SECURING MICROSERVICES AND MICROSERVICE ARCHITECTURES: A SYSTEMATIC MAPPING STUDY

A PREPRINT

Abdelhakim Hannousse
Department of Computer Science
Universté 8 Mai 1945, Guelma
BP 401, Guelma 24000, Algeria
hannousse.abdelhakim@univ-guelma.dz

Salima Yahiouche
Department of Computer Science
LRS laboratory, Badji Mokhtar University
BP 12, Annaba 23000, Algeria
yahiouche.salima@univ-annaba.dz

June 24, 2020

ABSTRACT

Context. Microservice architectures (MSA) are becoming trending alternatives to existing software development paradigms notably for developing complex and distributed applications. Microservices emerged as an architectural design pattern aiming to address the scalability and ease the maintenance of online services. However, security breaches have increased threatening availability, integrity and confidentiality of microservice-based systems.
Objective. A growing body of literature is found addressing security threats and security mechanisms to individual microservices and microservice architectures. The aim of this study is to provide a helpful guide to developers about already recognized threats on microservices and how they can be detected, mitigated or prevented; we also aim to identify potential research gaps on securing MSA.
Method. In this paper, we conduct a systematic mapping in order to categorize threats on MSA with their security proposals. Therefore, we extracted threats and details of proposed solutions reported in selected studies. Obtained results are used to design a lightweight ontology for security patterns of MSA. The ontology can be queried to identify source of threats, security mechanisms used to prevent each threat, applicability layer and validation techniques used for each mechanism.
Results. The systematic search yielded 1067 studies of which 46 are selected as primary studies. The results of the mapping revealed an unbalanced research focus in favor of external attacks; auditing and enforcing access control are the most investigated techniques compared with prevention and mitigation. Additionally, we found that most proposed solutions are soft-infrastructure applicable layer compared with other layers such as communication and deployment. We also found that performance analysis and case studies are the most used validation techniques of security proposals.
Conclusion. More researches are needed for securing MSA properly. We advocate more researches addressing internal attacks, proposing mitigation techniques and concerning more layers of MAS including communication and deployment.

***K*eywords** microservices · microservice architectures · security · systematic mapping

1 Introduction

Nowadays, systems are becoming more complex, larger and more expensive due to the rapid growth of requirements and adoption of new technologies. Moreover, due to competitors, many companies need to make changes to their systems as fast as possible and without affecting their systems availability. This requires appropriate designs, architectural styles and development processes. Software engineering provides different paradigms to partially meet those needs by decomposing software systems into fine-grained software units for better modularity, maintainability and reusability, and hence reduce time-to-market.

Recently, microservice architectures (MSA) [1] has emerged as a new architectural style allowing building software systems by composing lightweight services that perform very cohesive business functions. Microservices are the mainstay of MSA. A mircoservice is a fine-grained software unit that can be created, initialized, duplicated, and destroyed independently from other microservices of the same system. Moreover, micorservices can be deployed across heterogeneous execution platforms over the network. Using microservices enables high scalability and flexibility of large scale and distributed software systems.

Although the advantages brought by adopting microservice architectures in developing complex systems, MSA as a novel technology comes with many flaws [2] and security is one of the serious challenges that need to be tackled [1]. In fact, security is a longstanding problem in networking systems, but with microsevices, security becomes more challenging. This is due to the large number of entry points and overload on communication traffic emerged by decomposing systems into smaller, independent and distributed software units. Moreover, trusts cannot simply be established between individual microservices in the network that often come from different and unknown providers.

Due to the massive attacks reported on companies adopting MSA such as Netflix and Amazon[1], dealing with security breaches became an urgent need. Several works in the literature have noticed the need to investigate security of MSA [1, 3, 4]. However, security threats are diverse and are continually increasing. Security proposals are also increasing and varies from securing individual microservices into complete architectures and infrastructures.

In this article, we conduct a systematic mapping study to uncover the main threats menacing the security of microservice-based systems. We systematically identify existing studies addressing threats and proposing security solutions to MSA. We apply a thorough protocol to extract, classify, and organize reported threats with the security solutions proposed to mitigate and prevent them. The contributions of the study can be summarized as:

1. identify the most relevant threats concerning microservices and microservice architectures
2. point out the set of security mechanisms used to detect, mitigate and prevent those threats
3. determine the set of techniques and tools used to examine and validate proposed solutions
4. end up with a lightweight ontology for security in microservice architectures

The reminder of this paper is structured as follows: section 2 gives a succinct background on used techniques and approaches in this paper, specifically, microservice architectures and systematic mapping elaboration process; section 3 overviews and discusses related works; section 4 details our research methodology; section 5 presents and discusses the mapping results; section 6 presents the proposed ontology for MSA; section 7 discusses threats to validity related to the study and section 8 concludes the paper.

2 Background

2.1 Micorservice Architectures

Microservices is a trending architectural style that aim to design complex systems as collections of fine grained and loosely coupled software artifacts called microservices; each microservice implements a small part or even a single function of the business logic of applications [3]. Their efficient loose coupling enables their development using different programming languages, use different database technologies, and be tested in isolation with respect to the rest of underlying systems. Microservices may communicate with each other directly using an HTTP resource API or indirectly by means of message brokers (see Figure 1). Microservices can either be deployed in virtual machines or lightweight containers. The use of containers for deploying microservices is preferred due to their simplicity, lower cost, and their fast initialization and execution.

Regarding software quality attributes, adopting microservices increases reusability and interoperability, enables scalability and enhances maintainability of complex software systems. Within adequate distributed platforms and technologies, microservices can easily be deployed, replicated, replaced, and destroyed independently without affecting systems availability. Moreover, implementing a single business capability per microservices allows their use in different applications and application domains. The main characteristics that differentiate microservices architectural style from monolithic and its ancestor service-oriented architectures is the smaller size, scalability and independence of each unit constituting a system.

Microservices are getting more attention and becoming adopted in industry. Currently, microservices are used by widely recognized companies such as Coca Cola, Amazon, eBay and NetFlix. Specifically, microservices are becoming more popular in software and IT service companies [5].

[1] https://threatpost.com

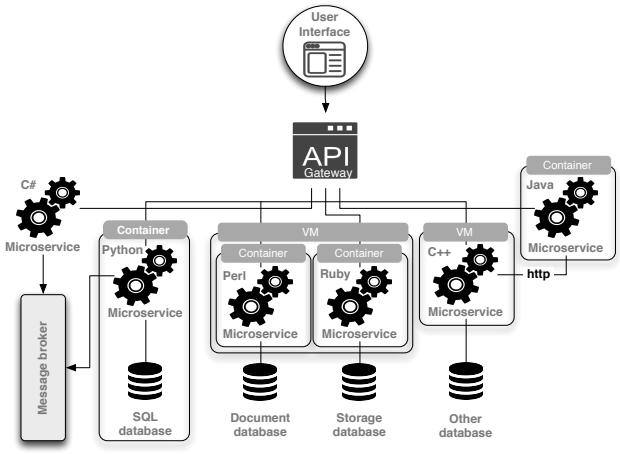

Figure 1: Microservice architecture

Although the advantages brought by adopting microservice architectures in developing complex systems, security is one of the serious challenges that need to be tackled. Thus, there is an urgent need to identify and check current trends in overcoming security challenges in microservice architectures which is the aim of the present paper.

2.2 Systematic mapping

A systematic mapping is a kind of evidence-based software engineering (EBSE) [6]. The aim of a systematic mapping is to provide an overview of a research area by building a classification scheme and structuring evidences on a research field.

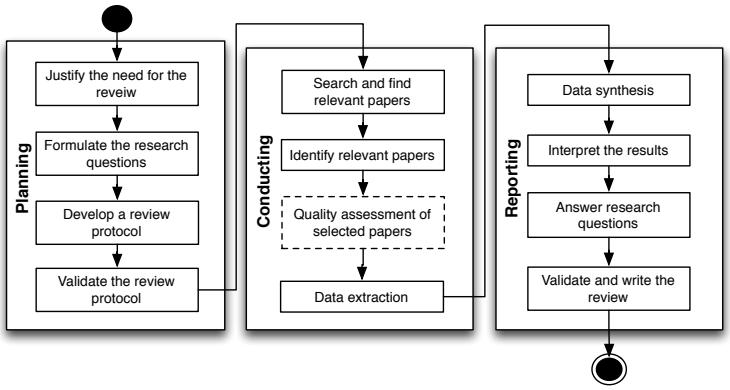

Figure 2: The overall process for elaborating systematic mapping

Peterson et al. [7, 8] has proposed an overall process for the elaboration of systematic mappings. The process is composed of three main steps: *planning*, *conducting* and *reporting*. By planning, one can start by justifying the need and scope of the mapping, formulate the set of research questions, develop and validate a protocol specifying all the decisions relevant to conducting the mapping. The protocol includes the identification of search terms, search strategy, literature sources that need to be used to retrieve relevant papers, how and in which base found papers are selected and included in the mapping, what data need to be extracted from selected papers and how extracted data are synthesized and classified. By conducting, the initially validated protocol from the planning step is executed; thus, the identified sources are used to retrieve papers, found papers are examined for relevance; useful data are then extracted from admitted papers and extracted data are synthesized and classified. By reporting, extracted data from primary papers are visualized, results are interpreted, research questions are answered and the mapping is validated and documented. Figure 2 depicts the overall process for conducting systematic mapping studies as proposed in [8]. The quality assessment step is depicted in Figure 2 within a dashed line box since it is optional as stated in [6, 8].

3 Related Work

We found in the literature several secondary studies (systematic reviews or mappings) dedicated to investigate the state-of-the-art of MSA in general. Surprisingly, few works are found focusing on security aspects in MSA. All found studies, except the work of Vale et al. [9], are either platform or technology dependent investigations.

Vale et al. [9] conducted a similar investigation to our study. They conducted a systematic mapping to reveal adopted security mechanisms for microservice-based systems. The study examined 26 papers published from November 2018 to March 2019. Vale et al. [9] focused only on security mechanisms and categorized the 18 identified mechanisms according to their focus, and classified validation techniques according to their nature. The study revealed that (1) authentication and authorization are the most frequently adopted mechanisms for securing microservices, (2) case studies and experiments are the most validation techniques used for security proposals, (3) absence of patterns for microservices-based systems security. Our study is broader and improves Vale et al. [9] work in several ways. In this study we include published papers since 2011. Moreover, besides security mechanisms, we also focus on identifying security threats and the applicability of proposed solutions regarding their execution platforms and architectural layers.

Yu et al. [10] presented a survey on security in microservice-based fog applications. The survey included papers published between 2010 and 2017. The focus of Yu et al. [10] was domain specific; they focused on determining security challenges and potential solutions of adopting microservices in fog computing. The security issues identified by the study concerns containers, data, and network vulnerabilities. They also proposed a solution for inter-service communication in fog applications.

Monteiro et al. [11] identified a set of elements related to microservices implementation in cloud computing. They reported the same security aspects discussed by Yu et al. [10]. They concluded that availability and trustworthiness are the two major security requirements in MSA.

Nkomo et al. [12] conducted a systematic review on practices that can be incorporated into the development process of microservice-based systems. The focus of Nkomo et al. [12] was to propose general guidelines where security can be tackled earlier when developing microservice-based applications putting much emphasis on microservices composition. They ended up with five security-focused development activities: (1) identify security requirements of microservice composition, (2) adopt secure programming best practices, (3) validate security requirements and secure programming best practices, (4) secure configuration of runtime infrastructure, and (5) continuously monitor the behavior of microservices. Considering security as a primary concern throughout the life-cycle of microservice-based systems is mandatory; however, experiences show that all security threats cannot be identified earlier especially with the continuous evolving of technologies.

Sultan et al. [13] presented a survey on the security of containers; they identified main threats due to images, registries, orchestration, containers themselves, side channels and host OS risks. They distinguished two kinds of solutions to containers security: software-based and hardware-based solutions without further investigation of proposed solutions. Belair et al. [14] complements the work of Sultan et al. [13] by proposing a taxonomy for containers' security proposals. Belair et al. [14] focused on security solutions at the infrastructure level putting much emphasis on data transmission through virtualization. Three categories were identified: configuration-based, code-based and rule-based defense. They reported the fact that Linux security model (LSM), the powerful defense framework targeting Linux, cannot be easily adapted to containers to improve security.

Compared with the works of Yu et al. [10] and Monteiro et al. [11], our study is domain and platform independent and includes more recent endeavors with broader focus on proposed solutions to security threats. Compared with the works in [13] and [14], we focus on our study on security issues concerning MSA in general and not only containers.

4 Research methodology

In this section we present the details of the protocol adopted for conducting this mapping study. Following the guidelines of Peterson et al. [7], a systematic mapping study should include the following primary steps: a definition of research questions, search for relevant papers, screening of found papers, propose or use an existing classification scheme, data extraction and studies mapping. In the sequel, we describe the details of each step.

4.1 Research Questions

The aim of this study is to identify the set of security vulnerabilities and how to be tackled in microservice-based systems. Thus, we formulate our research questions in light of the aims of our study and following the guidelines of Kuhrmann et al. [15]. This study is conducted with five main questions in mind:

RQ1. What are the most addressed security threats, risks, and vulnerabilities of microservices and microservice architectures and how they can be classified? This research question distinguishes the list of mostly treated vulnerabilities from those needing further investigations.

RQ2. What are existing approaches and techniques used for securing mircoservices and microservice architectures and how they can be classified? This research question provides an overview of existing approaches and techniques used for securing microservice-based systems.

RQ3. At what level of architecture the proposed techniques and approaches are applicable for securing microservcies? This research question indicates where security is applied highlighting the less focused levels of microservice architectures .

RQ4. What domains or platforms are the focus of existing solutions for securing microservices and microservice architectures? This research question shows whether the focus of the proposed solutions is platform specific or platform independent.

RQ5. What kind of evidence is given regarding the evaluation and validation of proposed approaches and techniques for securing microservices and microservice architectures? This research question evaluates the maturity of existing security techniques highlighting the set of empirical strategies used to validate proposed solutions.

4.2 Search process

Search string used in this study is designed to be generic and simple. It is constructed based on search terms concerned with *population* and *intervention* as suggested by Petticrew and Roberts in [16]. Population refers to the application area which is microservices and microservice architectures where intervention is security, vulnerabilities and attacks. Accordingly, final adopted search string is :

("microservice" OR "micro-service" OR "micro service")
AND
("architecture" OR "design" OR "system" OR "structure")
AND
("security" OR "vulnerability" OR "attack")

For retrieving relevant studies, we followed the guidelines of Kuhrmann et al. [15]. Thus, we adopted the use of the following online academic libraries:

- IEEE Xplorer (`https://ieeexplore.ieee.org`)
- ACM Digital Library (`https://dl.acm.org`)
- SpringerLink (`https://link.springer.com`)
- ScienceDirect (`https://www.sciencedirect.com/`)
- Wiley Online Library (`https://onlinelibrary.wiley.com`)

To avoid missing relevant studies, we complement our automatic search by conducting recursive backward and forward snowballing on selected studies as suggested by Wohllin [17, 18]. By backward snowballing, we check the relevance of references in approved papers. By forward snowballing, we check the relevance of papers citing approved papers. The snowballing is recursively applied to each newly approved paper. Google Scholar is used as a sole source for forward snowballing.

4.3 Study selection process

The set of retrieved papers by automatic search followed two screening stages. In the first stage, titles and abstracts were read to measure relevance. In the second stage, full texts of papers were examined to check if they meet our inclusion criteria. The list of all the papers are screened separately by the two authors; decisions are exchanged and conflicts are discussed and solved. Found papers from snowballing are also screened separately by the two authors before deciding whether to be included or excluded.

4.4 Inclusion and exclusion criteria

The number of retrieved papers by online academic libraries is reduced by specifying a strict number of inclusion and exclusion criteria. In this study, only peer-reviewed papers from journals and conferences are included. The automatic search is conducted to cover all published papers since 2011 including early publications. The starting year 2011 is adopted since there was no consensus on the term microservice architectures prior 2011 [3]. Only English written papers addressing security aspects or security solutions to microservices or microservice architectures are included. The full list of adopted inclusion and exclusion criteria are presented in Table 1 and Table 2 respectively.

Table 1: Inclusion criteria

ID	Criteria
I1	papers published since 2011 including early publications
I2	papers written in English
I3	papers subject to peer reviews
I4	papers including studies conducted with security aspects of microservices or microservice architectures as their primary topics
I5	papers proposing frameworks, techniques, methods, or tools to secure microservices or microservice architectures
I6	papers presenting qualitative or quantitative evaluation of security techniques used for microservices or microservice architectures

Table 2: Exclusion criteria

ID	Criteria
E1	papers addressing security in distributed platforms and technologies such as clouds without explicit referring to microservices
E2	papers describing general aspects of microservice architectures without putting much emphasis on the security issue
E3	tutorial papers and editorials
E4	papers presenting reviews, surveys or secondary studies on the security of microservices or microservice architectures
E5	books or book chapters, because they usually undergo little peer review and present general ideas already published in journals or conferences
E6	papers without full text available

4.5 Data extraction process

Following the guidelines of Peterson et al. [7] a data extraction form is designed as illustrated in Table 3. Each paper is described in terms of its metadata such as year of publication, source and type. In addition, a set of required information for our analysis are extracted. These include the list of security threats or attacks addressed by the study, proposed solutions, application level of proposed solutions, validation method and application platforms.

Table 3: Data extraction form

ID	Data item	Description	RQ
D1	Study ID	first author name + year	
D2	Year	year of the publication	
D3	Source	source of the publication	
D4	Type	conference or journal paper	
D5	Category	analysis, solution proposal or case study	
D6	Threats	addressed security threats	RQ1
D7	Source of Threats	internal or external	RQ1
D8	Solution type	general protection measures, framework, technique, tool or methodology proposal	RQ2
D9	Security mechanisms	set of security mechanisms proposed or used in the study	RQ2
D10	Applicability level	architectural level where the security mechanism is applied	RQ3
D11	Validation method	verification and validation techniques used to check the feasibility of the proposed solution	RQ5
D12	Domain/Platform	domains or platforms applicable for the proposed solution	RQ4

4.6 Data synthesis

We noticed a lack of a consensus on detailed taxonomies for security threats and security mechanisms; this prevents mapping all the selected studies to appropriate and distinct categories answering research questions RQ1 and RQ2. Moreover, due to the diversity of applications used in selected studies, their targeted platforms and used verification and validation techniques, it was necessary to properly categorize those studies answering RQ3, RQ4 and RQ5.

For mapping properly all the selected studies to proper categories for each research question, we used our experiences and existing taxonomies [11, 1, 19] in identifying categories and their relationships. We also used grounded theory [20] as a complementary approach to generate missing categories from extracted data items. Specifically, we used open coding and selective coding to identify categories and their relationships with existing categories from D6, D9 and D10-11. In this study, grounded theory is used in an iterative process, where categories and subcategories are changed in each iteration until reaching a stability state.

5 Results of the mapping

In this section we describe and detail the results of the mapping study answering the five research questions outlined in section 4.1.

5.1 Overview of selected studies

The search process is conducted in December 2019 and yielded 46 distinct papers published since 2011. The designed query is applied to the set of selected libraries. Table 4 shows the number of returned papers by each library.

Table 4: Number of studies returned by each repository.

Repository	Search results
IEEE Xplorer	50
ACM Digital Library	46
SpringerLink	678
ScienceDirect	255
Wiley Online Library	38
Total	1067

The set of 1067 retrieved papers by the different search engines are gathered and duplicate papers are removed. This reduces the number to 1065. By screening titles and abstracts of remaining papers, 1015 papers are excluded for their irrelevance. After checking the inclusion and exclusion criteria, only 37 papers are approved. By conducting recursive backward and forward snowballing, 9 more papers are added. Two snowballing cycles are performed before reaching a steady state. In the first round, 7 new papers are included; in the second round, 2 more papers are added. Figure 3 depicts the overall selection process.

Figure 4 shows the distribution of selected studies according to their publication year and source. We notice that, although the earlier emergence of MSA in 2011, the interest into securing microservices and microservice architectures is considered few years later and start getting more attentions since 2015. Figure 4 also shows that the maximum number of publications come from IEEE Xplorer and none from Wiley meets our inclusion criteria.

Table 5 shows the complete list of selected studies with their year and type of publication and how they are found.

Following the guidelines of Kuhrmann et al. [15], we also experienced the use of Word Clouds to analyze the appropriateness of our result set of primary studies. Figure 5 illustrates the most frequent words used in selected papers based on their titles and abstracts. The Figure shows that the most used words are microservice, security, application, architecture, system and service. Attacks, vulnerabilities and risks are rarely used in titles and abstracts.

5.2 MSA security threats (RQ1)

Microservice architecture as an emerging development paradigm in software engineering brings new security threats and vulnerabilities. These threats may come from insiders (i.e. internal attacks) or from outsiders (i.e. external attacks). For proper securing microservice-based systems, all threats, regardless of their origin, need to be detected and prevented using either available mitigation techniques or through proposing innovative solutions. In this study, we identify the focus of existing endeavors with respect to the source of threats (internal, external or both). Figure 6 depicts the distribution of identified and selected studies regarding the addressed source of threats. Figure 6 shows that 63% of primary studies focus on external attacks, only 13% focus on internal attacks and 24% focus on both source of threats. This clearly indicates an unbalanced research focus towards external attacks.

Due to the plethora of taxonomies of security threats and lack of a consensus among their categorization, we adopted, a classification based on targets of attacks. Accordingly, threats in MSA can be classified into:

- *User-based attacks:* attacks where users are involved directly (i.e. malicious user actions) or indirectly (i.e. inadvertent insider actions).

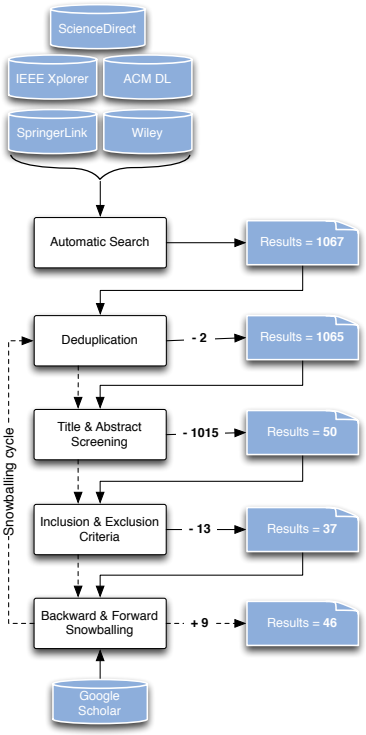

Figure 3: Paper selection process

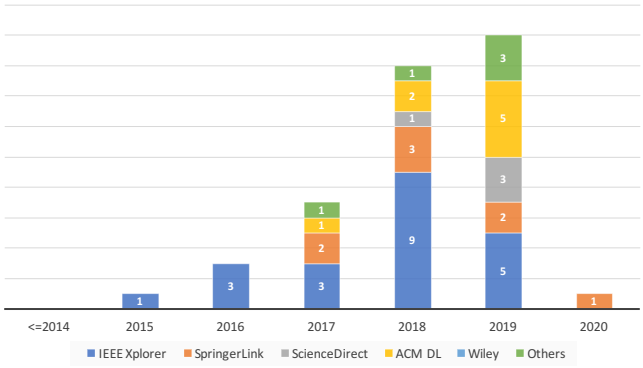

Figure 4: Distribution of selected studies by year and digital library

- *Data attacks:* threats targeting sensitive data that can be disclosed and manipulated by attackers.
- *Infrastructure attacks:* attacks targeting MSA architectural elements and platforms such as monitors, discovery service, message broker, load balancer, etc.
- *Software attacks:* threats involving code transformation or injection for malicious purposes.

Table 6 shows the set of MSA security threats addressed by primary studies grouped by category. The results revealed that unauthorized access, sensitive data exposure and compromising individual microservices are the most treated and addressed threats by contemporary studies. In addition, infrastructure attacks are the most diverse but with less addressed attacks in selected studies.

Although IBM X-Force [66] reported that 60% of all attacks were carried out by insiders, the study shows that only 13% of primary studies focus on internal attacks. This is probably due to the fact that external threats are easier to be handled compared with internal threats. External threats are common in networking systems and can usually be identified and prevented by means of strong firewalls and intrusion detection systems; internal threats often requires

Table 5: List of selected studies: C: Conference paper, J: Journal paper, A: Automatic search, S: Snowballing

ID	Cite	Year	Type	Publisher	Search type
p1	[12]	2019	C	Springer	A
p2	[21]	2018	C	Springer	A
p3	[22]	2020	C	Springer	A
p4	[23]	2017	C	Springer	A
p5	[24]	2017	C	Springer	A
p6	[25]	2018	C	Springer	A
p7	[26]	2019	C	Springer	A
p8	[27]	2018	C	IEEE	A
p9	[28]	2018	C	IEEE	A
p10	[29]	2018	C	IEEE	A
p11	[30]	2016	C	IEEE	A
p12	[31]	2015	C	IEEE	A
p13	[32]	2018	C	IEEE	A
p14	[33]	2017	C	IEEE	A
p15	[34]	2017	C	IEEE	A
p16	[35]	2016	C	IEEE	A
p17	[1]	2018	C	IEEE	A
p18	[36]	2018	C	IEEE	A
p19	[37]	2019	J	IEEE	A
p20	[38]	2019	C	IEEE	A
p21	[39]	2019	C	IEEE	A
p22	[40]	2018	C	IEEE	A
p23	[41]	2018	C	IEEE	A
p24	[42]	2019	J	IEEE	A
p25	[43]	2017	C	IEEE	S
p26	[44]	2018	C	IEEE	S
p27	[45]	2019	J	IEEE	S
p28	[46]	2016	J	IEEE	S
p29	[47]	2019	J	JSW	S
p30	[48]	2017	J	IOP	S
p31	[49]	2019	C	CITRENZ	S
p32	[50]	2018	J	IJERT	S
p33	[51]	2019	J	IASKS	S
p34	[52]	2018	C	Springer	A
p35	[53]	2019	C	ACM	A
p36	[54]	2017	C	ACM	A
p37	[55]	2018	C	ACM	A
p38	[57]	2018	C	ACM	A
p39	[58]	2019	C	ACM	A
p40	[59]	2019	C	ACM	A
p41	[60]	2019	C	ACM	A
p42	[61]	2019	C	ACM	A
p43	[62]	2019	J	Science Direct	A
p44	[63]	2018	J	Science Direct	A
p45	[64]	2019	J	Science Direct	A
p46	[65]	2019	J	Science Direct	A

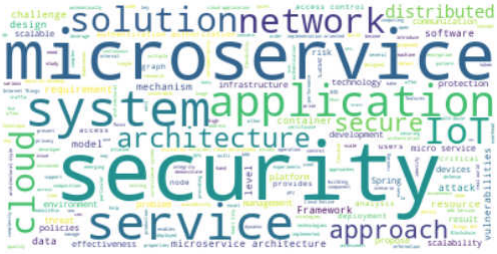

Figure 5: Keywords cloud of primary studies

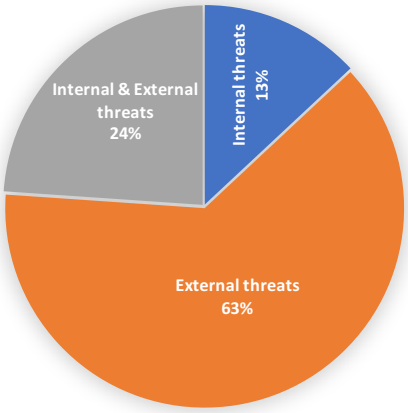

Figure 6: MSA security source of threats

considerable policy changes and continuous monitoring of internal traffics. This is owing to privileges awarded and sensitive data exposed to insiders.

The diversity of attacks is due to the adoption of Zero Trust model [67] that suggests to afford no default trust to users, devices, applications, or packets; instead every action and entity need to be authenticated and authorized appropriately. Moreover, infrastructure attacks are less addressed due to their complexity since most attacks require low level solutions especially those related to hardware, nodes and operating systems. Attacks from other categories often require high level or software-based solutions that can easily be integrated into existing platforms and technologies. This justifies why software, user-based, and data attacks earned more attention than infrastructure attacks. Thus, we advocate for research studies that investigate threats caused by insiders in microservice-based applications. In addition, we suggest to investigate all OWASP identified vulnerabilities with their effects when adopting microservice architectures.

5.3 Microservice security mechanisms (RQ2)

Due to the diversity of proposed solutions, we classify MSA security mechanisms addressed in primary studies regarding the nature of their proposals as follows:

- *General protection measures:* use of general security techniques to mitigate common known threats in MSA, or a set of general guidelines on choosing appropriate languages and technologies.
- *Framework-based solutions:* architectural frameworks for MSA incorporating specific modules to handle some security aspects and mechanisms such as authorization, continuous monitoring, diagnosis.
- *Technique-based solutions:* newly designed or adopted techniques from other domains to mitigate or prevent some security threats in MSA.
- *Tool-based solutions:* newly developed tools implementing security measures.
- *Algorithm-based solutions:* new algorithms conceived for the detection or prevention of security threats.
- *Protocol-based solutions:* new protocols conceived for the protection of communications among the different MSA architectural elements.
- *Analysis:* experimentation, comparison or discussion of existing security mechanisms of MSA.

Our investigation (see Figure 7) shows that 33% of the studies proposed new techniques for securing MSA and 31% proposed framework-based solutions and 13% proposed general protection measures. Few studies developed new tools, algorithms or protocols. Specifically, the authors of P17 have analyzed existing security mechanisms and proposed a framework based on the insights of the conducted analysis.

Proposed solutions for securing microservices and microservice architectures can be classified into proposals for enforcing Access Control (i.e. authentication and/or authorization policies), auditing, mitigation and prevention:

- *Access Control - Authentication:* techniques used to verify the identity of users requiring access to MSA resources and data.

Table 6: MSA security threats per category

Threats	Percentile	Studies
User-based Attacks	**58.70%**	
Brute Force Attack	4.35%	[47, 49]
Cross-Site Request Forgery (CSRF)	13.00%	[26, 47, 25, 49, 52, 54]
Spoofing	4.35%	[25, 44]
Malicious Insider	4.35%	[42, 26]
Unauthorized Access	50.00%	[12, 22, 26, 27, 28, 30, 32, 33, 34, 25, 40, 43, 44, 47, 48, 49, 50, 55, 57, 58, 62, 63, 64]
Violate non-repudiation	2.17%	[21]
Data Attacks	**47.83%**	
Eavesdropping	2.17%	[62]
Heartblead	4.35%	[25, 44]
Man in The Middle attack (MiTM)	6.52%	[34, 25, 59]
Padding Oracle On Downgraded Legacy Encryption attack (POODLE)	2.17%	[25]
Replay attack	6.52%	[34, 51, 63]
Sensitive data exposure	30.43%	[21, 23, 28, 32, 33, 25, 38, 42, 43, 46, 58, 62, 64, 65]
Sniffing attack	6.52%	[22, 26, 25]
Infrastructure Attacks	**41.30%**	
Compromise containers	15.21%	[1, 34, 37, 52, 53, 54, 60]
Compromise virtual machines	8.70%	[1, 31, 52, 54]
Compromise discovery service	4.35%	[12, 25]
Compromise hypervisor	8.70%	[23, 1, 25, 46]
Compromise management interface	2.17%	[42]
Compromise network nodes	2.17%	[1]
Compromise operating systems	4.35%	[23, 46]
Downgrade attack	4.35%	[21, 55]
Hardware backdoors	2.17%	[25]
Malicious images	4.35%	[25, 60]
Malicious provider	2.17%	[25]
Misconfiguration	4.35%	[25, 40]
Port scan attack	2.17%	[22]
Sandbox escape	2.17%	[25]
Session hijacking	4.35%	[26, 25]
Stress attack	4.35%	[41, 42]
Cold boot attack	2.17%	[23]
Software Attacks	**50.00%**	
Code reuse attack	4.35%	[36, 59]
Compromise microservices	32.61%	[24, 1, 29, 31, 25, 37, 39, 40, 44, 51, 52, 54, 60, 63, 65]
Disrupt sensitive operation	2.17%	[21]
Denial of Service (DoS)	17.39%	[34, 25, 47, 49, 52, 54, 61]
Injection	15.22%	[26, 1, 47, 49, 52, 54, 61]

- *Access Control - Authorization:* techniques used to check users' permissions for accessing specific MSA resources or data.
- *Auditing:* techniques applied at runtime for discovering security gaps and may: (1) subsequently initiate appropriate measures or (2) simply report security breaches to relevant supervisory authority.
- *Mitigation:* techniques that limit the damage of attacks when they appear. Mitigation techniques can be integrated into existing mircoservice-based systems.
- *Prevention:* techniques that try to stop attacks from happening in the first place. Prevention techniques need to be considered when developing new mircoservice-based systems.

Table 7 shows the list of proposed solutions mapped into our classification with the proportion rate for each proposal with respect to the total set of primary studies. The results show that much emphasis is put on proposing auditing techniques (43.48%), enforcing authentication and/or authorization (39.13%[2]), and prevention (34.78%), where less attention is being paid to mitigation (10.87%).

The much emphasis put on authentication and authorization techniques is defensible. In fact, authentication and authorization are basic security mechanisms to any secure system. They form a front defense line in the protection of the different microservice architecture elements (i.e. individual microservices, API gateway, containers, microservice registry, etc.). However, studies considering authentication and authorization are less innovative since they propose combination of existing techniques and standards. For example, the authors of P8 proposed a combination of OAuth 2.0, JWT, Open ID and SSO used by a special authentication and authorization orchestrator.

[2]This value is calculated by collecting all the papers from authorization and/or authentication categories and removing duplicates; the obtained number is divided by the total number of papers.

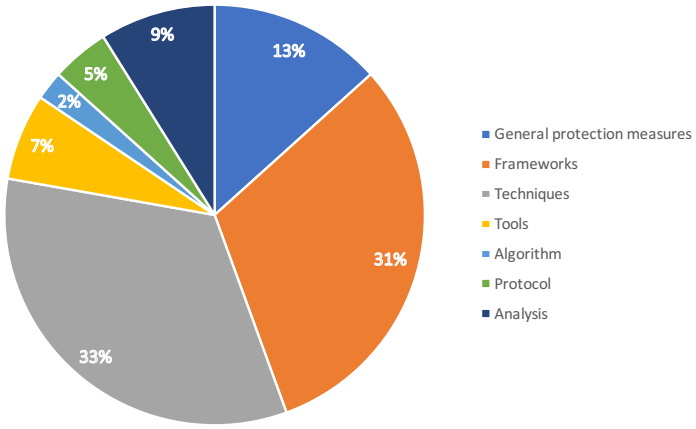

Figure 7: MSA security solutions

Besides general continuous monitoring and code analysis, audition proposals are showing the integration of artificial intelligence techniques such as machine learning and self-learning algorithms (P10, P22, P35). Those techniques are based on runtime analysis of user and/or microservice behaviors and (semi-)automatically take predefined actions in reaction to suspicious behaviors.

Most, if not all, proposals for mitigation are Moving Target Defense-based solutions (MTD) [68]. The idea behind MTD is to continuously perform transformation of system components and configurations preventing attackers from acquiring knowledge about target systems to be used to initiate harmful attacks. This includes, periodically update or restart microservices, IP shuffling, and live migration of microservices. Specifically, the authors of P21 proposed deception through live cloning and sandboxing of suspicious containers respecting the same network overloading and performance to deceive attackers .

Prevention proposals are the most diverse techniques. They varies from using physical computing devices such as Hardware Security Module (HMS), powerful techniques and technologies such as encryption and Blockchain into adopting software design decisions such as using secure programming languages and smart contracts.

Due to the lower rate of mitigation techniques and their applicability to existing microservice-based systems, we advocate more research studies on mitigation techniques.

5.3.1 Micorservice security application levels (RQ3)

The adoption of MSA as an architectural design of distributed applications introduces security vulnerabilities in different architectural layers. Thus, security measures need to be taken in every layer of MSA. In this study, we distinguish the following layers:

- *Microservice:* Individual microservices are the mainstays of MSA, those micorservices can be blocked or compromized through injection of malicious code. Thus, security measures to adopt only trusted micorservices and protect them from internal and external attacks need to be taken.
- *Composition:* Connections among microservices can be broken. Moreover, compromising a single microservice may affect the security of the whole system due to the insecure configuration options for individual microservices, their locations and inter-connections. Several security measures need to be taken at this level to secure the overall architecture of microservice-based systems.
- *API:* Finely tuned attacks on APIs can bypass traditional security measures provided by API gateways. Hence, assets can be accessed and controlled by malicious users. Appropriate security measures should be taken at API gateways to avoid such vulnerabilities.
- *Communication:* Data exchanged between microservices through event-buses can be intercepted and altered by malicious insiders. Thus, securing communication channels between microservices is mandatory for securing microservice-based systems.

Table 7: MSA security mechanism per category

Security Mechanisms	Percentile	Studies
Authentication	**28.26%**	
Centralized Access Control Manager	6.52%	[26, 30, 34]
Certificates	6.52%	[12, 25, 44]
Open ID	6.52%	[27, 45, 57]
Single Sign On (SSO)	4.35%	[27, 48]
White-list HTTP/IP	4.35%	[12, 50]
HIP exchange protocol	2.17%	[34]
J-PAKE protocol	4.35%	[51, 63]
Distribute sessions	2.17%	[48]
HTTP signatures	2.17%	[50]
Authorization	**21.74%**	
Attribute Based Access Control (ABAC)	2.17%	[30]
Role Based Access Control (RBAC)	6.52%	[26, 51, 57]
R/W Permission to message broker	2.17%	[12]
OAuth 2	17.39%	[26, 27, 30, 45, 47, 49, 50, 57]
Authentication & Authorization	**23.91%**	
JSON Web Token (JWT)	17.39%	[12, 27, 25, 33, 45, 48, 50, 57]
Firewalls	8.70%	[22, 29, 30, 45]
Auditing	**43.48%**	
Continuous monitoring	34.78%	[21, 22, 24, 1, 29, 31, 39, 40, 41, 45, 52, 53, 54, 58, 60, 64]
Scan container images	2.17%	[12]
Static/Dynamic code analysis	8.70%	[25, 46, 58, 65]
Machine learning	6.52%	[29, 40, 53]
Intrusion detection	4.35%	[1, 45]
Mitigation	**10.87%**	
Roll-back/Restart microservices	2.17%	[1]
Scale up/down N-variant microservices	2.17%	[1]
Short-lived tokens	2.17%	[26]
Diversification	4.35%	[1, 36]
IP shuffling	2.17%	[37]
Live migration	4.35%	[37, 39]
Deception	2.17%	[39]
Isolation of suspicious microservices	2.17%	[1]
Prevention	**34.78%**	
Blockchain technology	4.35%	[28, 32]
Encryption	10.87%	[21, 34, 1, 43, 62]
Hardware Security Module (HMS)	2.17%	[25]
Least privilege	2.17%	[25]
No shared memory access	2.17%	[25]
Proper design	6.52%	[35, 38, 59]
Secure languages	4.35%	[25, 55]
Smart contracts	6.52%	[28, 30, 32]
TLS protocol	6.52%	[23, 33, 25]
SGX technology with enclaves	6.52%	[23, 25, 46]

- *Deployment:* Containers holding micorservices can also be sources of vulnerabilities. Containers can be compromised through gaining an unauthorized access or deriving vulnerabilities from using images from untrusted sources. Thus, appropriate security measures should also be taken at this level.
- *Soft-Infrastructure:* Infrastructure vulnerabilities are lower level vulnerabilities that can affect practically every software entity running on the network including monitors, registries, message brokers, load balancers and other orchestrators. Thus, introducing techniques at this level to guarantee the security of the diverse software network entities and the safety of their configuration is of higher importance.
- *Hard-Infrastructure:* Hardware components are also vulnerable to attacks. Attackers may use bugs and backdoors intentionally or unintentionally introduced at manufacturing [69] to initiate attacks. These vulnerabilities need to be tackled by introducing appropriate error and backdoor detection mechanisms.

Table 8 shows the distribution of solutions provided by primary studies per their application layers. The study revealed that much emphasis are put to conceive solutions applicable at soft-infrastructure and API gateways where less attention is being paid to composition and hard-infrastructure layers.

Although, microservices are the mainstay of MSA, securing individual microservices is not getting a higher rate. The less interest in hard-infrastructure solutions is defensible due to their complexity and cost-intensive compared with soft-infrastructure based solutions. However, individual microservices, their composition and communication should have much attention than that has been revealed. Specifically, communication protection is of a high importance regarding the huge number and nature of transmitted data in the communication channels.

Table 8: MSA security application levels

Application Layer	Percentile	Studies
Microservice	20.00%	[24, 1, 36, 43, 44, 45, 58, 61, 65]
Composition	6.67%	[35, 38, 45]
API	35.56%	[12, 21, 26, 27, 28, 30, 32, 33, 34, 25, 43, 47, 48, 49, 50, 57]
Communication	22.22%	[23, 27, 28, 32, 33, 34, 25, 51, 57, 63]
Deployment	13.33%	[23, 37, 42, 43, 46, 64]
Soft-Infrastructure	68.89%	[12, 21, 22, 24, 1, 26, 27, 28, 29, 30, 31, 33, 34, 25, 36, 39, 40, 41, 44, 45, 46, 48, 52, 53, 54, 55, 57, 58, 60, 62, 65]
Hard-Infrastructure	8.89%	[23, 37, 39, 46]

5.3.2 MSA security mechanisms target platforms (RQ4)

The identified papers in this study are classified regarding the target platforms and applications for their proposed solutions. Table 9 shows that 34.78% of papers proposed MSA security solutions that work for different platforms; closer proportion is found for solutions to deal with securing microservices in the cloud platform. Few studies proposed platform specific solutions such as 5G platform, IoT, Web applications, kubernetes platforms and Springer framework.

Table 9: MSA security application platforms

Applications/ Platforms	Percentile	Studies
IoT applications	13.04%	[29, 33, 40, 43, 44, 55]
Cloud platforms	28.26%	[23, 27, 30, 31, 34, 37, 41, 42, 46, 52, 54, 62, 65]
Osmotic computing	2.17%	[32]
Container-based platforms	10.87%	[36, 37, 39, 53, 60]
Web applications	6.52%	[41, 61, 64]
Springer platform	4.35%	[47, 49]
Kubernetes platform	2.17%	[22]
5G platform	2.17%	[57]
Independent	34.78%	[12, 21, 24, 1, 26, 28, 35, 25, 38, 45, 48, 50, 51, 58, 59, 63]

Cloud-focused and platform independent solutions are found within a higher rates, 34.78% and 28.26%, respectively. The interest to cloud computing is understandable due to different facilities provided to companies by adopting MSA for developing their applications. Adopting MSA for developing applications in the cloud allow companies to integrate existing legacy systems, to grow with demands and to use up-to-date and intuitive interfaces. Solutions provided for IoT applications are also getting more attentions due to the specificity and the growing needs to those applications in the market.

5.3.3 Micorservice security V&V methods (RQ5)

For validating the proposed solutions, we distinguished the use of several verification and validation approaches:

- *Validation by simulation:* this includes: (1) use of simulated lab environments or testbeds for testing proposed designs, (2) simulation of attacks with check bypassing of proposed security measures.
- *Manual testing:* this includes: (1) use case-based testing, (2) emulate attacks, and (3) reconfigure-testing cycles.
- *Performance analysis:* this is performed by measuring overheads, latency, throughput, memory storage, CPU usage, response time, and traffic measurement.
- *Qualitative analysis:* compare or verify and validate a set of qualitative requirements. These include: causing single point to failure, complexity of cracking, complexity of implementation and potential inherent bottleneck.
- *Quantitative analysis:* comparing the proposed solution with similar proposals using quantitative metrics such as session sustainability, and popular machine learning metrics.
- *Adhoc metrics:* proposing specific metrics for the evaluation of proposals. For example, P18 proposed a diversification index as a security measure to validate the proposal. Authors of P21 used a quality of deception metric to evaluate the proposed deception mechanism.
- *Case study based validation:* use case studies to validate the feasibility of the proposed solution.
- *Proof of concept (POC):* develop a prototype to demonstrate the feasibility of the proposed solution.
- *Tool-based testing:* use unit-based testing tools such as IntelliJ IDEA[3].

[3]IntelliJ IDEA: `https://www.jetbrains.com/fr-fr/idea/`

- *Formal verification:* use model checkers or theorem provers to check the validity of specified properties.
- *Complexity measuring:* estimate temporal complexity of proposed algorithms implementing solutions.
- *Methodology based analysis:* use predefined and well-known methodologies such as OWASP risk rating, attack surface or security risk comparison.

Table 10: MSA security verification and validation methods

V&V methods	Percentile	Studies
Simulation	6.52%	[22, 33, 44]
Manual Testing	8.70%	[39, 41, 49, 64]
Performance analysis	39.13%	[23, 1, 26, 29, 31, 33, 34, 25, 37, 39, 40, 41, 44, 46, 54, 55, 62, 65]
Qualitative analysis	10.87%	[24, 27, 48, 50, 62]
Quantitative analysis	8.70%	[34, 65, 40, 53]
Adhoc metrics	6.52%	[36, 39, 40]
Case study based validation	26.09%	[21, 28, 30, 33, 35, 43, 51, 58, 59, 60, 61, 63]
Proof of concept (POC)	2.17%	[47]
Tool-based testing	10.87%	[49, 52, 57, 59, 65]
Formal verification	2.17%	[38]
Complexity measuring	6.52%	[1, 39, 42]
Methodology-based validation	4.35%	[36, 42]

Table 10 shows that performance analysis and case study based validation are the most adopted techniques for verification and validation. Formal verification, POC and methodology-based analysis are the least used methods for validation. This is due to the nature of proposed solutions. Formal verification can only be adopted when formal specification of systems and their properties are described (only P20), where complexity measures are adopted for algorithmic based solutions (only P6). Validation by simulation and adhoc metrics are found used in equal measure with a rate equal to 6.52%. Most used simulation methods used simulated lab environments or testbeds. Only P3 used simulated attacks to evaluate the proposed technique. Note that most examined studies used more than one validation techniques and 2 studies (P17 and P27) proposed solutions without using any mentioned validation technique. Instead, they discussed the details of proposed solutions and claimed that they are sufficient enough to mitigate the addressed security threat(s).

6 An ontology for securing MSA

Making the results of our study practical and extendable and due to the static nature of taxonomies, we propose an ontology-based representation of our results. Within an ontology, one can describe relationships among ontology concepts and individuals. In our study, relationships between security threats and security mechanisms are mandatory. In addition, mapping security mechanisms to their applicability architectural levels and platforms is necessary. Figure 8 describes the overall MSA ontology retrieved from this study. The ontology is developed using Protégé [4] and its consistency and coherence are checked using the Protégé Debugger option. The main concepts of the ontology are:

1. *SecurityMechnism*: describes the set of security mechanisms proposed for MSA that have been retrieved by this study.
2. *SecurityThreat*: describes the set of security threats threatening microservices and microservice architecture that have been retrieved by this study.
3. *ArchitecturalLayer*: describes the different architectural layers of MSA.
4. *TargetPlatform*: describes the set of platforms addressed by the security mechanisms retrieved by this study. For platform independent solutions, an individual named *Independent* is added to this concept class.
5. *SolutionType*: describes the solution type for each proposed mechanism.
6. *ThreatSource*: describes the source of each threat: *internal* or *external*.
7. *Security_V&V_Technique*: describes the verification and validation techniques used for recognized security mechanisms.

Relationships between classes are reflected through Protégé object properties presented in Table 11.

For the usability of the ontology, OWL-DL queries can be used to investigate the ontology structure. Table 12 describes some of useful queries described in OWL-DL. In Table 12, Q1 can be used to retrieve the list of security

[4] https://protege.stanford.edu/

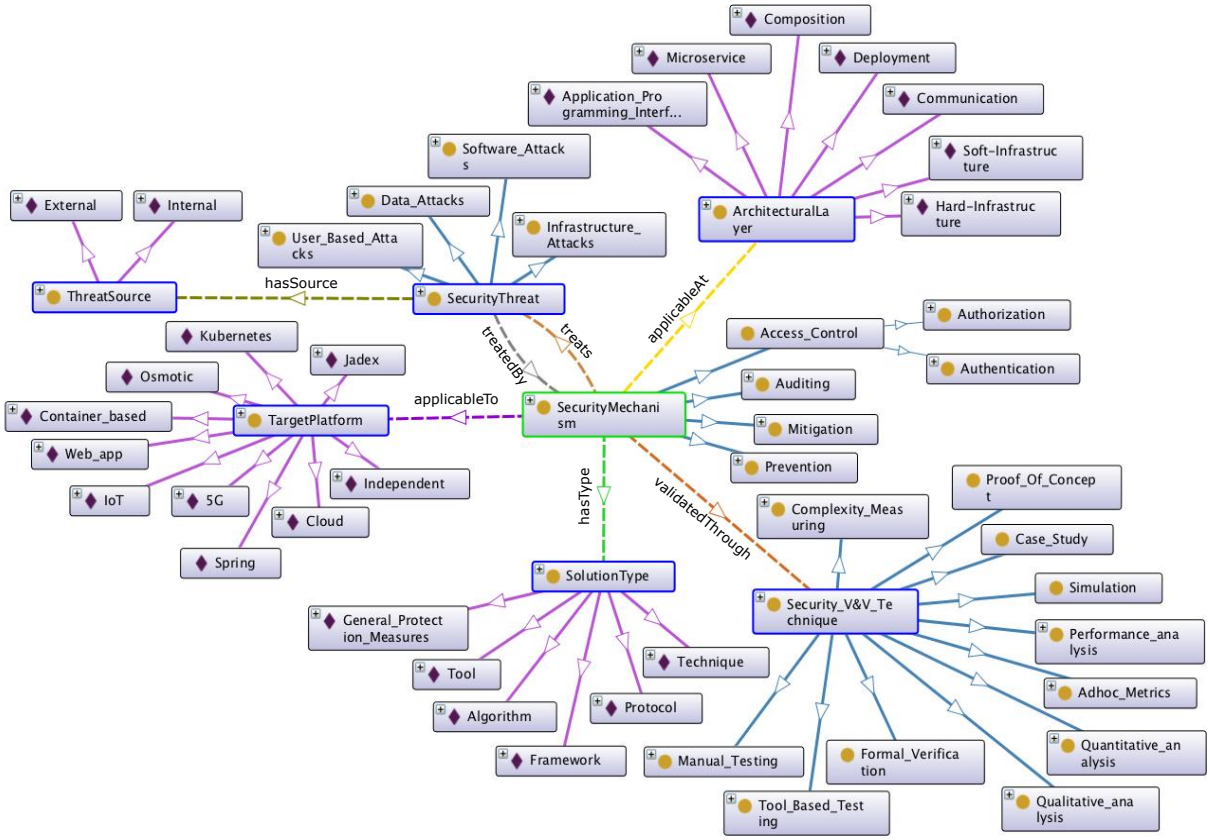

Figure 8: Ontology for MSA security

Table 11: Relationships between ontology classes

Object Properties	Domain	Rang
applicableAt	SecurityMechanism	ArchitecturalLayer
applicableTo	SecurityMechanism	TargetPlatform
hasSource	SecurityThreat	ThreatSource
hasType	SecurityMechanism	SolutionType
treatedBy	SecurityThreat	SecurityMechanism
treats	SecurityMechanism	SecurityThreat
validatedThrough	SecurityMechanism	Security_V&V_Technique

mechanisms applied to deal with *Unauthorized_access* threat. Q2 returns the list of security threats that can be alleviated by continuous monitoring. Q3 returns the list of security mechanisms applied at individual microservices. Finally, Q4 returns the list of internal threats recognized in this study. The overall ontology is available at `https://github.com/hannousse/MSASecurity` in an OWL format.

Table 12: OWL-DL Query samples

ID	Query
Q1	SecurityMechanism **and** (treats **value** Unauthorized_Access)
Q2	SecurityThreat **and** (treatedBy **value** Continuous_Monitoring)
Q3	SecurityMechanism **and** (applicableAt **value** Micorservice)
Q4	SecurityThreat **and** (hasSource **value** Internal)

7 Threats to validity

In this section we discuss the threats to the validity and how we mitigated their effects on the obtained results.

An internal validity threat to our study concerns the identification of primary studies from the large set of papers found in the literature. For these sake, we adopted the guidelines of Kuhrmann et al. [15] for the selection of search engines. To void bias to search engines, we completed our search by snowballing technique [18] over already identified papers. The use of several iterations of the snowballing technique allowed the identification of nine more relevant papers in which five were not indexed by the selected search engines. For ensuring the inclusion of high quality papers, we adopted a set of strict inclusion and exclusion criteria that accept only peer-reviewed journal and conference papers for their completeness and sufficient results. However, since only papers explicitly referring to microservices or microservice architectures were included, some papers focusing on securing the deployment layer, specifically Docker containers [70] were omitted.

A conclusion validity threat to our study concerns the adoption of taxonomies for security threats and mechanisms. In fact, several taxonomies are investigated [11, 1, 19], however, none of those taxonomies enable the proper classification of all the identified studies. Thus, we used open and selective coding from grounded theory [20] and we adopted a classification based on deeper analysis of the focus and the proposed solutions of identified papers. Some of the categories of our classifications are already used in existing taxonomies, some they are either used as they are or adapted to fulfill the context of our study.

8 Conclusion

In this study, we conducted a systematic mapping on securing microservices focusing on threats, nature, applicability platforms, and validation techniques of security proposals. The study examined 46 papers published since 2011. The results revealed that unauthorized access, sensitive data exposure and compromising individual microservices are the most treated and addressed threats by contemporary studies. The results also revealed that auditing, enforcing access control, and prevention based solutions are the most proposed security mechanisms. Additionally, we found that most proposed solutions are applicable at soft-infrastructure layer of MSA. Our study shows that 34.78% of papers proposed MSA security solutions that work for different platforms, the same proportion is noticed for cloud-based solutions. Finally, we found that most verification and validation methods were based on performance analysis, and case studies. We also proposed and made available of an ontology summarizing and gathering the retrieved results. the proposed ontology can be used as a guide to developers about already recognized threats and security mechanisms for MSA. We noticed that most addressed threats are well-known for other architectural styles and few are concerned directly with MSA. Specifically, compromising individual microservices that can radically lead to a chain defection in MSA. Moreover, continuous monitoring became very popular among MSA designers to prevent possibly future threats. Encryption remain the most used technique facing sensitive data exposure. Regarding noticed unbalanced research focus on external attacks and prevention techniques, we advocate more studies studying internal attacks and proposing mitigation techniques. Moreover, more studies are suggested for treating individual microservice and communication layers vulnerabilities.

References

[1] T. Yarygina and A. H. Bagge. Overcoming security challenges in microservice architectures. In *2018 IEEE Symposium on Service-Oriented System Engineering (SOSE)*, pages 11–20, March 2018.

[2] Saša Baškarada, Vivian Nguyen, and Andy Koronios. Architecting microservices: Practical opportunities and challenges. *Journal of Computer Information Systems*, pages 1–9, 2018.

[3] Nicola Dragoni, Saverio Giallorenzo, Alberto Lluch Lafuente, Manuel Mazzara, Fabrizio Montesi, Ruslan Mustafin, and Larisa Safina. *Microservices: Yesterday, Today, and Tomorrow*, chapter 12, pages 195–216. Springer International Publishing, Cham, 2017.

[4] Nuha Alshuqayran, Nour Ali, and Roger Evans. A systematic mapping study in microservice architecture. In *2016 IEEE 9th International Conference on Service-Oriented Computing and Applications (SOCA)*, pages 44–51. IEEE, 2016.

[5] J. Bogner, J. Fritzsch, S. Wagner, and A. Zimmermann. Microservices in industry: Insights into technologies, characteristics, and software quality. In *2019 IEEE International Conference on Software Architecture Companion (ICSA-C)*, pages 187–195, 2019.

[6] Barbara Ann Kitchenham, David Budgen, and Pearl Brereton. *Evidence-Based Software Engineering and Systematic Reviews*. Chapman & Hall/CRC, 2015.

[7] Kai Petersen, Robert Feldt, Shahid Mujtaba, and Michael Mattsson. Systematic mapping studies in software engineering. In *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*, EASE'08, pages 68–77, Swindon, UK, 2008.

[8] Kai Petersen, Sairam Vakkalanka, and Ludwik Kuzniarz. Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, 64:1–18, 2015.

[9] Anelis Pereira Vale, Gastón Márquez, Hernán Astudillo, and Eduardo B Fernandez. Security mechanisms used in microservices-based systems: A systematic mapping. In *XLV Latin American Computing Conference*, pages 1–10, 2019.

[10] Dongjin Yu, Yike Jin, Yuqun Zhang, and Xi Zheng. A survey on security issues in services communication of microservices-enabled fog applications. *Concurrency and Computation: Practice and Experience*, 31(22):e4436, 2019. e4436 cpe.4436.

[11] Luciano de Aguiar Monteiro, Washington Henrique Carvalho Almeida, Raphael Rodrigues Hazin, Anderson Cavalcanti de Lima, Sahra Karolina Gomes e Silva, and Felipe Silva Ferraz. A survey on microservice security–trends in architecture, privacy and standardization on cloud computing environments. *International Journal on Advances in Security*, 11(3-4):201–213, 2018.

[12] Peter Nkomo and Marijke Coetzee. Software development activities for secure microservices. In Sanjay Misra, Osvaldo Gervasi, Beniamino Murgante, Elena Stankova, Vladimir Korkhov, Carmelo Torre, Ana Maria A.C. Rocha, David Taniar, Bernady O. Apduhan, and Eufemia Tarantino, editors, *Poceedings of International Conference on Computational Science and Its Applications, ICCSA 2019*, pages 573–585, Cham, 2019. Springer International Publishing.

[13] S. Sultan, I. Ahmad, and T. Dimitriou. Container security: Issues, challenges, and the road ahead. *IEEE Access*, 7:52976–52996, 2019.

[14] Maxime Bélair, Sylvie Laniepce, and Jean-Marc Menaud. Leveraging kernel security mechanisms to improve container security: A survey. In *Proceedings of the 14th International Conference on Availability, Reliability and Security*, ARES '19, pages 76:1–76:6, New York, NY, USA, 2019. ACM.

[15] Marco Kuhrmann, Daniel Méndez Fernández, and Maya Daneva. On the pragmatic design of literature studies in software engineering: an experience-based guideline. *Empirical Software Engineering*, 22(6):2852–2891, 2017.

[16] Mark Petticrew and Helen Roberts. *Systematic Reviews in the Social Sciences: A Practical Guide*. John Wiley & Sons, Ltd, 2006.

[17] Claes Wohlin. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, EASE '14, pages 1–10, New York, NY, USA, 2014. Association for Computing Machinery.

[18] Claes Wohlin. Second-generation systematic literature studies using snowballing. In *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering*, EASE '16, pages 1–6, New York, NY, USA, 2016. Association for Computing Machinery.

[19] OWASP. Owasp top 10: The ten most critical web application security risks. Technical report, OWASP Foundation, 2017.

[20] Anselm L. Strauss and Juliet M. Corbin. *Basics of qualitative research: techniques and procedures for developing grounded theory*. Sage Publications, Thousand Oaks, Calif, 1998.

[21] Mohsen Ahmadvand, Alexander Pretschner, Keith Ball, and Daniel Eyring. Integrity protection against insiders in microservice-based infrastructures: From threats to a security framework. In Manuel Mazzara, Iulian Ober, and Gwen Salaün, editors, *Proceedings of Federation of International Conferences on Software Technologies: Applications and Foundations*, pages 573–588, Cham, 2018. Springer International Publishing.

[22] Nico Surantha and Felix Ivan. Secure kubernetes networking design based on zero trust model: A case study of financial service enterprise in indonesia. In Leonard Barolli, Fatos Xhafa, and Omar K. Hussain, editors, *Proceedings of International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pages 348–361, Cham, 2020. Springer International Publishing.

[23] Stefan Brenner, Tobias Hundt, Giovanni Mazzeo, and Rüdiger Kapitza. Secure cloud micro services using intel sgx. In Lydia Y. Chen and Hans P. Reiser, editors, *Proceedings of IFIP International Conference on Distributed Applications and Interoperable Systems*, pages 177–191, Cham, 2017. Springer International Publishing.

[24] Christian Otterstad and Tetiana Yarygina. Low-level exploitation mitigation by diverse microservices. In Flavio De Paoli, Stefan Schulte, and Einar Broch Johnsen, editors, *Proceedings of the 6th IFIP WG 2.14 European Conference on Service-Oriented and Cloud Computing*, pages 49–56, Cham, 2017. Springer International Publishing.

[25] Tetiana Yarygina and Christian Otterstad. A game of microservices: Automated intrusion response. In Silvia Bonomi and Etienne Rivière, editors, *Proceedings of IFIP International Conference on Distributed Applications and Interoperable Systems*, pages 169–177, Cham, 2018. Springer International Publishing.

[26] Antonio Nehme, Vitor Jesus, Khaled Mahbub, and Ali Abdallah. Fine-grained access control for microservices. In Nur Zincir-Heywood, Guillaume Bonfante, Mourad Debbabi, and Joaquin Garcia-Alfaro, editors, *Proceedings of the International Symposium on Foundations and Practice of Security*, pages 285–300, Cham, 2019. Springer International Publishing.

[27] A. Bánáti, E. Kail, K. Karóczkai, and M. Kozlovszky. Authentication and authorization orchestrator for microservice-based software architectures. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1180–1184, May 2018.

[28] D. Nagothu, R. Xu, S. Y. Nikouei, and Y. Chen. A microservice-enabled architecture for smart surveillance using blockchain technology. In *2018 IEEE International Smart Cities Conference (ISC2)*, pages 1–4, Sep. 2018.

[29] M. Pahl, F. Aubet, and S. Liebald. Graph-based iot microservice security. In *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, pages 1–3, April 2018.

[30] Tran Quang Thanh, S. Covaci, T. Magedanz, P. Gouvas, and A. Zafeiropoulos. Embedding security and privacy into the development and operation of cloud applications and services. In *2016 17th International Telecommunications Network Strategy and Planning Symposium (Networks)*, pages 31–36, Sep. 2016.

[31] Y. Sun, S. Nanda, and T. Jaeger. Security-as-a-service for microservices-based cloud applications. In *2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)*, pages 50–57, Nov 2015.

[32] A. Buzachis and M. Villari. Basic principles of osmotic computing: Secure and dependable microelements (mels) orchestration leveraging blockchain facilities. In *2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion)*, pages 47–52, Dec 2018.

[33] V. M. George and Q. H. Mahmoud. Claimsware: A claims-based middleware for securing iot services. In *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, volume 1, pages 649–654, July 2017.

[34] A. Ranjbar, M. Komu, P. Salmela, and T. Aura. Synaptic: Secure and persistent connectivity for containers. In *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, pages 262–267, May 2017.

[35] M. Ahmadvand and A. Ibrahim. Requirements reconciliation for scalable and secure microservice (de)composition. In *2016 IEEE 24th International Requirements Engineering Conference Workshops (REW)*, pages 68–73, Sep. 2016.

[36] K. A. Torkura, M. I. H. Sukmana, and A. V. D. M. Kayem. A cyber risk based moving target defense mechanism for microservice architectures. In *2018 IEEE Intl Conf on Parallel Distributed Processing with Applications, Ubiquitous Computing Communications, Big Data Cloud Computing, Social Computing Networking, Sustainable Computing Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, pages 932–939, Dec 2018.

[37] H. Jin, Z. Li, D. Zou, and B. Yuan. Dseom: A framework for dynamic security evaluation and optimization of mtd in container-based cloud. *IEEE Transactions on Dependable and Secure Computing*, pages 1–12, 2019.

[38] C. Gerking and D. Schubert. Component-based refinement and verification of information-flow security policies for cyber-physical microservice architectures. In *2019 IEEE International Conference on Software Architecture (ICSA)*, pages 61–70, March 2019.

[39] A. Osman, P. Bruckner, H. Salah, F. H. P. Fitzek, T. Strufe, and M. Fischer. Sandnet: Towards high quality of deception in container-based microservice architectures. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pages 1–7, May 2019.

[40] M. Pahl and F. Aubet. All eyes on you: Distributed multi-dimensional iot microservice anomaly detection. In *2018 14th International Conference on Network and Service Management (CNSM)*, pages 72–80, Nov 2018.

[41] R. Ravichandiran, H. Bannazadeh, and A. Leon-Garcia. Anomaly detection using resource behaviour analysis for autoscaling systems. In *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, pages 192–196, June 2018.

[42] Z. Wen, T. Lin, R. Yang, S. Ji, r. Ranjan, A. Romanovsky, C. Lin, and J. Xu. Ga-par: Dependable microservice orchestration framework for geo-distributed clouds. *IEEE Transactions on Parallel and Distributed Systems*, pages 1–16, 2019.

[43] D. Lu, D. Huang, A. Walenstein, and D. Medhi. A secure microservice framework for iot. In *2017 IEEE Symposium on Service-Oriented System Engineering (SOSE)*, pages 9–18, April 2017.

[44] M. Pahl and L. Donini. Securing iot microservices with certificates. In *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, pages 1–5, April 2018.

[45] A. Nehme, V. Jesus, K. Mahbub, and A. Abdallah. Securing microservices. *IT Professional*, 21(1):42–49, Jan 2019.

[46] C. Fetzer. Building critical applications using microservices. *IEEE Security Privacy*, 14(6):86–89, Nov 2016.

[47] Quy Nguyen and Oras F. Baker. Applying spring security framework and oauth2 to protect microservice architecture API. *JSW*, 14(6):257–264, 2019.

[48] Xiuyu He and Xudong Yang. Authentication and authorization of end user in microservice architecture. *Journal of Physics: Conference Series*, 910:012060, oct 2017.

[49] Oras Baker and Quy Nguyen. A novel approach to secure microservice architecture from owasp vulnerabilities. In *Proceedings of the 10th Annual CITRENZ Conference (2019)*, pages 54–58, Nelson, NZ, October 2019. ITx New Zealand's Conference of IT.

[50] Jyothi Salibindla. Microservices api security. *International Journal of Engineering Research & Technology*, 7(1):277–281, 2018.

[51] Kai Jander, Lars Braubach, and Alexander Pokahr. Practical defense-in-depth solution for microservice systems. *Journal of Ubiquitous Systems and Pervasive Networks*, 11(1):17 – 25, 2019.

[52] Kennedy A Torkura, Muhammad IH Sukmana, Feng Cheng, and Christoph Meinel. Cavas: Neutralizing application and container security vulnerabilities in the cloud native era. In *International Conference on Security and Privacy in Communication Systems*, pages 471–490. Springer, 2018.

[53] Jiyu Chen, Heqing Huang, and Hao Chen. Informer: Irregular traffic detection for containerized microservices rpc in the real world. In *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*, SEC '19, pages 389–394, New York, NY, USA, 2019. ACM.

[54] Kennedy A. Torkura, Muhammad I.H. Sukmana, and Christoph Meinel. Integrating continuous security assessments in microservices and cloud native applications. In *Proceedings of the10th International Conference on Utility and Cloud Computing*, UCC '17, pages 171–180, New York, NY, USA, 2017. ACM.

[55] Sven Akkermans, Bruno Crispo, Wouter Joosen, and Danny Hughes. Polyglot cerberos: Resource security, interoperability and multi-tenancy for iot services on a multilingual platform. In *Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, MobiQuitous '18, pages 59–68, New York, NY, USA, 2018. ACM.

[56] Jethro G. Beekman and Donald E. Porter. Challenges for scaling applications across enclaves. In *Proceedings of the 2Nd Workshop on System Software for Trusted Execution*, SysTEX'17, pages 8:1–8:2, New York, NY, USA, 2017. ACM.

[57] Daniel Guija and Muhammad Shuaib Siddiqui. Identity and access control for micro-services based 5g nfv platforms. In *Proceedings of the 13th International Conference on Availability, Reliability and Security*, ARES 2018, pages 46:1–46:10, New York, NY, USA, 2018. ACM.

[58] Xing Li, Yan Chen, and Zhiqiang Lin. Towards automated inter-service authorization for microservice applications. In *Proceedings of the ACM SIGCOMM 2019 Conference Posters and Demos*, SIGCOMM Posters and Demos '19, pages 3–5, New York, NY, USA, 2019. ACM.

[59] Gastón Márquez and Hernán Astudillo. Identifying availability tactics to support security architectural design of microservice-based systems. In *Proceedings of the 13th European Conference on Software Architecture - Volume 2*, ECSA '19, pages 123–129, New York, NY, USA, 2019. ACM.

[60] Amjad Ibrahim, Stevica Bozhinoski, and Alexander Pretschner. Attack graph generation for microservice architecture. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, SAC '19, pages 1235–1242, New York, NY, USA, 2019. ACM.

[61] Dimitri Michel Stallenberg and Annibale Panichella. Jcomix: A search-based tool to detect xml injection vulnerabilities in web applications. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ESEC/FSE 2019, pages 1090–1094, New York, NY, USA, 2019. ACM.

[62] Michel Krämer, Sven Frese, and Arjan Kuijper. Implementing secure applications in smart city clouds using microservices. *Future Generation Computer Systems*, 99:308 – 320, 2019.

[63] Kai Jander, Lars Braubach, and Alexander Pokahr. Defense-in-depth and role authentication for microservice systems. *Procedia Computer Science*, 130:456 – 463, 2018. The 9th International Conference on Ambient Systems, Networks and Technologies (ANT 2018) / The 8th International Conference on Sustainable Energy Information Technology (SEIT-2018) / Affiliated Workshops.

[64] Sarra Abidi, Mehrez Essafi, Chirine Ghedira Guegan, Myriam Fakhri, Hamad Witti, and Henda Hjjami Ben Ghezala. A web service security governance approach based on dedicated micro-services. *Procedia Computer Science*, 159:372 – 386, 2019. Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 23rd International Conference KES2019.

[65] Marwa Elsayed and Mohammad Zulkernine. Offering security diagnosis as a service for cloud saas applications. *Journal of Information Security and Applications*, 44:32 – 48, 2019.

[66] IBM. An integrated approach to insider threat protection, 2016.

[67] John Kindervag, Stephanie Balaouras, and Kelley Mak. Build security into your network's dna: The zero trust network architecture. Technical report, Forrester Research, November 2012.

[68] Rui Zhuang, Scott A. DeLoach, and Xinming Ou. Towards a theory of moving target defense. In *Proceedings of the First ACM Workshop on Moving Target Defense*, MTD '14, pages 31–40, New York, NY, USA, 2014. Association for Computing Machinery.

[69] Vasilios Mavroudis, Andrea Cerulli, Petr Svenda, Dan Cvrcek, Dusan Klinec, and George Danezis. A touch of evil: High-assurance cryptographic hardware from untrusted components. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, pages 1583–1600, New York, NY, USA, 2017. Association for Computing Machinery.

[70] Dirk Merkel. Docker: lightweight linux containers for consistent development and deployment. *Linux journal*, 2014(239):2, 2014.