The Journal of Systems and Software 100 (2015) 91–102

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Profiling and classifying the behavior of malicious codes

Mamoun Alazab

Australian National University, Canberra, Australia

ARTICLE INFO

Article history:
Received 9 October 2013
Received in revised form 5 September 2014
Accepted 19 October 2014
Available online 25 October 2014

Keywords:
Cybercrime
Malware
Profiling

ABSTRACT

Malware is a major security threat confronting computer systems and networks and has increased in scale and impact from the early days of ICT. Traditional protection mechanisms are largely incapable of dealing with the diversity and volume of malware variants which is evident today. This paper examines the evolution of malware including the nature of its activity and variants, and the implication of this for computer security industry practices.

As a first step to address this challenge, I propose a framework to extract features statically and dynamically from malware that reflect the behavior of its code such as the Windows Application Programming Interface (API) calls. Similarity based mining and machine learning methods have been employed to profile and classify malware behaviors. This method is based on the sequences of API sequence calls and frequency of appearance.

Experimental analysis results using large datasets show that the proposed method is effective in identifying known malware variants, and also classifies malware with high accuracy and low false alarm rates. This encouraging result indicates that classification is a viable approach for similarity detection to help detect malware. This work advances the detection of zero-day malware and offers researchers another method for understanding impact.

1. Introduction

Cybercrime is growing and rapidly adapts to new opportunities in cyberspace. The Internet makes it easy for cyber criminals to operate virtually from abroad and remain anonymous online. It is therefore challenging to identify the source of the crime such as malware. Advanced technologies such as The Onion Router (the ability to apply multiple proxy routing that inhibits the ability to trace-back), Obfuscation (a term used to describe modification of a program to disguise its purpose, see details in Section 2), Dynamic Domain Name System (DNS), and Virtual Private Network (VPN) services are all capable of facilitating cybercrime. These methods enable cyber criminals to further their criminal operations by developing new forms and/or variants of malware to assist in the propagation of malware, which enables offenders to more effectively achieve online anonymity, to avoid network surveillance, and traffic analysis by regulators. When investigating cybercrime there are basic questions that arise. *Where* and *when* was the crime committed? *What* technologies were involved? *How* was the crime committed? *Who* committed the crime and *why* was the crime committed? Most research focuses on one aspect and neglects others and this maybe that practitioners have limited time to address problems and/or little or no experience with the wider problems caused by the Advanced Persistent Threat (APT) in cybercrime. A learning profiling approach could be helpful when time is limited and consequences are critical. Efforts by law enforcement and computer analysts working in government and non-government have helped to suppress cybercrime activities (Cowley, 2012; Anderson et al., 2013; Broadhurst et al., 2013). Most effective have been efforts that have focused on the analysis of the technical aspects of Internet crime, for example, deciphering malicious codes, identifying malware (Alazab and Venkatraman, 2013), shutting down botnets and other methods used by cybercriminals, and designing increasingly sophisticated strategies to protect computer systems. At the same time, a general lack of knowledge of the perpetrators themselves, their motivation, the development of new malware code, and their *modus operandi*, leads to many failures in the suppression of cybercrime and, therefore, cybercrime remains a fast growing global threat (Alazab et al., 2013).

The majority of malicious software is recycled and has not been written from scratch. In 2014, Symantec noted (Symantec Corporation, 2014) that the number of genuinely new malware families created has slowed as malware coders are working to perfect existing malware. Indeed, some malicious codes are being reused and modified. For example, in 2010 Symantec detected more than 286 million new malware variants (Symantec, 2011)

E-mail address: mamoun.alazab@anu.edu.au

http://dx.doi.org/10.1016/j.jss.2014.10.031
0164-1212/© 2014 Elsevier Inc. All rights reserved.

including 90,000 unique variants of the Zeus toolkit and, more recently, over 20 million variants were found in 2013 alone (Dell, 2014). The similarity between many botnet variants like Hlux, Waledac, Nuwar, Kelihos and Storm suggest that these bots were developed by the same botnet coders (details in Bureau, 2011). This year the Fox-IT InTEL technical report (Fox-IT InTELL, 2014) verified that the malware family Tilon was also linked with Zeus and the SpyEye malware family. It can be argued that malware codes have sufficiently distinct features from each other that could allow for profiling and further analysis. The lack of a comprehensive analysis of malware – not only for the purpose of detection – makes it hard to identify novel forms of attack, characterize the method of attack, and predict similar attacks in the future.

Behavior reflects personality and in the online world it reflects offenders. Profiling is an investigative tool that consists of analyzing the crime scene and likely behavior of the offender (in this case malware code) and using the information from this to determine the probable identity of the cybercriminal. Profiling criminal types may be derived from inductive methods that draw on the statistically derived characteristics of offenders from resolved cases, or by applying more speculative deductive methods based on the assumption that the *modus operandi* of the crime will be repeated in the same way by the same offender – in short the crime will have a signature (Douglas et al., 1986). Profiling in criminology is not a new method and the concept has been deployed with varying success in solving crimes, especially crimes that are violent and may show signs of offender pathology. Malware profiling is relatively new as a method for crime analysis. It is difficult to build the right profile for each and every malware code simply because malware codes can be programmed by someone, adjusted by another, and used by another. The motives and the circumstances should always be considered when a profile of malware code is assembled and so a holistic approach is required. The role of an approach similar to the notion of behavior profiling could be a promising approach, although there are many problems involved in successfully distinguishing legitimate behavior from malicious ones in the cybercrime context. To use a profiling in the Windows Operating System (OS) required close understanding of the hardware and software of the computer system.

The focus of this paper is to explore the properties and characteristics of the Portable Execution (PE) files. Similarity based detection of malware has been proposed using the sequences of Windows Application Programming Interface (API) calls and their frequency of appearance. This is an effective method to not only classify malware from benign files, but also to identify malware variants within existing malware families. The similarity analysis technique related to data mining provides a consistent method to help in tracking possible groups as it can profile malware behavior and actions that may match previous profile attacks. This approach can help to recognize future attacks, improve malware detection tools, and develop better user education about the dangers of malicious software.

2. Background

Literature surveys on malware detection and malware variants show that there is no single technique that could detect all types of malware as most rely on syntactic properties rather than the semantics of malicious code (Alzarooni, 2012). However, generally there are two techniques commonly used for malware detection: *signature-based* detection and *anomaly-based* detection (Dinaburg et al., 2008; Lawton, 2002; Birrer et al., 2009). All of the Anti-Virus (AV) engines use malware signatures as an essential method to identify malicious content. The malware signature is a byte sequence that uniquely identifies a specific malware. Typically, a malware detector uses the signature to identify the malware by the byte sequence, which acts like a fingerprint of the malware program. Most AV programs are search engines supplied with a database containing information about existing or known malware, and this can be used to search for code signatures such as byte sequences while scanning the system. A malware detector scans the system for characteristic byte sequences or signatures that match with the one in the database and if found blocks the malware access to the system. The signature matching process is called signature-based detection and most traditional AV engines use this method. It is a very efficient and effective method to detect known malware (Venkatraman, 2009). However, a major drawback is the inability of this method to detect new and unknown malicious code as new signature generation involves manual processing and requires detailed code analysis. To overcome signature-based methods, obfuscation techniques such as polymorphic malware is used which has an in-built polymorphic program/engine that can generate new variants each time it is executed and, as well, a novel signature. Therefore, signature based approaches would fail to detect such malware. On the other hand, anomaly-based detection uses the knowledge of normal behavior patterns to decide if a program code is malicious. This approach has the ability to detect even some zero day attacks. However, it is very difficult to accurately specify the system or program's behavior and thus these approaches usually result in more false positives than signature based methods.

2.1. Obfuscation techniques

Malware coders are continually developing new techniques for transforming binary code that cannot be detected by AV scanners, and their level of sophistication is continuing to grow. Malware coders are applying obfuscating techniques (O'Kane et al., 2011) such as *packing* (Guo et al., 2008), *polymorphism* (Stepan, 2005), *Oligomorphic*, and *metamorphism* (Qinghua and Reeves, 2007) in order to transform existing malware code with signatures that are either disguised or are changed from those that might be held in a signature directory without affecting the original functionality or purpose. By looking at real world malware threats such as Parite, Rimecud, Alcan, Rbot, CeeInject, Nachi Bagle, and many other malware, it found that coders are recycling existing malware with different signatures by using obfuscation techniques instead of creating entirely new codes. This has created a serious threat for computer security. We define the term obfuscation to mean the modification of the program code in a way that preserves its functionally with the aim to reduce vulnerability of any kind to static/dynamic analysis and to deter reverse engineering by making the code difficult to understand and less readable. Code Obfuscation is an evasion method used to morph the program code in such a way as to make it hard to read and difficult to understand without changing the main goal of the program. Obfuscation techniques are used by malware coders as well as legitimate software developers. They both use code obfuscation techniques for different reasons. For instance, cybercriminal use it to evade antivirus scanners since it modifies the program code to produce offspring copies, which have the same functionality but with different byte sequence or 'malware signature' that is not recognized by antivirus scanners. But legitimate software developers use it for various reasons such as reducing file sizes, protecting against piracy, etc.

Malware coders are continually developing new techniques for creating and applying obfuscation techniques $T(P)$ on a malware program (p) to produce an obfuscated program (p'), thereby making it very difficult to reserve engineer and decipher the signature successfully, even though the two programs, p and p' have the same functionality and exhibit the same affect. On the other hand, since p and p' have different byte sequences, AV engines and reverse engineers are applying de-obfuscation techniques $D(p')$ on