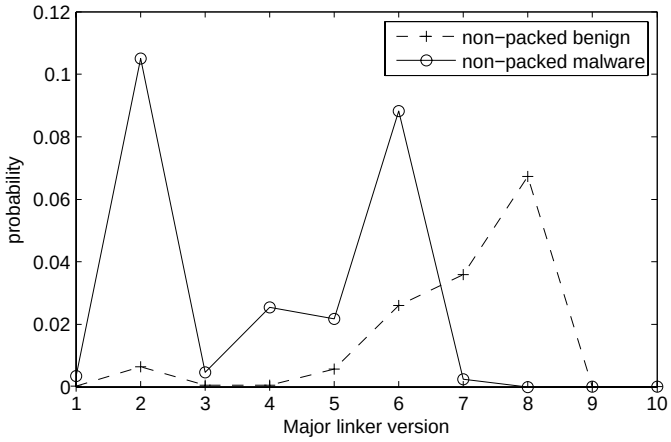

Figure 6: Distribution plot of 'major linker version' feature.

2.2.2 M-2(p)

The purpose of this sub-module is to distinguish between packed benign and packed malware executables. Intuitively, we select the subset of features which are: (1) robust to packing of executable, and (2) show high discrimination ability between packed benign and packed malware files.

To achieve the first objective, we take a sample of non-packed PE files and compress them using several well-known packers such as ASPack, PeTite, PECompact, and UPX. To identify features which are robust to packing, we compare the distributions of every feature for packed and non-packed executables. We use a well-known Kullback-Leibler divergence (D_{KL}) measure to quantify the difference between two distributions. KL divergence between distributions of a feature for packed/non-packed executables is defined as:

$$D_{KL}(P||Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)} \tag{1}$$

Figure 7 shows the distribution of KL divergence across all structural features. Intuitively speaking, we should select the features having least divergence values.

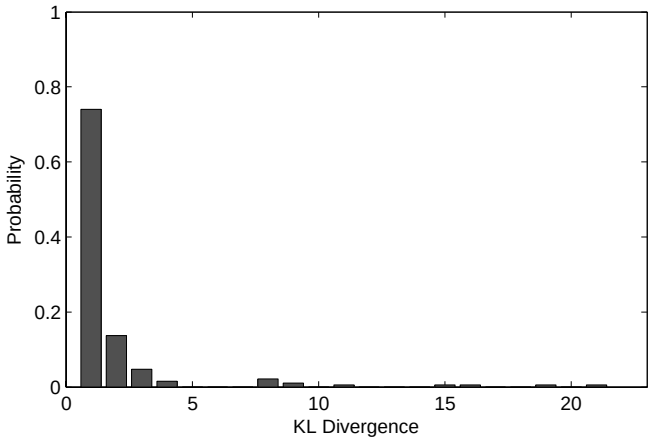

Figure 7: Distribution plot of KL divergence of all features across packed and non-packed PE files.

After quantifying the relative robustness of features to packing, we now carry out an analysis to identify the most distinguishing features. We use information gain to quantify the distinguishing power of every feature between packed benign and packed malicious executables. Figure 8 shows the normal distribution plot of information gain of all features.

We also enlist the portions of a given PE file from which the features are selected for M-2(np) and M-2(p) in Figure 1. Once the features are extracted in M-2(np) and M-2(p), they are fed to a decision tree (J48 variant) for classification. The decision trees have the capability to develop accurate training models, even for high-dimensional input feature space, and have relatively lower testing complexity.

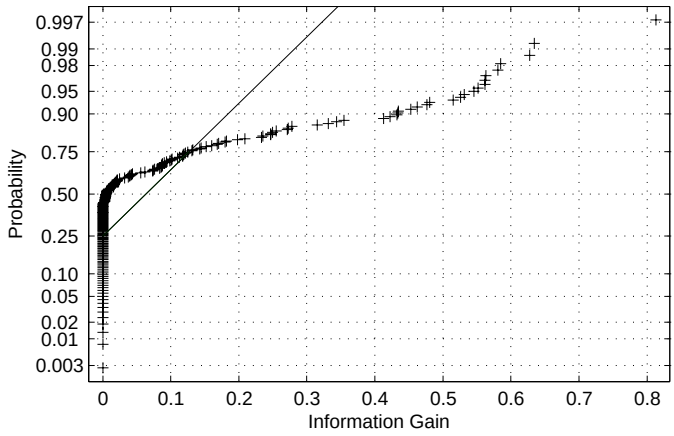

Figure 8: Normal distribution plot for information of structural features for packed PE files.

3 Results & Discussions

In this section, we report the results of our PE-Probe on a large malware collection that consists of malicious and benign executables. We have obtained about half a million malicious executables from `Offensive Computing.net` [4]. We have also obtained several thousand benign executables from freshly installed Windows machines and online resources such as `sourceforge` and CNET's `download.com`.

We tabulate the basic characteristics of the executables used in our study in Table 1. It is evident that both malware and benign executables considerably vary in their sizes – ranging from a few bytes to more than one hundred mega-bytes. Note that the average sizes of some malware categories exceed those of benign files. We have also analyzed – on the basis of the outcome of M-1 – the percentage of packed executables for all malware categories and benign files. The percentage of packed samples exceeds 60% for email flooders. It is interesting to note that benign executable dataset has less than 2% packed samples. Most of the packed benign executables are compressed setup installers. These samples are neither adequate nor are true representatives of the real-world packed benign files. Therefore, we increase the quantity and diversity of packed benign files by manually packing benign files using well-known packers such as ASPack, PeTite, PECompact, and UPX.

Table 2 shows the detection accuracy results of PE-Probe for all malware categories. The reported results are obtained using 10-fold cross validation procedure – the dataset is divided into 10 equal folds and nine folds are used for training and the left over fold is used for testing. This procedure is carried out for each of the ten folds and the averaged results are reported. The decisions made by PE-Probe may fall into one of the following categories for every test file [1]:

1. Detects a malicious executable, true positive (TP).

Table 1: Statistics of the executable dataset used in this study.

Malware Category	Minimum Size (B)	Maximum Size (MB)	Average Size (KB)	Packed (%)
Benign	817	107.1	371.6	1.9
Backdoor	43	25.7	224.9	56.3
Constructor	62	7.23	265.4	50.0
DoS	610	1.33	131.0	50.6
Email Flooder	894	15.0	294.5	48.3
Email Worm	28	45.3	50.3	62.9
Exploit	57	4.6	133.6	43.6
Flooder	248	1.6	169.8	49.9
Hoax	25	8.7	87.2	47.2
AdWare	68	24.2	548.7	46.9
FraudTool	36	8.1	170.4	56.9
Porn	7008	0.2	103.2	59.1
Rootkit	75	2.7	85.1	57.3
Virtool	31	2.9	84.0	51.0
Worm	28	10.4	394.8	53.2
Virus	10	12.5	54.5	35.8
Trojan	8	46.8	252.6	39.2

2. Detects a benign executable, false positive (FP).

3. Does not detect a malicious executable, false negative (FN).

4. Does not detect a benign executable, true negative (TN).

The detection accuracy of PE-Probe is reported using two separate parameters: (1) detection rate (DR), and (2) false alarm rate (FAR). Mathematically,

$$DR = \frac{TP}{TP + FN} \tag{2}$$

$$FAR = \frac{FP}{FP + TN} \tag{3}$$

Table 2 shows the detection accuracy results separately for packed and non-packed executables. On the average, the packed executables are classified with a DR of 0.996 and an FAR of 0.003. In comparison, non-packed executables are classified with a DR of 0.994 and an FAR of 0.008. The reason for having relatively high detection rate for packed executables might be because of the artifacts introduced by packers, which are commonly used by malware writers. It is interesting to know that trojans are detected with 100% accuracy. In comparison, it is relatively difficult for PE-Probe to detect AdWare and Worm executables.

It is important to emphasize that – in addition to high DR and low FAR of malware detection for both packed/non-packed executables – our framework provides valuable forensic information that can be extracted from decision tree model. Figure 9 shows a portion of the developed model for discrimination between constructor and benign executables. The features with relatively higher discrimination power are at a higher level in the tree. An interesting insight in Figure 9 is that a significant proportion of benign

Table 2: Detection accuracy results of PE-Probe for all malware categories.

Malware Category	Detection Accuracy			
-	Packed		Non-Packed	
-	Detection Rate	False Alarm Rate	Detection Rate	False Alarm Rate
Backdoor	0.999	0.001	0.998	0.002
Constructor	0.997	0.003	0.995	0.005
DoS	0.997	0.003	0.999	0.001
Email Flooder	0.995	0.005	0.996	0.004
Email Worm	0.995	0.005	0.999	0.001
Exploit	0.996	0.004	0.996	0.04
Flooder	0.997	0.003	1.000	0.000
Hoax	0.998	0.002	0.990	0.010
AdWare	0.989	0.011	0.978	0.022
FraudTool	0.998	0.002	0.981	0.019
Porn	1.000	0.000	1.000	0.000
Rootkit	0.998	0.002	1.000	0.000
Virtool	0.996	0.004	1.000	0.000
Worm	0.988	0.012	0.980	0.020
Virus	0.998	0.002	0.996	0.004
Trojan	1.000	0.000	1.000	0.000
Average	0.996	0.003	0.994	0.008

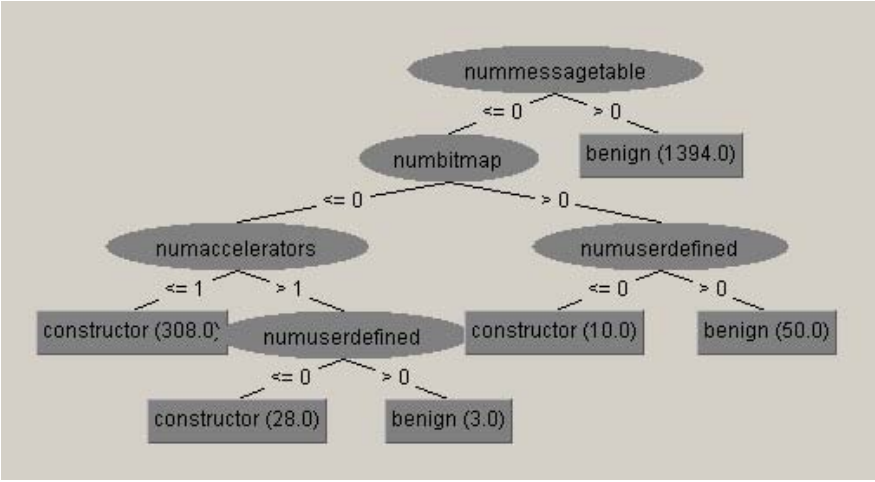

Figure 9: Visualization of a portion of the decision tree developed by J48 algorithm.

files have more than one entry in the message table. Moreover, the malicious executables have fewer entries in resource section – such as bitmaps, accelerators, and user-defined resources – than those in benign executables.

4 Conclusion

In this paper, we have proposed an accurate malicious executable detection technique called PE-Probe. As compared to its earlier counterpart, PE-Miner, PE-Probe deploys a lightweight packer detection module which is based on well-studied heuristics. In the next step, the customized models for packed and non-packed executables are used depending upon the results of the packer detection module. PE-Probe removes any bias in the structural features because of different packing techniques. Consequently, the detection ability provided by PE-Probe is definitely more robust. The results reported in this study are carried out using an extensive malware collection obtained from `OffensiveComputing.net`, which consists of approximately half a million malware samples.

References

[1] T. Fawcett, "ROC Graphs: Notes and Practical Considerations for Researchers", TR HPL-2003-4, HP Labs, USA, 2004.

[2] M.G. Kang, P. Poosankam, H. Yin, "Renovo: A hidden code extractor for packed executables", 5th ACM Workshop on Recurring Malcode (WORM), ACM Press, 2007.

[3] R. Lyda, J. Hamrock, "Using entropy analysis to find encrypted and packed malware", IEEE Security and Privacy (S&P), 5(2), pp. 40-45, 2007.

[4] Offensive Computing, Community Malicious Code Research and Analysis, available at `http://www.offensivecomputing.net/`.

[5] PEiD, available at `http://www.peid.info/`.

[6] Protection ID - the ultimate Protection Scanner, available at `http://pid.gamecopyworld.com/`.

[7] R. Perdisci, A. Lanzi, W. Lee, "Classification of Packed Executables for Accurate Computer Virus Detection", Pattern Recognition Letters, pp. 1941-1946, 29(14), 2008.

[8] P. Royal, M. Halpin, D. Dagon, R. Edmonds, W. Lee, "Polyunpack: Automating the hidden-code extraction of unpack-executing malware", 22nd Annual Computer Security Applications Conference (ACSAC), 2006.

[9] M.Z. Shafiq, S.M. Tabish, F. Mirza, M. Farooq, "PE-Miner: Mining Structural Information to Detect Malicious Executables in Realtime", International Symposium On Recent Advances In Intrusion Detection (RAID), Lecture Notes in Computer Science, Springer, France, September, 2009.

[10] M.Z. Shafiq, S.M. Tabish, F. Mirza, M. Farooq, "A Framework for Efficient Mining of Structural Information to Detect Zero-Day Malicious Portable Executables", TR-nexGINRC-2009-21, Next Generation Intelligent Networks Research Center, Pakistan, 2009.

[11] Symantec Internet Security Threat Report, Volume XIV, 2009.

[12] G. Taha, "Counterattacking the packers", McAfee Avert Labs, Aylesbury, UK.