Detection of Packed Malware

Dhruwajita Devi and Sukumar Nandi
Department of Computer Science and Engineering
Indian Institute of Technology Guwahati
Guwahati - 781039, Assam, India
{dhruwajita.devi, sukumar}@iitg.ernet.in

ABSTRACT
Packing is the most popular obfuscation technique used by malware writers' community in present scenario. The traditional signature-based anti-virus software had played a major role in malware detection, until the dawn of the trend of packed malware. Hence to evade detection of the malwares, a malicious writer relies on packers' softwares; which transforms the binary appearance of the programs without affecting its execution semantics. Therefore the biggest challenge today for malware detection techniques is to figure out whether a given binary is packed or not.

In this paper, we apply pattern recognition technique for detection of packed malware binaries. The objective of our approach is to take out the best set of features from Windows Portable executable files in order to pass it to our classification model. The classification model works in two phases, in the first phase it classifies the packed and non-packed executables. Once an executable is classified as packed, the second phase of classification concludes whether it is packed benign or packed malware executable. We worked with the UPX packer for this approach and have been able to achieve more than 99.9% accuracy in the first phase of classification. We achieved more than 95% accuracy in the second phase of classification as well.

Keywords
Portable Executable, Packer, Packed, Non-Packed

1. INTRODUCTION
In order to evade detection by the anti-virus (AV) software and the reverse engineers, the malware writers' community uses the obfuscation technique of packing malwares. Polymorphism, metamorphism, packing, and encryption, etc. are some of these techniques. Though several open source and commercial executables packers are available in the market, Executable Packing is the most popular of all [14]. Packing means an executable file packed inside another executable file [19]. A Packer or executable packing tool is basically software which generates a new program while an executable program is fed as an input. The new program embeds an encrypted version of the input executable and a decryption routine. At the time of execution, the outer executable unpacks the contents of the inner executable into memory and the inner most executable is the original executable.

Further packing is the technique of compression and decompression. It combines both the compression and/or decompression code into a single executable file [18]. The decompression code is needed because at the time of runtime the packed executable must have to be unpacked or decompressed.

There exist sophisticated packers which use more advanced technique to gain victory over reverse-engineering, bypass and disable unpackers. One of the techniques is Multi-layer packing. It uses a combination of potentially different packers, and then generates a large number of packed binaries from the same input binary [2]. Multi-layer packing itself could be used as an indication of malware. Another technique is Anti-unpacking. It falls into two major categories: passive and active. These techniques are designed to make it difficult to uncover the logic of an unpacker. Passive anti-unpacking techniques are intended to make disassembly harder, which in turns makes it difficult to identify and reverse the unpacking algorithm. Active techniques are intended to protect the running binary against having the fully unpacked image intercepted and extracted. It can be further classified into three subcategories: Anti-dumping, anti-debugging, and anti-emulating. Enigma and Themida are some of the packers which promote their use of all of these techniques.

In this paper, pattern recognition techniques are applied for detection of packed malware binaries. The proposed framework works in two phases, in the first phase it classifies the packed and non-packed executables. Once an executable is classified as packed, the second phase of classification finds packed benign or packed malware executable. The UPX packer is used in the proposed frame work and has been able to achieve more than 99.9% accuracy in the first phase of classification and 95% accuracy in the second phase of classification.

In this paper, section 2 includes the related work. Section 3 provides a brief description of UPX packer. The proposed model is described in section 4. Experiments are included in section 5 and in its subsection. Section 6 and 7 include the comparison part and the conclusion respectively.

2. RELATED WORK
In paper [3], the approach captures an intrinsic nature of hidden code execution where the original code should be present in memory. The code is executed at some point at run-time. Hence, this approach monitors the runtime execution of the code. In paper [2], the Symantec Researchers develop software called Justin, is the abbreviation for Just-In-Time AV scanning. This approach basically uses some heuristics e.g. dirty page execution, stack pointer check and command-line argument access etc. to identify the end of unpacking of an executable at the time of execution. As soon as it detects the unpacking it invokes AV scanning against the process image at that time. By this way it detects the ingoing malicious activity. In paper [4], OmniUnpack monitors execution of a program in real-time to detect packing of an executable. After the executable gets unpacked it gives the malicious payload directly to the detector software. PolyUnpack [5] uses the technique of comparing the memory image with the static code model. All the above mentioned approaches are basically dealing with the runtime analysis of malware.

In paper [11] the author uses reverse engineering technique to determine the functioning of NsPack packer. They analyze the implementation and the unpacking technique of the respective packer using one of the famous reverse engineering tool OllyDbg.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SecurIT'12, August 17-19, 2012, Kollam, Kerala, India
Copyright 2012 ACM 978-1-4503-1822-8/12/08... $15.00

Sophos Ex. 1018
Sophos Ltd. and Sophos Inc., v. Webroot Inc.
PGR2023-00031
Page 1 of 1