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

FIIT-XXXXXX-82385

Tomáš Belluš

Immutable infrastructures for security demanding environments

Výskumný zámer

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.

December 2019

Contents

Chapter 1

Motivation

Threat intelligence research often focuses on solving a variety of questions important for better understanding the paradigms and techniques used in securing these systems, some of the more common are:

1. How can systems be exploited?
2. How does a malicious actor obtain foothold inside the system?
3. What techniques could the malicious actor use?

There is a variety of approaches to identifying practices and techniques used by the malicious actors, e.g. post-mortem incident analysis [6] or proactive malicious actor baiting using honeypots and honeynets [1].

Malicious actors follow a multi-phase process [12] [13] to completely plan the sophisticated attack. The phases fall under so called Cyber/Intrusion Kill Chain and include - reconnaissance, weaponization, delivery, exploitation, installation, command and control, actions on objective. Briefly, the attack begins with analyzing and discovering the target objective, followed by preparation of the exploit to compromise the system and getting required control to fulfill a goal. The idea is related to the US military process of targeting (F2T2EA), where any failure breaks the chain [8].

As well as the malicious actors, security experts can use equally complex tools and practices to detect, mitigate or reproduce an attack. Tools like sandboxes, sinkholes, honeypots and many active detection tools (e.g. WAF, IPS/IDS, FW, etc.).

Typical practice among security experts is to study the hacking practices by practicing ethical hacking often abbreviated as a "red" team. While

[1] Honyenet is a robust network honeypot consisting network components used to detect network vulnerabilities [11].

the opposing team is known as the "blue" team - mitigating, detecting and stopping the activity. In the "fight" of blue versus red, the blue team must keep up with the red teams' new state-of-the-art techniques and zero-day exploitations. Zero-day exploitation targets vulnerabilities not yet addressed by vendors and not publicly disclosed [14]. Such vulnerabilities are almost impossible to mitigate and very difficult to detect. White hat or ethical hackers - hackers looking for exploits with no intent to harm - search for these vulnerabilities in their own environments by reading all those lines of code or dynamically - utilizing honeypots.

Crucial element of a honeypot is the ability to quickly reconfigure or duplicate it into a new environment. Regarding that honeypots are vulnerable to zero-day vulnerabilities, it must be mitigated either by catching th maliscious actor on the peripheral systems (via Firewall ACLs) or by the ability to recover from the breakdown by restoring the honeypot's state.

Chapter 2

Analysis

"The primary technology that makes immutable infrastructure possible at any scale is virtualization (both software and hardware) across networking, servers, and storage" [10]. According to this, virtualization is the inseparable element of immutable infrastructure. In terms of security, it operates an isolation layer protecting the underlying layers.

2.1 Best practice in infrastructure orchestration

2.1.1 Evolution

The evolution of virtualization and rise of containers to even more isolated services (i.e. concepts Software-as-a-Service and Function-as-a-Service) led to resource sharing and its maximum utilization. Application and services are divided compared to the traditional monolithic architectures with all services stacked in one server [7].

2.1.2 Immutable and idempotent infrastructure

To understand the applications of immutable infrastructure one must identify its characteristics. Any system or unit of a system (i.e. container, VM, server) is pre-configured and deployed to its final state in a target environment. The state is immutable - it forbids any further configuration changes or additional networking. Required changes are applied in the *infrastructure configuration* (e.g. container images) and redeployed to apply the updated state. In comparison, a mutable infrastructure is the exact opposite, such as almost every computer, that requires manual step-by-step

installation of the OS and the underlying applications and network configurations.

"In mathematics, a number that is idempotent keeps the same value when multiplied by itself, no matter how many times the function is applied" [1]. In computing, it is an operation executed only when the result will differ [2]. Having a complex infrastructure composed of multiple services allows segmentation. Instead of upgrading/modifying a whole infrastructure, only the desired segments are addressed - idempotent infrastructure [3]. It resides with orchestration and the security of segmentation of services (refer to section 2.1.2). Orchestration is the question of how, when and what combined in a single process of the mechanism used.

The immutability has a profound effect on the security of the infrastructure [9]. Since every change is made by redeploying the infrastructure, all footprints of the malicious actor vanish despite possibly being capable of outlasting a reboot.

The immutable infrastructure paradigm ensures the whole environment may be recovered. Usage of immutable infrastructure may introduce the possibility of designing a complex honeynet composed of multiple interconnected network segments and devices. As the whole infrastructure is described in a configuration, the modification of a service is the matter of changing or setting a variable. When such configurations, utilizing the immutability, is used in production, the deployment of mirror infrastructure is matter of configuring the right hosts.

Escape attack mitigation

Escape attacks exploit vulnerabilities in shared resources or stack layer (i.e. kernel for KVM, application for functions or hardware for virtualization). A case study, with a goal to mitigate any form of escape attacks, at the Swedish Police Authority by Christian Abdelmassih - *Docker and Kubernetes in high security environments*. The author discusses application isolation techniques while providing orchestration with Kubernetes.

One technique is to isolate applications by containerization (e.g. Docker, LXC, etc.), which share the host's kernel. Second technique is separation by (bare-metal) hypervisor - virtual machines (VM). This means that the VM has it's own operation system (OS) and shares hardware resources of the host. In conclusion, the idea is to utilize bare-metal hypervisor (cloud native) to minimize the attack surface for escape attacks, because containers are vulnerable to kernel driven escape attacks.

To separate non-related application and utilize containers, he suggests segmentation of Kubernetes pods to logical classes (e.i. class O, class P and

class PG) [4]. These classes are to be organized and orchestrated by Kubernetes and should make sure nodes and pods have the same class tags. Even though the presented solution lacks verification, it introduces well defined concepts, possible applicable for my thesis, as the the author finalizes that "In order to utilize this in clouds in a scalable manner, there are additional requirements for automation that must be satisfied. For example, automating the creation of virtual machines, attaching them to the Kubernetes cluster. Most importantly one must also implement and verify that the application classifications are respected at all times" [5].

2.1.3 Honeypots & Sandboxes

Both honeypots and sandboxes are similarly used in dynamic threat analysis. Difference is in preparation and isolation. Honeypot monitors the activity of a malicious actor or malware in an emulated or production-like environment. Sandbox is an isolated environment for dynamic malware analysis with all preparation that is needed based on the knowledge of the malware. Usually, honeypots are configured to detect specific attacks with emulated applications or not emulated to detect zero-day vulnerabilities and monitor behavior of potential exploitations [15]. Within a sandbox, all activity is monitored as much as possible, while knowing the outcome of the attack, to know the source of the attack.

A particularly interesting topic is to *detect zero-day vulnerabilities* in a honeypot and, if necessary, replicating and more in depth analyzing the exploitation in a sandbox. Analyzing malware in a sandbox provides virtualization, which mitigates MBR overwrite. On the other hand, most malwares verify the environment they're in - detection of virtualization or sandbox awareness.

Chapter 3

Master's thesis overview

3.1 Research questions

How to improve virtualization/emulation detection in a honeypot or a sandbox?

What is the most detectable phase in the Cyber Kill Chain?

The underlying technology for virtualization and containerization (if any) is crucial to reach immutability. What provisioning and orchestration mechanisms are needed to ensure an immutable infrastructure?

3.1.1 Thesis goal

Design a honeypot or a sandbox, in Linux, as a dynamic analytic tool dedicated to detecting zero-day vulnerabilities and mitigating detection. Utilize the state-of-the-art infrastructure deployment and orchestration paradigms to ensure automated and a secure system. Mitigating escape attacks will not be in scope of this thesis, but the introduced segmentation technique will be taken into account to segregate related services.

3.1.2 Plan of work

DP I

- analyze virtualization (e.g KVM), emulation and virtualization detection mitigation practices
- analyze cross-platform orchestration and provisioning mechanisms
- comparison of existing solutions and discussion of possible improvements

- design of ensuring immutability and the whole stack of the system supported by analysis

DP II

- discuss possible attack vectors meant for monitoring
- implementation of the system skeleton and configuration of the orchestartion mechanism
  - create base images for the system
  - automated deployment
- document the implementation
- assignment revision of necessary

DP III

- test honeypot implementation on live server
- analyze all retrieved data and indicators of compromise
- apply required modifications to images or core functionality
- finalize the thesis documentation and evaluate reached goals

Chapter 4

Further Literature

1. paper about *Zero-Day Attack Signatures Detection Using Honeypot* https://pdfs.semanticscholar.org/d4e4/5e81b8d8878ca99648c3fc890ede1ae01b
pdf
2. thesis about *Docker and Kubernetes in high security environments* https://kth.diva-portal.org/smash/get/diva2:1231856/FULLTEXT02.pdf
3. paper about *Adaptive and Flexible Virtual Honeynet* https://www.researchgate.net/publication/285598988_Adaptive_and_Flexible_Virtual_Honeynet
4. conference paper about *Automated Configuration of Monitoring Systems in an Immutable Infrastructure* https://link.springer.com/chapter/10.1007%2F978-3-030-01171-0_21
5. thesis on *Dynamic shifting of virtual network topologies fornetwork attack prevention* https://digitalcommons.calpoly.edu/cgi/viewcontent.cgi?article=3389&context=theses
6. paper on *Detecting honeypots and other suspicious environments* https://ieeexplore.ieee.org/abstract/document/1495930
7. thesis on *Design and Implementation of a Real-TimeHoneypot System for the Detection andPrevention of Systems Attacks* https://repository.stcloudstate.edu/cgi/viewcontent.cgi?article=1011&context=msia_etds

Bibliography

[1] https://www.vocabulary.com/dictionary/idempotent.

[2] https://www.techopedia.com/definition/14420/idempotence.

[3]

[4] Christian Abdelmassih. Container orchestration in security demandingenvironments at the swedish police authority. https://kth.diva-portal.org/smash/get/diva2:1231856/FULLTEXT02.pdf [8.12.2019].

[5] Christian Abdelmassih. Docker and kubernetes in high security environments. https://medium.com/@chrismessiah/docker-and-kubernetes-in-high-security-environments-d851645e8b99, January 2019.

[6] Tim Bandos. Learning from a security incident: A postmortem checklist. https://digitalguardian.com/blog/learning-security-incident-post-mortem-checklist, November 2016. [9.12.2019].

[7] Darryl Bowler. The rise of microservice architecture and its implications to continuous (business) delivery and devops 2.0. https://www.accenture.com/us-en/blogs/blogs-microservice-architecture-devops [9.12.2019].

[8] et al. Eric M. Hutchins. Intelligence-driven computer network defenseinformed by analysis of adversary campaigns andintrusion kill chains.

[9] Rob Hirschfeld. Thirteen ways containers are more secure than virtual machines. https://thenewstack.io/thirteen-ways-containers-secure-virtual-machines/. [9.12.2019].

[10] Sumo Logic. Mutable and immutable infrastructure. https://www.sumologic.com/insight/mutable-immutable-infrastructure/, May 2019. [8.12.2019].

[11] et al. Naomi J. Alpern. Learn more about honeynets, 2010. https://www.sciencedirect.com/topics/computer-science/honeynets [9.12.2019].

[12] Graig Reeds. The seven phases of a cyber attack. https://maritime-executive.com/blog/the-seven-phases-of-a-cyber-attack, 2018.

[13] Lance Spitzner. Applying security awareness to the cyber kill chain. https://www.sans.org/security-awareness-training/blog/applying-security-awareness-cyber-kill-chain.

[14] Symantec. Zero-day vulnerability: What it is, and how it works. https://us.norton.com/internetsecurity-emerging-threats-how-do-zero-day-vulnerabilities-work-3html. [8.12.2019].

[15] et al. Tomasz Grudziecki. Proactive detection of security incidents.