Microservice Architecture: Security Issues and Mitigation Approaches

Peter Harvey
Department of Computer Science
Seattle University
Seattle, US
pharvey@seattleu.edu

Narissa Tsuboi
Department of Computer Science
Seattle University
Seattle, US
tsuboin@seattleu.edu

***Abstract*— A microservice architecture is a loosely coupled network of microservices, deployed in a cloud-native application [1]. This architecture is still considered new and is rapidly implemented to achieve high scale business needs. The speed and scalability of microservice architecture is well established, but what are the challenges of securing this complex, rapidly changing architecture? The aim of this literature review is to summarize the current issues in microservice security and find best practices that multiple sources agree on. This literature review was conducted on four white literature sources, deemed "core papers" and found that there are many issues associated with microservice architecture, the most of which occur at the microservice and application layer. Best practices and standards are not well recorded and accepted across the industry, but some sources agreed on multiple best practices. This paper can be used to assess the issues and best practices in microservice security from a mix of white and grey literature.**

***Keywords*—microservice software architecture, security**

I. Introduction

A microservice architecture is a loosely coupled network of microservices, deployed in a cloud-native application [1]. A microservice is an independent service that aligns with a business domain and can be developed independently of others, increasing development speed and flexibility. Microservices exchange information through their own API interfaces across a network and must handle authorization and authentication to its endpoints. Overall, microservice architecture offers robustness, ease of deployment, speed, and scale – all significant advantages for companies operating at scale [2]. The speed and scalability of microservice architecture is well established, but what are the challenges of securing this complex, rapidly changing architecture?

In this paper – we surveyed microservice architecture security issues, their causes, and mitigations, asking:

- RQ1. What microservice security issues exist? What are their causes?
- RQ2. What strategies do practitioners employ to address the security issues identified in RQ1?

We addressed these research questions by searching for core papers fundamental to the latest literature on microservice architecture and security, analyzing the papers and writing a literature review of the four papers. We used Seattle University's PRIMO discovery service to gain access to peer-reviewed literature, books, and articles hosted in partner databases as well as Google Scholar. Our search produced 11 candidate papers. Then, we screened the papers for relevance to the research questions by following the first and second passes in *How to read a paper* by S. Keshav [3]. After this screening, we identified the four core papers and performed a literature review from the perspective of RQ1 and RQ2.

Our contributions are a synthesis of two systematic literature reviews, one empirical study on microservice best practices, and a case study of a security framework. This paper identifies issues in securing microservices presented in these core papers and aggregates best practices that overlap between the papers. This paper can be used to assess the issues and best practices in microservice security from a mix of white and grey literature.

The paper is organized as follows: In Section 2, we present four core paper analyses. Section 3 synthesizes the core paper analyses and their contributions. In Section 4, we present an overview of the current state of the art for microservice architecture security. In Section 5, we conclude the paper and describe our ideas for future directions for microservice architecture security research.

II. Core Paper Analysis

A. *SoK: Security of Microservice Applications: A Practioners' Perspective on Challenges and Best Practices*

This systematic grey literature review aggregated and distilled current microservice security discussions from microservice practitioners to identify improvements to existing methodologies for microservice security. The motivation for a grey literature review was to focus on literature that is not published or available commercially with the goal of filling the gaps in the white literature on this topic. The study posed the following research questions: (1) What are the challenges reported by practitioners in the field of microservice security and (2) how practitioners address these challenges and what are their recommendations to overcome them.

To answer these questions the authors conducted a systematic literature review (SLR) following the "Guidelines for conducting systematic mapping studies in software engineering: An update" by Petersen et al. including study selection and quality assessment, data extractions, analysis and classification, and validity evaluation. Studies were tested for reliability using the CRAAP test (Currency, Relevance, Accuracy, Authority and Purpose); and coded using the grounded theory approach. The resulting references included in the study were from 2015 to 2021, as no search results existed for 2011 through 2014 because of how new the concept was. The corpus consists of blogs and technical articles, followed by Q&A forums, code repositories, and presentations.

The SLR identified challenges, best practices, and technical solutions in the grey literature. In order of frequency, the 7 categories of challenges for securing microservices were trust between services, large attack area, testing, container management, low visibility, secret management, and polyglot architecture. The best practices were defense in depth, encrypt sensitive data, DevSecOps, immutable container, least privilege permissions, secure-by-design, and rate throttling. The technical solutions were standards (Oauth, Oauth 2.0), protocols (Open ID connect, mTLS), tokens (JWT, PASETO) and other like MFA.

The results of this SLR show alignment and misalignment areas and overlooked areas between white and grey literature on microservice security. For example, this SLR found agreement between white and grey literature on the use of immutable containers to protect microservices from security threats during deployment. An example of misalignment was found on the perceived benefits of using polyglot architectures. White literature encourages the use of polyglot architectures to lower susceptibility to lateral attacks; however, grey literature disagrees, stating that the additional security challenges introduced like different life cycles of services built in different stacks, versioning issues, and the required expertise to maintain a polyglot microservice architecture. Two overlooked areas were identified: (1) principle of least privilege and (2) rate throttling. The principle of least privilege ensures that each role under a Role-Based-Access-Control schema has access to the least number of resources, which increases protection from internal attacks. Rate throttling slows down the response time of a microservice if the service's typical behavior has deviated. Rate throttling can be used to catch the start of an infrastructure attack like a DDOS attack.

This systematic literature review identified challenges, best practices, and technical solutions for microservice security. It also identified overlooked areas in white literature - highlighting the need to regularly survey the grey literature on microservices to hear from real microservices practitioners what best practices are effective.

B. *Smells and refactorings for microservices security: A multivocal literature review*

This core paper conducted a multivocal review of white and grey literature on microservice security, distilled the 10 most common smells for securing microservices into a taxonomy with direct associations to the ISO/IEC 25010 security property each smell violates, and presented the recommended refactoring as identified in their multivocal review.

This study performed a literature review following a systematic approach based on Kitchenham and Charters Guidelines for performing Systematic Literature Reviews in Software Engineering with variation added for grey literature. By utilizing both white and grey literature, the results of this study include findings for both state-of-the-art *and* state-of-practice security smell refactoring. Valid sources included at least one security smell and a technical solution for resolving the smell. The study analyzed 58 sources where 40 were grey literature.

The results of the coded papers from the literature review showed that only three out of the five ISO/IEC 25010 security properties directly corresponded to a security smell which were, in order of frequency, confidentiality, integrity, and authenticity. The top three covered security smells were (1) non-secured service-to-service communications, (2) insufficient access control, (3) publicly accessible microservices. The associated refactoring for each, using mutual transport layer security (TLS) for non-secured service-to-service communications, are using OAuth 2.0 for insufficient access control, and utilizing an API gateway for a microservice instead of making it a global instance publicly available.

In summary, the most covered security smells corresponded to confidentiality, integrity, and authenticity (ISO/IEC 25010) and service-to-service interactions and communication.

C. *An empirical study of security practices for microservices systems*

The purpose of the study described in this paper was to document microservices-specific security practices and to understand to what extent practitioners in the field consider these practices useful.

This study conducted an empirical study followed by a survey of microservices practitioners. This involved collecting and manually analyzing 861 microservices security points, which included 567 issues, 9 documents, and 3 wiki pages from 10 GitHub open source microservices systems and 306 stack overflow posts concerning security in microservices systems. Three phases of identification – a pilot phase, main phase, and feedback phase, by analysts were used to identify and extract security practices from these microservices security points. Finally, an industrial survey completed by 74 microservices practitioners was conducted to seek their perceptions about the usefulness of the identified security practices. The study discovered 28 microservices security practices grouped into 6 categories (Authorization and Authentication, Token and Credentials, Internal and External Microservices, Microservices Communications, Private Microservices, and Database and Environments). Results from the industrial survey show that the survey participants deemed all 28 identified practices as useful. However, the study acknowledges the difficulty in adopting all 28 practices and thus highlights what are considered to be the 8 most important practices and asserts that not all practices are appropriate for all contexts and domains. The authors also make recommendations for future studies – to study how the identified practices are used in different domains and contexts, to understand why some practices are controversial, and to pay attention to security practices in other or less explored aspects of microservices systems.

As the study itself acknowledges, there are a few points of contention on whether the findings of the study are valid. One is the fact that the microservices security points were only identified from two sources – GitHub and Stack Overflow. While these are among the most popular platforms for discussing the microservices architecture, there are numerous other platforms or resources that would have added value had they been included. To extend on this, a few of the selected GitHub repositories, specifically "eShopOnContainers" and "microservices-demo" are demo repositories to aid in the demonstration of microservices technologies. While the security points discussed regarding these repositories may be valid, the fact that they are not deployed and exposed to real-word security threats brings into question whether they should have been included in this study or not. Regardless, the findings of the study are valuable reference points when designing, building, and maintaining a microservices architecture.

D. Overcoming Security Challenges in Microservice Architectures

This paper undergoes a holistic approach to provide a taxonomy of microservices security, assess the security implications of the microservice architecture, and survey related contemporary solutions for microservice security.

This study begins with the analysis of security implications of microservice design through a careful literature review of the subject, discussions with practitioners, and personal experience. Next, trends in industry practice regarding microservice security are examined. Finally, a microservice security framework called MiSSFire was implemented that provides a standard way to embed security mechanisms into microservices. The performance of this framework was evaluated using a toy microservice-based bank system to determine the extent to which usage of the security framework downgrades the throughput of the bank system.

The important security properties determined by this paper's literature review include (1) do one thing and do it well, (2) automated and immutable deployment, (3) isolation through loose coupling, (4) diversity through system heterogeneity, (5) fail fast. Prominent industry trends in microservices security highlighted by this paper include Docker Swarm and Netflix's use of Mutual Transport Layer Security with a self-hosted Public Key Infrastructure, principal propagation via security tokens, and fine-grained authorization. Finally, the results of the performance analysis done on the open source microservice security framework MiSSFire show that there is little extra cost in using the framework, likely due to the overall higher overhead of communication between microservices.

Overall, this study takes a well-rounded approach to defining the important properties of microservice security and presents a few examples of effective designs in industry. The open-source framework for microservice security seems to be valuable, but its analysis is limited to the performance cost of using it within a microservice system. The analysis and discussion of this security framework could be improved by going more in depth into its design, and the possible questions to its validity.

III. SYNTHESIS

In this literature review we found that issues in securing microservices can be attributed to the nature and properties of a microservice architecture itself as well as a lack of securing microservice design as it is developed.

Issues relating to the nature of the microservice architecture include not proactively securing the large attack area, the novelty of the architecture and developer inexperience, lack of frameworks in securing the architecture, and the rapid development of microservice systems. For example, the use of API endpoints to communicate with a microservice creates a large attack area. The ability for each microservice to host its own interface and endpoints to be reached across the network is a foundational premise of the architecture, and as such, a large attack area is created by default. Additionally, with microservices being a relatively new architecture, developer inexperience compounded with the lack of frameworks to follow may be a cause of security issues later on it development [4]. Lastly, microservice architectures are typically used to implement highly scaled business objectives in a limited development period. Due to the rapid nature of microservice architecture development in this context, security and testing for security may not be prioritized.

Issues relating to developers not proactively securing microservices during design were also prevalent in our review. The two systematic literature reviews both generated taxonomies of issues that can be avoided by implementing a security at the outset – what called "defense in depth". Additionally, each literature review's findings showed that security issues more frequently arise at the service or application layers during service interactions and communications. They also agree that confidentiality, integrity, and trust are the most infringed upon objectives in microservice architecture.

The best practices outlined by the SoK literature review were the most abstract and conceptual guidelines for securing microservices. For example, the "defense in depth" best practice states that "sensitive resources should be safeguarded by multiple levels of security measures…". It also suggests encrypting sensitive data but does not recommend how. In contrast, the refactorings presented in and the best practices in the empirical study on best practices for securing microservices presented more detailed suggestions. See Table 1 below for areas of overlap between their best practices.

Synthesized Best Practices	SoK [4]	Smells [5]	Empirical [6]	Overcoming security challenges [7]
Use a token / credential method (OAuth, OAuth2.0, JSON Web Token, certificates)	✓	✓	✓	✓
Use an API Gateway to secure, authorize, and route	✓	✓	✓	
Encrypt sensitive data	✓	✓	✓	
Use Mutual TLS for Service-to-Service Communication	✓	✓	✓	✓

Table 1: Synthesized best practices in microservice security.

IV. OVERVIEW OF THE CURRENT STATE OF THE ART

Due to the microservice architecture being a relatively new concept, the state of the art of the security of microservice architectures has had limited time to go through multiple iterations and improvements. Three of our core papers - [5], [4], and [6] were all published in 2022 or 2023, meaning their findings on microservice security issues, causes, and mitigations are so recent they can be considered state of the art. These findings include issues like access control and data exposure, causes like architectural complexity and immature standards, and mitigations like using Oauth 2.0, Open ID connect, and API gateways. However, the authors of [5] did publish a follow-on empirical study on general issues, causes, and solutions in the microservice architecture, which includes those related to security [7]. They identified and classified 37 types of security issues into 4 subcategories (Authentication and Authorization, Access Control, Secure Certificate and Connection, and Encryption and Decryption) and described solutions for security issues similar to those from [5], [4], and [6] and those described in the synthesis of our paper.

Our core papers were recently cited in other relevant papers. [8] performs a systematic literature review for solutions to security challenges in cloud-native systems. In this effort, it cites the well-known smells of microservice security vulnerability, and their potential mitigations described in [6], as well as the 28 practices for protecting microservices highlighted and corroborated by practitioners in [5]. [5] is also cited in [9] in its discussion of microservice architecture security being an issue that required serious treatment when designing a microservice-based application used for monitoring cattle health in real time. [10] describes the design of Microusity, a tool for performing RESTful API testing on a specific type of microservice pattern and cites [11] in the need for testing support frameworks in RESTful APIs since they play a huge role in integrating microservice systems together.

V. Conclusion & Future Directions

Through the synthesis of [5], [4], [11], and [6] our paper identified four major categories of security issues that exist for a microservice architecture along with the underlying cause for each category. These categories include difficulty to establish trust (authentication/authorization), large attack area (an API for every microservice), DevOps, and testing. These result from the inherent complexity of a microservice architecture and weak standards, poor testing, and developer inexperience that occur during system development. Mitigation strategies for these security issues include using established security tools like OAuth 2.0, MTLS, and API Gateways along with prioritizing security throughout the system lifecycle.

For future work, we would explore polyglot architectures to better understand their security implications for a microservice architecture. Both [4] and [11] offer conflicting statements regarding the benefits and drawbacks of intentionally using polyglot architectures which makes this topic unclear for those wishing to prioritize security when designing their microservice systems. A study can be performed by surveying microservice practitioners to understand this issue.

More studies can be performed by surveying microservice practitioners to gain insight on industry practices that may not have been covered in the grey and white literatures analyzed in [5], [4], [11], and [6]. These surveys can also be used to find and do case studies on concrete examples of security breaches in microservice systems to understand if there were any underlying design patterns that led to vulnerabilities and allowed the breach to occur. Further surveys can be performed to understand the reason for security being a low priority during most microservice systems' lifecycles and possible ways to prioritize it.

Lastly, future work could include searching for more open-source or close-source microservice security frameworks similar to MiSSFire [11]. A study on these frameworks can be done by performing tests to compare their performance and effectiveness.

References

[1] J. Lewis and M. Fowler, "Microservices: a definition of this new architectural term," 25 March 2014. [Online]. Available: https://martinfowler.com/articles/microservices.html#footnote-etymology.

[2] S. Newman, "What Are Microservices?," O'Reilly Media, Inc., 2019.

[3] S. Keshav, "How to read a paper," *ACM SIGCOMM Computer Communication Review,* vol. 37, no. 3, pp. 83-84, 20 July 2007.

[4] P. Billawa, A. Bambhore Tukaram, N. E. Díaz Ferreyra, J.-P. Steghöfer, R. Scandariato and G. Simhandl, "SoK: Security of microservice applications: A practitioners' perspective on challenges and best practices," in *Proceedings of the 17th International Conference on Availability, Reliability and Security*, Vienna, 2022.

[5] A. Rezaei Nasab, M. Shahin, S. A. Hoseyni Raviz, P. Liang, A. Mashmool and V. Lenarduzzi, "An empirical study of security practices for microservices systems," *Journal of Systems and Software,* vol. 198, p. 111563, 2023.

[6] F. Ponce, J. Soldani, H. Astudillo and A. Brogi, "Smells and refactorings for microservices security: A multivocal literature review," *Journal of Systems and Software,* vol. 192, p. 111393, October 2022.

[7] "Understanding the Issues, Their Causes and Solutions in Microservices Systems," *arXiv preprint,* vol. 2302, no. 01894, 2023.

[8] M. S. Rahaman, A. Islam, T. Cerny and S. Hutton, "Static-analysis-based solutions to security challenges in cloud-native systems: systematic mapping study," *Sensors,* vol. 23, no. 4, 2023.

[9] I. Shabani, T. Biba and B. Cico, "Design of a cattle-health-monitoring system using microservices and IoT devices," *Computers,* vol. 11, no. 79, 2022.

[10] P. Rattanukul, C. Makaranond, P. Watanakulcharus, C. Ragkhitwetsagul, T. Nearunchorn, V. Visoottiviseth, C. Morakot and T. Sunetnanta, "Microusity: A testing tool for backends for frontends (BFF) microservice systems," *arXiv preprint,* vol. 2301, no. 11150, 2023.

[11] T. Yarygina and A. H. Bagge, "Overcoming security challenges in microservice architectures," in *2018 IEEE Symposium on Service-Oriented System Engineering (SOSE),* Bamberg, 2018.