CCIT Report #921 March 2018

Split Malware: Avoiding Behavioral Analysis Detection

Ofir Shwartz
Electrical Engineering Dept., Technion, Israel
ofirshw@tx.technion.ac.il

Yitzhak Birk
Electrical Engineering Dept., Technion, Israel
birk@ee.technion.ac.il

Abstract

Computer malware is one of the greatest dangers to the modern society, allowing attackers to uncover restricted data and to control a wide range of critical infrastructure. Furthermore, computer malware evolve rapidly, forcing anti-malware vendors to put most of their efforts on developing techniques for detecting new and therefore previously unknown malware.
We present Split Malware, a method for splitting malware into small pieces. Each piece is not discovered by anti-malware tools, yet together they perform a malicious task.

1. Introduction

Computers and computer systems are involved in nearly every aspect of our lives. They control everything, from our most sensitive data to major infrastructure, therefore we rely on their stability and reliability.

Modern risks of computer systems include attacks that take advantage of existing software bugs (e.g., operating system vulnerabilities), or malicious executables that may be implanted in various ways. Many attacks involve a small piece of software that either acts on its own and spreads autonomously, or acts as a local agent that receives instructions from remote. These are commonly referred to as "viruses", "malware" (malicious-software), and "Trojan horses".

In attempt to protect against those, "anti-malware" (or "anti-virus") solutions are commonly used, aiming to *detect* the malicious files, and then to *protect* the system against those. Malware detection is based on two main methods:

- **Signature based detection**, wherein before running an executable file, a hash calculated from its content is compared against a list of known hashes.
- **Behavioral analysis detection,** wherein each running program is tracked against previously defined sequences of events.

While signature based detection excel in negligible false positive rate and by its ability to detect threats before these are allowed to run, it is only useful against known files that their signatures were added specifically into the signature database by the anti-malware company. On the other hand, behavioral analysis uses known attack methods (rather than specific files) for detecting unknown malware files.

Using various obfuscation techniques, from encryption [1] to polymorphism [1], malware continuously weaken the signature based detection mechanism. This is because similar malware functionality is embedded in many different versions of the executable, each results in a different hash value. Obviously, behavioral analysis may detect such malware much easily, assuming a known sequence of events for a group of similar malware.

In this paper we present Split Malware, a novel technique for malware to overcome behavioral analysis detection. Our main observation is that each event tracked by behavioral analysis is separately allowed (or else the operating system would have simply blocked it), and only specific sequences of those is considered a threat. Therefore, splitting malware into small pieces breaks the sequence into small sequences (up to a single event per piece), where the communication between those may be done covertly [4]. Tracking only those small sequences will raise an unreasonable false positive rate, while grouping permutations of **all** the currently running programs to detect unified sequences cannot scale with the number of running programs.

2. Split Malware

The sole purpose of Split Malware is to bypass malware detection by behavioral analysis techniques. Behavioral analysis tracks running programs for sequences of events, and compares those to ones commonly used by malware. Each unique event is permitted, but specific sequences of those are prohibited. Therefore by splitting a malware file into smaller pieces, where each performs some of the desired operations (events), the tracked sequence is broken into small pieces that should not attract special attention.

The exact method used to implant the split malware pieces into the victim computer and making them run is orthogonal to this work, so it is not discussed here.

Splitting a malware should take into account the type of events commonly tracked by anti-malware. A good example for such events are the operating system services (system calls) invoked by the executable. The naïve approach would therefore be to split a malware into a main executable that runs the main program, and invokes each system call (event) via a different piece (one per system call). As a result, each executable only raises a single event, and no sequence is actually recorded.
Having a malware split into pieced raises the need for communication between those, both for triggering the other executables to perform a necessary operation (including passing parameters), and for responding (done / return value). An obvious choice may simply be to send messages between these applications using the operating system services, yet this allows the anti-malware software to group the communicating pieces in a set that is monitored together. Alternatively, covert channels may be used [2,3], limiting the communication rate between the pieces. However, malware commonly operate in the background striving not to attract any attention, so communication rates of several hundreds of bytes per second, offered by many covert channels, should suffice.

3. Conclusions

Split Malware is a novel technique allowing unknown malware to bypass behavioral analysis detection techniques. By splitting existing malware into small pieces, each triggers a small portion of a prohibited sequence of events, the original malicious operation may be executed yet the malware remains undetected. Split Malware is thus a new challenge for anti-malware companies and researchers striving to detect unknown malware.

References

[1] M. Schiffman, “A Brief History of Malware Obfuscation: Part 1 of 2”, https://blogs.cisco.com/security/a_brief_history_of_malware_obfuscation_part_1_of_2, Feb. 2010.

[2] O. Shwartz and Y. Birk, "Sound Covert: A Fast and Silent Communication Channel through the Audio Buffer." Parallel, Distributed and Network-based Processing (PDP), 2017 25th Euromicro International Conference on. IEEE, 2017.

[3] Brouchier, J., Kean, T., Marsh, C. and Naccache, D., 2009. Temperature attacks. Security & Privacy, IEEE, 7(2), pp.79-82.

[4] Lampson, B.W., 1973. A note on the confinement problem. Communications of the ACM, 16(10), pp.613-615.