Toward Automatic Discovery of Malware Signature for Anti-Virus Cloud Computing

Wei Yan and Erik Wu

Advanced Threats Research
Trend Micro, Inc.
USA

Abstract. Security vendors are facing a serious problem of defeating the complexity of malwares. With the popularity and the variety of zero-day malware over the Internet, generating their signatures for detecting via anti-virus (AV) scan engines becomes an important reactive security function. However, AV security products consume much of the PC memory and resources due to their large signature files. AV cloud computing becomes a popular solution for this problem. In this paper, a novel Automatic Malware Signature Discovery System for AV cloud (AMSDS) is proposed to generate malware signatures from both static and dynamic aspects. Our experiments on millions-scale samples suggest that AMSDS outperforms most state-of-the-art automatic signature generation techniques of both industry and academia.

Keywords: anti-virus, network security, malware, cloud computing.

1 Introduction

Malwares are used to compromise computers and to steal the users private data by exploiting software vulnerabilities[1,2]. In cases which malwares are the zero-day threats, generating their signatures for detecting via anti-virus (AV) scan engine becomes an important reactive security function. However, modern malwares can easily bypass AV scanners by using code obfuscation, which can prevent malicious file contents from being detected. Current malware signature generation technique always involves in heavy manual work by studying emulation traces with hours or even days delay. Therefore, security researchers are facing great challenges in overcoming the complexity of malwares, and fighting against the malware backlog is nothing new.

To effectively handle the scale and magnitude of new malware variants, anti-virus functionality is moved into the cloud. In this paper, Automatic Malware Signature Discovery System (AMSDS), a novel and lightweight desktop agent for AV cloud is described. AMSDS keeps a good workload balance between the desktop and cloud services. It can automatically generate a lightweight signature database with the size hundreds times smaller than traditional signature ones. In the AV cloud model, users do not need to install a large virus signature file, but a lightweight set of "cloud signatures". The benefits include easy deployment,

J. Zhou (Ed.): Complex 2009, Part I, LNICST 4, pp. 724–728, 2009.

low costs of operation, and fast signature updating. Further, AMSDS signatures can be easily integrated into existing AV products.

We begin in Section 2 by presenting a brief introduction of virus executable file format. In Section 3, we expound how AMSDS generates cloud desktop patterns. We present experimental results in Section 4, and close in Section 5.

2 Virus Executable File Format

Executable files are special-formatted file objects that can be understood and executed by operating systems. Examples of modern executable formats include Portable Executable format (PE) for Windows, Executable and Linkable Format (ELF) for Linux and Mach Object (Mach-O) for Mac OS. This paper is focused on the PE format[3] as it is the most popular format for executables, libraries, and drivers in Windows.

A PE file comprises various sections and headers which describe the section data, import table, export table, resources, etc. A PE file starts with the DOS executable header, which is followed by the PE header. The PE header begins with the signature bits "PE". The PE header also includes some general file properties, such as the number of sections, machine type, and time stamp. Another type of header is called optional header, which contains an array of important information segments. The optional header is followed by the section table headers, summarizing each section's raw size, virtual size, section name, etc. Finally, at the end of the PE file is the section data, which contains the file's Original Entry Point (OEP). OEP refers to the execution entry point of a PE file, where the file execution begins. To search a PE file for malwares, a scanner typically scans the segments at certain offsets from OEP for the known signatures. PE tools facilitate the ease to view, analyze and edit WIN32 PE files.

Existing commercial security applications search the binary files for pre-defined signatures to identify known malwares. Unfortunately, this technique can be easily fooled by obfuscated viruses, which use software packers [2](programs that compress and encrypt executable files in disk and restore the original executable image, when loaded into memory), to protect the viruses' internal code and data structures from being detected by security software.

3 AMSDS

AMSDS aims to generate intelligent malware signatures for AV cloud desktop. Each client has a lightweight AMSDS signature file, whose size is hundreds times smaller than traditional signature databases. Only when a suspicious file cannot be detected by AMSDS patterns, clients will send a request to a cloud server, where exists the full traditional pattern database.

In this paper, our assumption is based on the fact that the samples of the same malware family must have some identical binary raw sequences. Among those sequences, there exists a set of identical binary strings. Our AMSDS signatures are generated from those strings. AMSDS firstly parses a suspicious PE sample

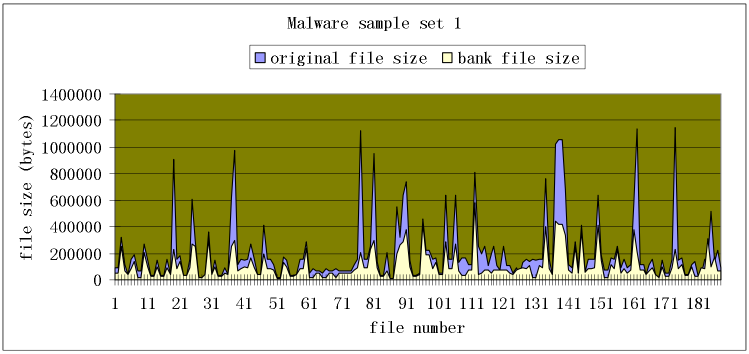

Fig. 1. AMSDS de-noise PE sample files

and list its internal structures, such as PE header, optional header, section table, import table, export table, and the resources. To speed up the signature generation, AMSDS will discard some raw data of this sample, and only reserves the segments where hackers may insert their malicious codes. This process is called ”de-noise”. After the denoising, AMSDS can make the testing environment safer by destroying malware’s PE formats. Therefore, the malicious sample can not be executed. Fig. 1 shows the de-noise performance for some malware samples. Owing to the de-noise stage, AMSDS can shrink the original file sizes almost by half.

AMSDS is able to generate malware signatures from both static and dynamic aspects. With the incoming of malware samples, the intelligent converter in AMSDS parses malwares static information, and uses machine learning technique to find multiple disjoint binary sequences. Based on those invariant substrings, the static compound signatures are generated for inline matching. AMSDS can be also used to automatically generate behavior signaures. Nowadays, the emulator or sandbox is used to capture malware dynamic traces. However, the overwhelming static and dynamic “noise” generated by using code obfuscation make it hard to analyze. Therefore, current malware signature generation techniques always involve in heavy manual work by studying emulation traces with hours or even days delay. A malware emulator and its tailored malware behavior ontology are used to collect the dynamic execution token traces. Those tokens are abstract representations by traversing the malware behavior ontology. Afterwards, the malware behavior signatures are generated from those token streams.

4 Simulation

We describe our experimental results on on millions-scale samples in this section. By using the disjoint invariant segments, AMSDS achieves low false positive rate; By using “back up” signatures, AMSDS can defeat code obfuscation efficiently. Our testing sets include more than 10 millions benign samples and almost 35k malware samples from 40 families. We choose small training set from 30 to 50 samples for each family. The average malware detection rate is around 80%. (For larger training set, the detection rate will be higher.) As shown in Fig. 4, the false positive rate for benign samples is 0.001%. In the aspect of speed, AMSDS is comparable with current AV products. For example, AMSDS takes 1m26s seconds to scan a folder, which includes 20,000 files.

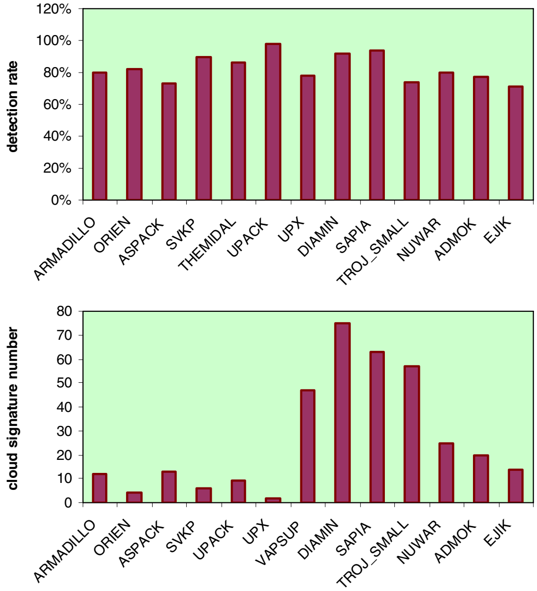

Fig. 2. Detection rate and pattern number

We also measured the malware detection rate for 30 families, which includes 10 Windows PE packer and 20 malware families. Fig.2 presents the detection rates and pattern number for each family. The total pattern file size is less than 10k. Our testing suggests that the average malware detection rate is around 80%. We also found an interesting result: the pattern number for each packer family is much less than that of malware family. The only reason we think is that few patterns are enough to capture the packers' unpacking semantics. Normally a packer's unpacking process involves four consecutive steps: decompression or decryption, anti-debugging checks, import table rebuilding, and jumping to OEP[2]. For packers, each above step always involves similar working flow. For example, in the import table building stage, a packer extracts the DLL names, followed by the trunk table addresses and APIs. For files with relocation tables, the packer stores the Relative Virtual Address (RVA) of the relocatable data blocks, which will be relocated when the relocation table needs rebuilding.

5 Conclusion

In this paper, we propose AMSDS, a ontology-based automatic signature generation system for zero-day malwares, which generates both static and dynamic tokens for AV cloud computing. Our approach is generic and flexible. Different otologies can be plugged in for various detecting purposes. The experiments show it outperforms state-of-the-art automatic signature generation techniques.

References

1. Grace, C.: Understanding intrusion detection systems. PC Network Advisor 122, 11–15 (2000)
2. Yan, W., Zhang, Z., Ansari, N.: Revealing packed malware. Journal of IEEE Security and Privacy 6(5), 65–69 (2008)
3. An In-Depth Look into the Win32 Portable Executable File Format, `http://msdn.microsoft.com/msdnmag/issues/02/02/PE/`