Modern DevSecOps Security

Needed Security Services for Microservices-based Applications (MSAs)

Felix Seifert

Essay for course
Automated Software Testing and DevOps

KTH Royal Institute of Technology
Stockholm, Sweden
April 27, 2021

Contents

1 Introduction

A Microservices-based Application (MSA) distributes the complexity of the whole application to multiple microservices, which together form the application [1]. Each microservice can use an arbitrary environment and are deployable independently from each other and hide their internal implementation details [2, 3]. Furthermore, an MSA with its functionalities for rebalancing and scaling is more dynamic than monolithic applications [4]. MSAs have therefore several advantages over classical monolitic applications.

DevOps is a term that describes the merger between the needs of development and operations to collaboratively develop software and deploy it to production [5]. The term *DevSecOps* attempts to satisfy the need of security in the DevOps process through integrating security experts into the team of developers and operators [5]. The main concerns of customers when adopting cloud computing services like MSAs and benefiting from their advantages are security and privacy [6]. With the rising popularity of MSAs, ensuring security and privacy for an MSA becomes an important part of the DevSecOps processes. This essay will focus on security without drawing attention to privacy and therefore leads to the research question: *What are the needed services for DevSecOps engineers to provide security for an MSA?*

2 Security

Figure 1 describes the security terms and their dependencies: An attack which is performed by an adversary, i.e. an attacker, causes a threat [7]. Threats harm the asset which should be protected; the idea of an adversary is therefore to harm the asset. In the case of MSAs, internal and external attackers could both try to harm the system and its data.

To resolve existing vulnerabilities of a naive MSA architecture, which has no implemented security measures, this essay finds and describes several countermeasures to achieve the subsequently stated security requirements. On one hand, countermeasures can resolve vulnerabilities. That means that these vulnerabilities cannot be used to cause threats anymore which means that countermeasures avoid intrusion [7]. On the other hand, countermeasures can remove threats as soon as they appear, i.e. they do not result in an incident and therefore tolerate intrusion [7]. The aim of countermeasures is always to prevent attacks performed by adversaries to enforce the security of the protected asset. Thus, countermeasures are also called security measures.

This essay uses the term security as information security. It consists of three requirements, which are aptly named as the CIA Triad [7, 8], with “C”, “I” and “A” representing confidentiality, integrity and availability respec-

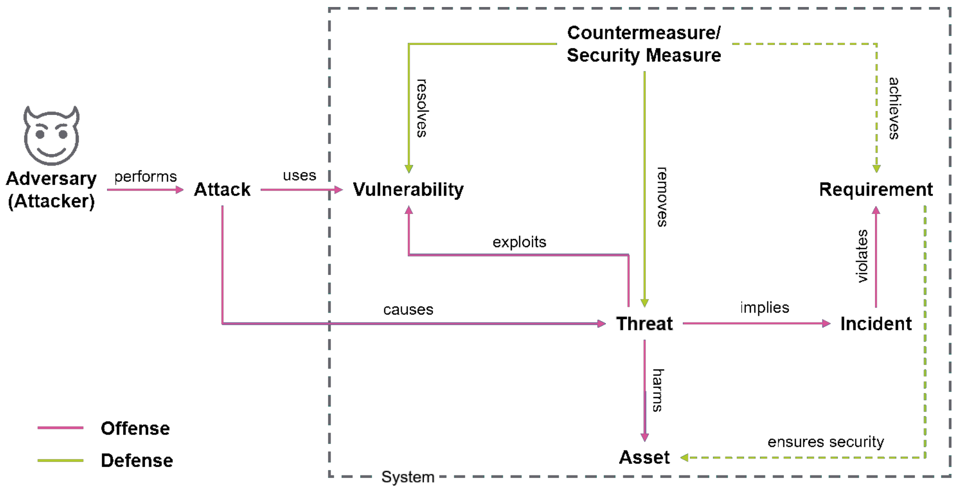

Figure 1: Security Terms and Their Dependencies (Adapted From [7])

tively [9]. These three requirements are not sufficient to ensure the security of the asset on their own [7]. They are the primary security requirements which can be expanded and refined by secondary security requirements [9]. The secondary requirement of authorisation refines the primary requirement confidentiality as a proper authorisation is required to ensure the confidentiality with several different entities. The secondary requirements authenticity and non-repudiation refine the primary requirement integrity. Accountability is added as another requirement to the CIA triad to allow an ex-post digital forensic [7].

- **Confidentiality**: Only authorised entities can read the information [8].
  - **Authorisation**: Only entities which have the right to access the information could perform certain actions on them [7].
- **Integrity**: The information is exact and unauthorised users cannot modify it [8].
  - **Authenticity**: The information is what it claims to be, especially that its named origin is its true origin and it did not get changed down the data stream [7].
  - **Non-repudiation**: The information's author cannot dispute its authorship [7].
- **Availability**: The information is accessible and usable by any authorised entity in a perpetual manner. [8]
- **Accountability**: The logging must include all activities which were triggered by the user to allow an ex-post forensic analysis [7].

3 Security in MSAs

Even though security in an MSA is often a compromise between costs and preventing more attacks via resolving more vulnerabilities [10], security is of the utmost importance to ease customers' concerns [6]. A common way of securing an MSA was adopting perimeter security [3]. Nevertheless, modern perspectives consider this method as insufficient [10]. Taking the more modern approach for microservice security of not considering the whole application as the trusted computing base ("trust no one") prevents more successful attacks and one compromised microservice cannot compromise the rest of the MSA [10]. Applying multiple, partly overlapping security measures is therefore judicious.

An MSA architecture requires multiple layers to be secured. An MSA has six different layers which are relevant for security: communication, application, service orchestration, hardware, virtualisation and cloud environment [10]. Subsequently, you find a description of these six layers.

3.1 Communication

The independent microservices of an MSA interact via sending messages [1]. Additionally, MSAs also communicate with the rest of the external world of the internet. Synchronous and asynchronous messaging are the two categories of communication of an MSA [11]. Asynchronous messaging is based on the publish-subscribe pattern[1] [11]. Asynchronous messaging protocols are non-blocking which means that the requesting service does not wait for an answer from any recipient and just goes on with its tasks. Synchronous messaging is based on the request-reply pattern[2] [11]. With blocking protocols for synchronous communication like Hypertext Transfer Protocol (HTTP) with a Representational State Transfer (REST) Application Programme Interface (API), the request-issuing service waits for the called service's response before continuing its task. In MSAs, HTTP REST API calls are favoured over communication via the Simple Object Access Protocol (SOAP) [12]. As SOAP can be both synchronous and asynchronous, this might suggest that synchronous communication is preferred. Nevertheless, asynchronous communication increases efficiency [13]. No matter which communication method is used, an MSA increases its attack surface by offering communication over universal APIs which are not bound to a specific programming language [1].

[1]Multiple services can subscribe to certain event types. Other services can publish events of this type to trigger certain actions of the subscribed services. A notification service, usually a message broker, creates the interconnection.

[2]One service calls the address of another service, waits for its response and then processes the received response.

3.2 Application

The application layer does not only describe the whole MSA but also every single component of it. As the trusted computing base is not the whole MSA but each component of this system, each component receives a similar protection like a normal monolithic application. With the communication layer securing the communication channels within the MSA and communication between the MSA and the external world of the internet, the application layer is responsible for restricting these channels only for authorised entities to prevent attacks (e.g. denial-of-service attacks) [10].

3.3 Service Orchestration

Service orchestration describes the composition of the different microservices which is needed to achieve both the MSA's goal as well as the MSA's technological requirements, i.e. how the MSA's components work together or at least, should work together. This could involve a service discovery for locating the system's components via a service registry and coping with the ever-changing network structure, i.e. different components of the system being stopped, restarted or relocated [10].

3.4 Hardware

Computing hardware is the fundamental building block of any computing system because each computing software runs on hardware. Cloud computing abstracts away from the hardware and pools several resources together into a hardware stack to make them available for an arbitrary number of customers [14] and several different applications co-exist on the same stack [12]. Manipulating the hardware layer can therefore have a great impact on influencing the security of a cloud computing system.

As the hardware layer is not specific to MSAs and this essay does not deal with safety, this essay excludes the hardware layer.

3.5 Cloud Environment

The model of cloud computing gives customers access to a shared pool of computing resources which are gathered in a cloud environment [14]. Even though the responsibility for the security of the underlying cloud system is given to the cloud service provider [6], securing the cloud environment and dealing with the increased network complexity of MSAs [1] are not specific to securing an MSA and therefore, this essay does not discuss this layer further.

3.6 Virtualisation

A benefit of virtualisation is to abstract away from the infrastructure [14] to achieve hardware independence and eliminate security issues through software isolation. While eliminating certain security issues through software isolation, the popular containerisation technologies which are employed for cloud computing introduce a new software layer and therefore, create new attack vectors [15]. To mitigate containerisation-induced vulnerabilities, each microservice must have a micro-firewall to protect them against each other. Virtualisation is a wide research topic in the field of not only MSAs but also monolithic applications which leads to this essay excluding this layer.

4 Needed Security Services

The aforementioned layers of an MSA from section 3 have to fulfil several demands to prevent attacks which cause harms to the asset. Figure 2 shows the mapping between the security requirements which are explained in section 2 and the needed security services to resolve vulnerabilities used by attackers. Table 1 then maps the security services to the different MSA layers.

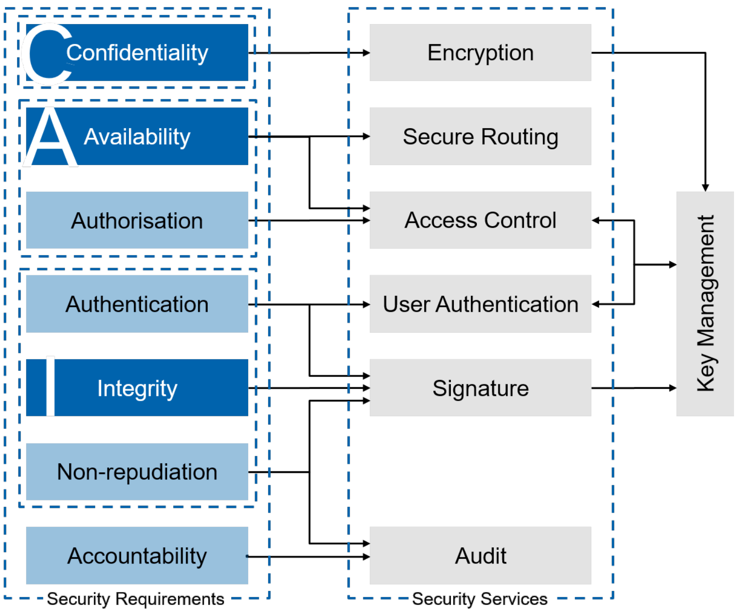

Figure 2: Mapping Between Security Requirements and Needed Security Services (Adapted From [7])

Table 1: Mapping Between MSA Layers and Security Services

MSA Layer	Security Requirement	Security Service
Communication	Confidentiality, authorisation, availability, integrity, authenticity, non-repudiation	Encryption, signature, user authentication, access control
Application	Authorisation, availability, authenticity, non-repudiation, accountability	User authentication, access control, audit
Service orchestration	Availability	Secure routing

5 Conclusion

This essay points out the three layers of an MSA which have to be secured to lower the aversion of customers concerning security. Table 1 provides an overview of the needed services to guarantee the security requirements of section 2 and therefore answers the research question of the needed security services for DevSecOps engineers.

References

[1] N. Dragoni, S. Giallorenzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, and L. Safina, "Microservices: Yesterday, Today, and Tomorrow," in *Present and Ulterior Software Engineering.* Cham: Springer International Publishing, 2017, pp. 195–216.

[2] J. Lewis and M. Fowler, "Microservices: A Definition of This New Architectural Term," 2014. [Online]. Available: https://martinfowler.com/articles/microservices.html

[3] S. Newman, *Building Microservices: Designing Fine-Grained Systems*, first edit ed. O'Reilly, 2015.

[4] R. Heinrich, A. van Hoorn, H. Knoche, F. Li, L. E. Lwakatare, C. Pahl, S. Schulte, and J. Wettinger, "Performance Engineering for Microservices," in *Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering Companion.* New York, NY, USA: ACM, apr 2017, pp. 223–226.

[5] H. Myrbakken and R. Colomo-Palacios, "DevSecOps: A multivocal literature review," in *Communications in Computer and Information Science*, vol. 770. Springer Verlag, 2017, pp. 17–29.

[6] H. Takabi, J. B. Joshi, and G.-J. Ahn, "Security and privacy challenges in cloud computing environments," *IEEE Security and Privacy*, vol. 8, no. 6, pp. 24–31, 2010.

[7] C. Esposito and M. Ciampi, "On Security in Publish/Subscribe Services: A Survey," *IEEE Communications Surveys and Tutorials*, vol. 17, no. 2, pp. 966–997, apr 2015.

[8] J. Andress, *The Basics of Information Security: Understanding the Fundamentals of InfoSec in Theory and Practice*, second edi ed. Syngress, 2014.

[9] A. Avižienis, J. C. Laprie, B. Randell, and C. Landwehr, "Basic Concepts and Taxonomy of Dependable and Secure Computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11–33, jan 2004.

[10] T. Yarygina and A. H. Bagge, "Overcoming Security Challenges in Microservice Architectures," in *2018 IEEE Symposium on Service-Oriented System Engineering (SOSE).* IEEE, mar 2018, pp. 11–20.

[11] R. Chandramouli, "Security Strategies for Microservices-based Application Systems," National Institute of Standards and Technology, Tech. Rep., aug 2019.

[12] A. Nehme, V. Jesus, K. Mahbub, and A. Abdallah, "Securing Microservices," *IT Professional*, vol. 21, no. 1, pp. 42–49, jan 2019.

[13] P. Siriwardena and N. Dias, *Microservices Security in Action.* Manning, 2020.

[14] P. M. Mell and T. Grance, "The NIST definition of cloud computing," National Institute of Standards and Technology, Gaithersburg, MD, Tech. Rep., 2011.

[15] T. Combe, A. Martin, and R. Di Pietro, "To Docker or Not to Docker: A Security Perspective," *IEEE Cloud Computing*, vol. 3, no. 5, pp. 54–62, 2016.