Implement directive rules on SIEM by detecting mal-behaviors

Ge Zhang, Evry AB

I. Introduction

AlienVAULT is a Security information and event management (SIEM) product that we are deploying nowadays to manage our networks. It aggregates security related logs from different sources, including servers, databases, network devices, etc. In addition, it can do a realtime monitoring for the network by correlating the received logs. For instance, if a server generates a number of logs with "login failes due to wrong password" in a small time interval, these logs will be correlated and will hit a directive rule and then will alert a possible brute force attack. Promising as it seems, however, AlienVAULT only provides a limited number of build-in directive rules so that those rules cannot satisfiy our needs to cover as many threats as possible. On the other hand, it must be a heavy workload to implement one directive for each specific threats. In this research, we aim to find a way to solve this problem.

II. Directive rules for mal-behaviors

Instead of to detect specific attacks or malwares, we would like to use directive rules to detect only mal-behaviors (suspicious behaviors), e.g., modify registry key to add an autostart application, cleanup local logs, etc. There are several reasons to do so:

- Easy to implement: It is impossible to implement directive rules to match all attacks. In addition, a long directive ruleset will reduce the efficiency of AlienVAULT and thus will cuase performance bottleneck. In contrast, mal-behaviors are repeatedly employed in different attacks and malwares. We can abstract the mal-behaviors which are easier to implement.
- Detect unknown attacks: It is hard to reuse attack-specific rules for new threats, e.g., new variant, zero-days, etc, because the rules are defined too restrict. Differently, the suspicious behavior rule can be defined more relax. As said, many malware and attacks usually share the same suspicious behavor and thus the detection should be still efficient.
- Defense in depth: In the network, we have also employed anti-malware and IDS, which work well to detect known threat based on signature. Attack-specific rules on log correlation will do the same type of detection. On the other hand, behavior detection on log correlation will complement to anti-malwarea and IDS to form a defense in depth.
- Forensic: When security incidents happen, the alarms trigered by directive rules based on behaviors will quickly help us to locate to the logs which are related.

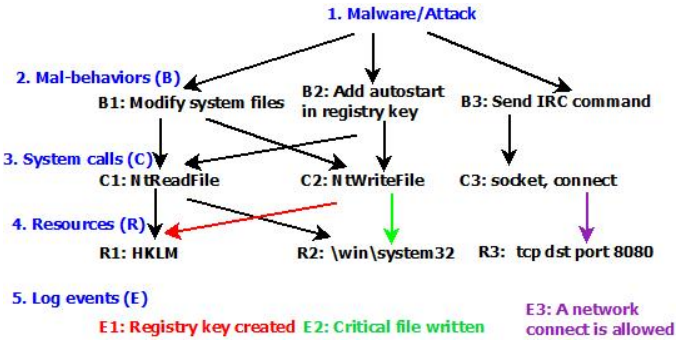

Fig. 1. A layered model from malware/attack (top) to log events (bottom)

A layered model is illustrated in Fig. 1. Malware/attack stands on the top of this model. It should contains one or more mal-behaviors, which listed on layer 2 (B_1 - B_3). To fulfil one mal-bahavior, the malware or attacker has to interactive with the operating system by using systems calls (layer 3, C_1 - C_3) against resources (layer 4, R_1 - R_3). The interaction bewteen system calls and reources will generate log events. The figure above uses different colors to indicate the log events generated correspondingly. For instance, using *NTWriteFile()* to modify registry keys to add an auto start will produce a log events: "Registry key created at HKLM\Software\...\Run".

Many previous research works focus on using the history of system calls to detect mal-behaviors [1], [2]. Several specific system calls in a sequence during a time interval indicates a possible of mal-behavior. That is, a traceback from layer 3 to layer 2 in Fig. 1. However, this method does not fit our central log environment because monitoring system calls for each asset will consume too much resources.

Differently, our research aims to traceback from layer 5 to layer 2. We do not generate logs for everything. We only focus on special resources those are usually manipulated by malware. For example, we select some system critical locations like "...\windows\system32" for file audit instead of the whole file system. We will use a learning method to collect the mal-behavior.

References

[1] Andrea Lanzi, Davide Balzarotti, Christopher Kruegel, Mihai Christodorescu, and Engin Kirda. Accessminer: using system-centric models for malware protection. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 399–412. ACM, 2010.

[2] Heng Yin, Dawn Song, Manuel Egele, Christopher Kruegel, and Engin Kirda. Panorama: capturing system-wide information flow for malware detection and analysis. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 116–127. ACM, 2007.