Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies

FIIT-XXXXXX-82385

Tomáš Belluš

Bait network based monitoring of malicious actors

Master thesis

Supervisor: Ing. Tibor Csóka, PhD.

May 2020

Slovak University of Technology in Bratislava

Faculty of Informatics and Information Technologies

FIIT-XXXXXX-82385

Tomáš Belluš

Bait network based monitoring of malicious actors

Master thesis

Study program: Information Security

Field of study: 9.2.4 Computer Engineering

Training workplace: Institute of Computer Engineering and Applied Informatics

Supervisor: Ing. Tibor Csóka, PhD.

May 2020

Contents

Chapter 1

Analysis

This chapter introduces malware analysis (see section 1.1) and differentiates the techniques (see subsection 1.1.1), outlines and describes mechanisms and environments utilized by cyber security professionals and companies (see subsection 1.1.2), describes the virtualization technology KVM (see subsection 1.2.1) and the orchestration mechanism Kubernetes (see subsection 1.2.2).

1.1 Malware analysis

Monitoring a malicious actor, sample or an entity in any environment is interchangeable from the analyst point of view. Therefore, knowing the malware analysis techniques is crucial for secure monitoring of malicious actors. Malware analysis may be static and dynamic with varying tools and mechanisms [3] [2].

1.1.1 Dynamic and static

Dynamic analysis is a process of actively monitoring, ideally in an isolated environment, the execution of a malware sample. Static analysis, as the name implies, inspects the function calls, readable strings, control and data flow of a malware sample (i.e. binary, program code).

Dynamic analysis may be unsafe and devastating, unless environment or system isolation is applied, but it's more precise and bypasses the reversing of self-modified code. Even though, dynamic analysis does not explore all execution paths, there are techniques resolving this drawback (i.e. virtual machine snapshots [2]). Furthermore, dynamic analysis in a isolated or virtualization environment is exposed to the risk of [isolation] detection [0].

Static analysis is safe [and inspects all execution paths of the binary], but may be difficult to interpret when malicious actors utilize the obfuscation, compression or encryption techniques. Advanced malware sample poses a challenge for a malware analyst due the usage of control-flow flattening [5] and other methods. It suggests that dynamic malware analysis bypasses this troublesome procedure and discloses the sample outcome or agenda.

Manual Egele, et al. in their article [3] highlight the problems of static malware analysis approaches. The authors introduce multiple state-of-the-art techniques and tools dedicated to dynamic malware analysis - Function Call Monitoring, Function Parameter Analysis focusing of the final values, Information Flow Tracking, Instruction Trace and Auto start Extensibility Points (monitor startup programs, cron jobs, etc.).

1.1.2 Mechanisms [and environments and tools]

These mechanisms must provide solution to dynamic malware analysis limitations in order to effectively capture the malicious actor's agenda. It should be a robust system implying the impression of a production environment.

1.1.2.1 Honeypot and honeynet

Honeypot is a bait service, system or a even whole network (honeynet) usually hosted on public server. Its main purpose is to be scanned, attacked or compromised by the malicious actor. Every honeypot provides the desired functionality of the target resource, to mimic the production environment, leaving the malicious actor unaware of the honeypot [4].

Based on the ENISA honeypot study [4], honeypots are classified from the level of interaction view and based on the attacked resource type. The Low Interaction Honeypot (LIH) provides very low availability of the host OS. Most services and application are mocked and simulated in a static environment. Everything accessible is controlled by a decoy application with absolutely minimal in-depth features (e.g. shell, configuration files, other programs etc.). LIHs are more secure for the host, but far less capable or useful for malware/attack detection and inspection.

On the contrary, High Interaction Honeypot (HIH) is a fully responsive system with live applications and services with minimal to none emulated functionalities. It provides the attacker a wide attack surface ranking the HIH far less secure with the whole OS at the malicious actor's disposal. The idea is to make HIHs believable as possible and isolating it from production environment including virtualization [6].

Honeypots are differentiated by the type of attacked resource. The server-side honeypot is the well-known honeypot with running service(s) and monitoring the activity of the server-side connections. The attacked resources are the services listening on the dedicated ports. It's main purpose is to detect and identify botnets and forced authentication/authorization attempts.

The client-side honeypot is deployed as a user application, which utilizes the server's services. The monitored subject is the application (e.g web-browser, document editor). It's main purpose is to detect client-side attacks originating from the application (i.e. web-browser attacks via web pages and plugins).

1.1.2.2 Sandbox

1.2 Virtualization technology

1.2.1 Kernel Virtual Machine

1.2.2 Kubernetes

Literature

[0] Aditya Anand. "Malware Analysis 101 - Sandboxing. Cons of using a VM". Sept. 29, 2019. URL: https://medium.com/bugbountywriteup/malware-analysis-101-sandboxing-746a06432334 (visited on 03/27/2020).

[2] Ujaliben Kalpesh Bavishi et al. "Malware Analysis". In: *ijarcsse* (2017). URL: https://ijarcsse.com/index.php/ijarcsse/article/view/507 (visited on 03/26/2020).

[3] Manuel Egele et al. "A Survey on Automated Dynamic Malware-Analysis Techniques and Tools". In: *ACM Computing Surveys. Article 6* 44.2 (Feb. 8, 2012). Article 6, pp. 5–21. DOI: https://dl.acm.org/doi/10.1145/2089125.2089126. URL: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.640.6356&rep=rep1&type=pdf (visited on 03/26/2020).

[4] Tomasz Grudziecki et al. "Proactive Detection of Security Incidents - Honeypots". enisa study. Nov. 2012. URL: https://www.enisa.europa.eu/publications/proactive-detection-of-security-incidents-II-honeypots.

[5] Vladislav Hrčka. "Stantinko's new cryptominer features unique obfuscation techniques". Mar. 19, 2020. URL: https://www.welivesecurity.com/2020/03/19/stantinko-new-cryptominer-unique-obfuscation-techniques/ (visited on 03/23/2020).

[6] Lenny Zeltser. “5 Steps to Building a Malware Analysis Toolkit Using Free Tools”. SANS Institute instructor. Mar. 2015. URL: https://zeltser.com/build-malware-analysis-toolkit/#allocate-virtual-systems.