



## Review article

## Securing microservices and microservice architectures: A systematic mapping study

Abdelhakim Hannousse<sup>a,\*</sup>, Salima Yahiouche<sup>b</sup><sup>a</sup> Department of Computer Science, Université 8 Mai 1945 Guelma, BP 401, Guelma 24000, Algeria<sup>b</sup> LRS laboratory, Badji Mokhtar University, BP 12, Annaba 23000, Algeria

## ARTICLE INFO

## Article history:

Received 21 June 2020

Received in revised form 4 May 2021

Accepted 16 June 2021

Available online 27 June 2021

## Keywords:

Microservices

Microservice architectures

Security

Systematic mapping

## ABSTRACT

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. 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. 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. 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.

© 2021 Elsevier Inc. All rights reserved.

## Contents

1. Introduction.....	2
2. Background.....	2
2.1. Microservice architectures.....	2
2.2. Systematic mapping.....	3
3. Related work.....	3
4. Research methodology.....	4
4.1. Research objectives.....	4
4.2. Research questions.....	4
4.3. Search process.....	4
4.4. Study selection process.....	5
4.5. Inclusion and exclusion criteria.....	5
4.6. Quality assessment.....	5
4.7. Data extraction process.....	6
4.8. Data synthesis.....	6
5. Results of the mapping.....	6
5.1. Overview of selected studies.....	6
5.2. MSA security threats (RQ1).....	8

\* Corresponding author.

E-mail addresses: [hannousse.abdelhakim@univ-guelma.dz](mailto:hannousse.abdelhakim@univ-guelma.dz) (A. Hannousse), [salima.yahiouche@univ-annaba.dz](mailto:salima.yahiouche@univ-annaba.dz) (S. Yahiouche).

5.3.	Microservice security mechanisms (RQ2).....	8
5.3.1.	Microservice security application levels (RQ3).....	9
5.3.2.	MSA security mechanisms target platforms (RQ4).....	10
5.3.3.	Microservice security V&V methods (RQ5).....	10
6.	An ontology for securing MSA.....	11
7.	Discussion.....	12
7.1.	Bias towards external threats, user and data attacks.....	12
7.2.	Lack of innovative solutions for authorization and mitigation.....	12
7.3.	Lack of appropriate solutions to secure individual microservices.....	13
7.4.	Lack of appropriate solutions to emerging technologies.....	13
7.5.	Absence of appropriate comparison techniques.....	13
8.	Threats to validity.....	13
8.1.	Construct validity.....	13
8.2.	Internal validity.....	14
8.3.	External validity.....	14
8.4.	Conclusion validity.....	14
9.	Conclusion.....	14
	Declaration of competing interest.....	14
	References.....	14

## 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 microservice is a fine-grained software unit that can be created, initialized, duplicated, and destroyed independently from other microservices of the same system. Moreover, microservices 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 microservices, 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<sup>1</sup> such as Netflix and Amazon, 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 vary from securing individual microservices into complete architectures and infrastructures.

In this article, we conduct a systematic mapping 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 the mapping results; Section 6 describes the proposed ontology for MSA; Section 7 highlights potential research gaps; Section 8 discusses threats to validity and Section 9 concludes the paper.

## 2. Background

### 2.1. Microservice 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 Fig. 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

<sup>1</sup> Threatpost: <https://threatpost.com>.

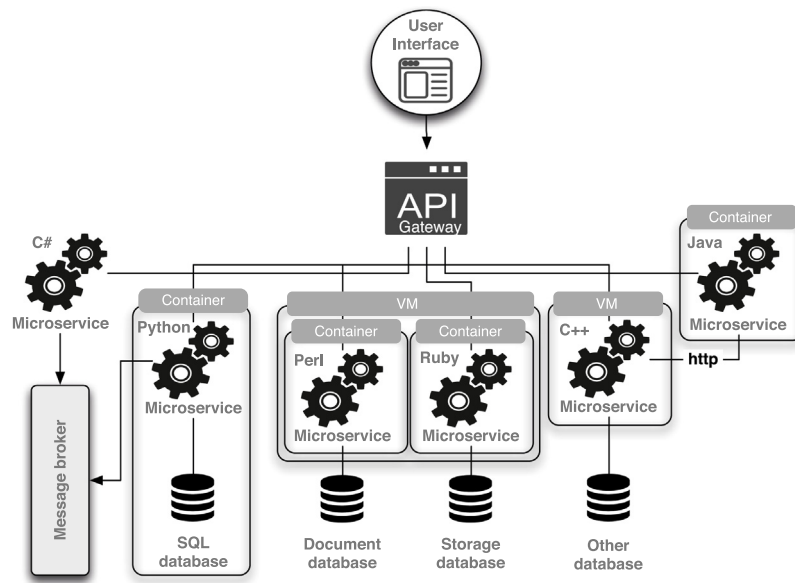


Fig. 1. Microservice architecture.

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].

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.

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. Fig. 2 depicts the overall process for conducting systematic mapping studies as proposed in [8]. The quality assessment step is depicted in Fig. 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

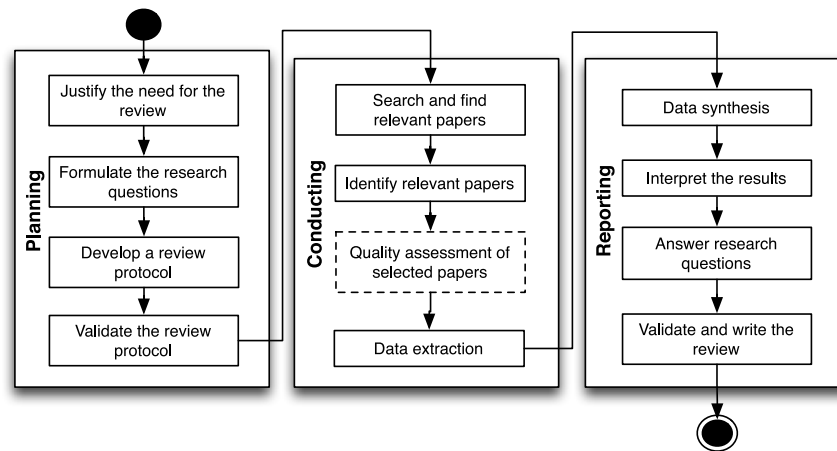


Fig. 2. The overall process for elaborating systematic mapping.

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 in 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] and Felderer and Craver [15], 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 objectives

The adoption of microservice architectures in developing high-sensitive systems such as industry, raises a question on the risks associated with the deployment of microservices in distributed and highly accessible platforms. In this study, we aim to identify the threats which could target individual microservices and microservice architectures and may endanger systems' security, stability and reliability. Moreover, we aim to collect and categorize all known threats and the endeavors to mitigate and prevent them in easy-to-handle, reliable and extensible structures such as ontology. This enables: (1) researchers and practitioners to have an overview of the risks that could threaten their existing or future systems and (2) share experiences on how to mitigate and prevent already recognized security threats. To sum up, the objectives of this research can be reformulated as follows:

- 01.** identify and categorize threats targeting individual microservices and microservice architectures.
- 02.** recognize and categorize the endeavors to detect, mitigate and prevent recognized security threats.
- 03.** discern the set of validation techniques and tools used to corroborate identified security mechanisms.
- 04.** deduce and share a lightweight ontology for securing microservice based systems.

### 4.2. Research questions

The aim of this study is to identify the set of security vulnerabilities and how they can 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. [16]. This study is conducted with five main questions in mind. These questions are described in Table 1 with their corresponding motivation and related objectives.

### 4.3. 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 [17]. Population refers to the application area which is microservices and microservice architectures where intervention

**Table 1**  
Research questions with their motivations and relevant objectives.

No	Research Question	Main Motivation	Objectives
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.	O1, O4
RQ2	What are existing approaches and techniques used for securing microservices 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.	O2, O4
RQ3	At what level of architecture the proposed techniques and approaches are applicable for securing microservices?	This research question indicates where security is applied highlighting the less focused levels of microservice architectures.	O2, O4
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.	O2, O4
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.	O3

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. [16]. 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 [18,19]. 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.

#### 4.4. 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.

#### 4.5. 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

**Table 2**  
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 3**  
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

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 Tables 2 and 3 respectively.

#### 4.6. Quality assessment

As advised by Peterson et al. [8], quality assessment of identified papers is also required for mapping studies to answer that sufficient information is available for data extraction. Therefore, we conduct a quality assessment process to evaluate the overall strength and included details of selected papers. A questionnaire formed of four items has been considered. The questionnaire is inspired from [20,21] and adapted to our research topic. The set of adopted assessment criteria are summarized in Table 4 with their associated scores.



**Table 4**  
Quality assessment criteria.

ID	Criteria	Score
QA1	Does the study clearly address any of security issues in microservice architectures?	Yes/No
QA2	Does the study present a clear solution to any security threat in microservice architectures?	{0, 0.5, 1}
QA3	Has the study been cited by other articles?	{0, 1}
QA4	Has the study been published in ranked journal or conference proceedings	{1, 0.5, 0}

In this study, score 1 is awarded to studies when an item could be answered as Yes and 0 when the answer was No (case of QA1). For QA2 criterion, score 1 is attributed to studies presenting a detailed and validated solution to a security threat, score 0.5 is attributed to studies providing only an overview of the solution or framework, and score 0 is attributed to studies without clear solutions. For QA3 item, score 1 is attributed to studies with number of citations greater or equal 3; score 0 is attributed to studies with number of citations less than 3. Google Scholar is considered to get the number of citations. This latter is adopted to not penalize early publications. For QA4 item, paper sources are ranked following the CORE conference ranking: A or B (+1), C (+0.5) and not ranked (0) and Journal Citation Reports (JCR): Q1-2(+1), Q3-4 (0.5), not ranked (0). The four adopted criteria are treated equally and the total score of each study is calculated by summing up their four measured values. Only studies with a quality score greater than or equal 2 are included in this study.

#### 4.7. Data extraction process

Following the guidelines of Peterson et al. [7] a data extraction form is designed as illustrated in Table 5. 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.

#### 4.8. 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

**Table 6**  
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

existing taxonomies [1,11,22] in identifying categories and their relationships. We also used grounded theory [23] 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 4.2.

#### 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 6 shows the number of returned papers by each library.

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. Fig. 3 depicts the overall selection process.

Fig. 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. Fig. 4 also shows that the maximum number of publications come from IEEE Xplorer and none from Wiley meets our inclusion criteria.

Table 7 shows the complete list of selected studies with their year, type of publication, how they are found and their measured quality assessment score.

**Table 5**  
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

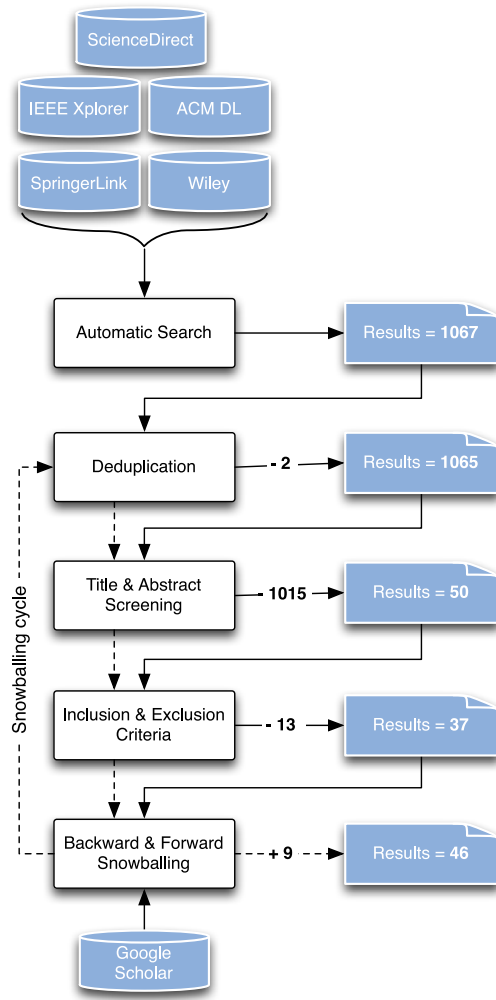


Fig. 3. Paper selection process.

Following the guidelines of Kuhrmann et al. [16], we also experienced the use of Word Clouds to analyze the appropriateness of our result set of primary studies. Fig. 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

**Table 7**  
List of selected studies with their quality assessment score: C: Conference paper, J: Journal paper, A: Automatic search, S: Snowballing.

ID	Cite	Year	Type	Publisher	Search type	Score
p1	[12]	2019	C	Springer	A	3
p2	[24]	2018	C	Springer	A	3
p3	[25]	2020	C	Springer	A	3
p4	[26]	2017	C	Springer	A	4
p5	[27]	2017	C	Springer	A	3
p6	[28]	2018	C	Springer	A	3
p7	[29]	2019	C	Springer	A	3
p8	[30]	2018	C	IEEE	A	3
p9	[31]	2018	C	IEEE	A	3
p10	[32]	2018	C	IEEE	A	4
p11	[33]	2016	C	IEEE	A	3
p12	[34]	2015	C	IEEE	A	3.5
p13	[35]	2018	C	IEEE	A	2
p14	[36]	2017	C	IEEE	A	4
p15	[37]	2017	C	IEEE	A	4
p16	[38]	2016	C	IEEE	A	3
p17	[1]	2018	C	IEEE	A	4
p18	[39]	2018	C	IEEE	A	3
p19	[40]	2019	J	IEEE	A	4
p20	[41]	2019	C	IEEE	A	3.5
p21	[42]	2019	C	IEEE	A	3
p22	[43]	2018	C	IEEE	A	4
p23	[44]	2018	C	IEEE	A	3
p24	[45]	2019	J	IEEE	A	4
p25	[46]	2017	C	IEEE	S	3
p26	[47]	2018	C	IEEE	S	4
p27	[48]	2019	J	IEEE	S	3
p28	[49]	2016	J	IEEE	S	4
p29	[50]	2019	J	JSW	S	3
p30	[51]	2017	J	IOP	S	3
p31	[52]	2019	C	CITRENZ	S	2
p32	[53]	2018	J	IJERT	S	3
p33	[54]	2019	J	IASKS	S	3
p34	[55]	2018	C	Springer	A	2
p35	[56]	2019	C	ACM	A	2
p36	[57]	2017	C	ACM	A	3
p37	[58]	2018	C	ACM	A	3
p38	[59]	2018	C	ACM	A	4
p39	[60]	2019	C	ACM	A	4
p40	[61]	2019	C	ACM	A	4
p41	[62]	2019	C	ACM	A	4
p42	[63]	2019	C	ACM	A	3
p43	[64]	2019	J	Science Direct	A	4
p44	[65]	2018	J	Science Direct	A	3
p45	[66]	2019	J	Science Direct	A	3
p46	[67]	2019	J	Science Direct	A	4

service. Attacks, vulnerabilities and risks are rarely used in titles and abstracts.

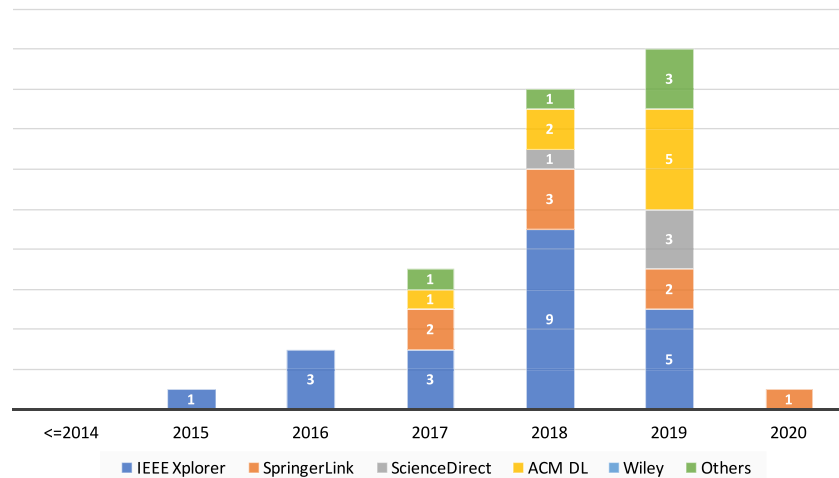
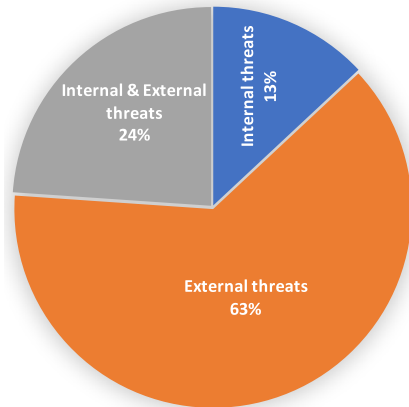


Fig. 4. Distribution of selected studies by year and digital library.



**Fig. 5.** Keywords cloud of primary studies.



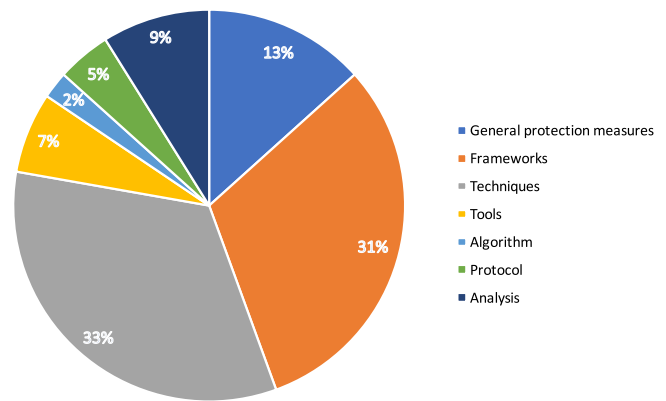
**Fig. 6.** MSA security source of threats.

## 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). Fig. 6 depicts the distribution of identified and selected studies regarding the addressed source of threats. The results show 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).
- *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.



**Fig. 7.** MSA security solutions.

Table 8 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.

### 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 Fig. 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 8**  
MSA security threats per category.

Threats	Percentage	Studies
<b>User-based Attacks</b>		<b>58.70%</b>
Brute Force Attack		[50,52]
Cross-Site Request Forgery (CSRF)		[28,29,50,52,55,57]
Spoofing		[28,47]
Malicious Insider		[29,45]
Unauthorized Access		[12,25,28–31,33,35–37,43,46,47,50–53,58–60,64–66]
Violate non-repudiation		[24]
<b>Data Attacks</b>		<b>47.83%</b>
Eavesdropping		[64]
Heartbleed		[28,47]
Man in The Middle attack (MiTM)		[28,37,61]
Padding Oracle On Downgraded Legacy Encryption attack (POODLE)		[28]
Replay attack		[37,54,65]
Sensitive data exposure		[24,26,28,31,35,36,41,45,46,49,60,64,66,67]
Sniffing attack		[25,28,29]
<b>Infrastructure Attacks</b>		<b>41.30%</b>
Compromise containers		[1,37,40,55–57,62]
Compromise virtual machines		[1,34,55,57]
Compromise discovery service		[12,28]
Compromise hypervisor		[1,26,28,49]
Compromise management interface		[45]
Compromise network nodes		[1]
Compromise operating systems		[26,49]
Downgrade attack		[24,58]
Hardware backdoors		[28]
Malicious images		[28,62]
Malicious provider		[28]
Misconfiguration		[28,43]
Port scan attack		[25]
Sandbox escape		[28]
Session hijacking		[28,29]
Stress attack		[44,45]
Cold boot attack		[26]
<b>Software Attacks</b>		<b>50.00%</b>
Code reuse attack		[39,61]
Compromise microservices		[1,27,28,32,34,40,42,43,47,54,55,57,62,65,67]
Disrupt sensitive operation		[24]
Denial of Service (DoS)		[28,37,50,52,55,57,63]
Injection		[1,29,50,52,55,57,63]

- *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 microservice-based systems.
- *Prevention*: techniques that try to stop attacks from happening in the first place. Prevention techniques need to be considered when developing new microservice-based systems.

Table 9 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%<sup>2</sup>), and prevention (34.78%), where less attention is being paid to mitigation (10.87%).

<sup>2</sup> 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.

### 5.3.1. Microservice 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 microservices can be blocked or compromised through injection of malicious code. Thus, security measures to adopt only trusted microservices 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 9**  
MSA security mechanism per category.

Security Mechanisms	Percentage	Studies
<b>Authentication</b>	<b>28.26%</b>	
Centralized Access Control Manager	6.52%	[29,33,37]
Certificates	6.52%	[12,28,47]
Open ID	6.52%	[30,48,59]
Single Sign On (SSO)	4.35%	[30,51]
White-list HTTP/IP	4.35%	[12,53]
HIP exchange protocol	2.17%	[37]
J-PAKE protocol	4.35%	[54,65]
Distribute sessions	2.17%	[51]
HTTP signatures	2.17%	[53]
<b>Authorization</b>	<b>21.74%</b>	
Attribute Based Access Control (ABAC)	2.17%	[33]
Role Based Access Control (RBAC)	6.52%	[29,54,59]
R/W Permission to message broker	2.17%	[12]
OAuth 2	17.39%	[29,30,33,48,50,52,53,59]
<b>Authentication &amp; Authorization</b>	<b>23.91%</b>	
JSON Web Token (JWT)	17.39%	[12,28,30,36,48,51,53,59]
Firewalls	8.70%	[25,32,33,48]
<b>Auditing</b>	<b>43.48%</b>	
Continuous monitoring	34.78%	[1,24,25,27,32,34,42–44,48,55–57,60,62,66]
Scan container images	2.17%	[12]
Static/Dynamic code analysis	8.70%	[28,49,60,67]
Machine learning	6.52%	[32,43,56]
Intrusion detection	4.35%	[1,48]
<b>Mitigation</b>	<b>10.87%</b>	
Roll-back/Restart microservices	2.17%	[1]
Scale up/down N-variant microservices	2.17%	[1]
Short-lived tokens	2.17%	[29]
Diversification	4.35%	[1,39]
IP shuffling	2.17%	[40]
Live migration	4.35%	[40,42]
Deception	2.17%	[42]
Isolation of suspicious microservices	2.17%	[1]
<b>Prevention</b>	<b>34.78%</b>	
Blockchain technology	4.35%	[31,35]
Encryption	10.87%	[1,24,37,46,64]
Hardware Security Module (HSM)	2.17%	[28]
Least privilege	2.17%	[28]
No shared memory access	2.17%	[28]
Proper design	6.52%	[38,41,61]
Secure languages	4.35%	[28,58]
Smart contracts	6.52%	[31,33,35]
TLS protocol	6.52%	[26,28,36]
SGX technology with enclaves	6.52%	[26,28,49]

- *Deployment*: Containers holding microservices 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 [68] to initiate attacks. These vulnerabilities need to be tackled by introducing appropriate error and backdoor detection mechanisms.

Table 10 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.

### 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 11 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.

### 5.3.3. Microservice 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.

**Table 10**  
MSA security application levels.

Application Layer	Percentage	Studies
Microservice	20.00%	[1,27,39,46–48,60,63,67]
Composition	6.67%	[38,41,48]
API	35.56%	[12,24,28–31,33,35–37,46,50–53,59]
Communication	22.22%	[26,28,30,31,35–37,54,59,65]
Deployment	13.33%	[26,40,45,46,49,66]
Soft-Infrastructure	68.89%	[1,12,24,25,27–34,36,37,39,42–44,47–49,51,55–60,62,64,67]
Hard-Infrastructure	8.89%	[26,40,42,49]

**Table 11**  
MSA security application platforms.

Applications/ Platforms	Percentage	Studies
IoT applications	13.04%	[32,36,43,46,47,58]
Cloud platforms	28.26%	[26,30,33,34,37,40,44,45,49,55,57,64,67]
Osmotic computing	2.17%	[35]
Container-based platforms	10.87%	[39,40,42,56,62]
Web applications	6.52%	[44,63,66]
Springer platform	4.35%	[50,52]
Kubernetes platform	2.17%	[25]
5G platform	2.17%	[59]
Independent	34.78%	[1,12,24,27–29,31,38,41,48,51,53,54,60,61,65]

**Table 12**  
MSA security verification and validation methods.

V&V methods	Percentage	Studies
Simulation	6.52%	[25,36,47]
Manual Testing	8.70%	[42,44,52,66]
Performance analysis	39.13%	[1,26,28,29,32,34,36,37,40,42–44,47,49,57,58,64,67]
Qualitative analysis	10.87%	[27,30,51,53,64]
Quantitative analysis	8.70%	[37,43,56,67]
Adhoc metrics	6.52%	[39,42,43]
Case study based validation	26.09%	[24,31,33,36,38,46,54,60–63,65]
Proof of concept (POC)	2.17%	[50]
Tool-based testing	10.87%	[52,55,59,61,67]
Formal verification	2.17%	[41]
Complexity measuring	6.52%	[1,42,45]
Methodology-based validation	4.35%	[39,45]

- *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.<sup>3</sup>
- *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 12 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. Validation by simulation and adhoc metrics are found used in equal measure with a rate equal to 6.52%. Most simulation methods adopted 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 similar to the approach proposed in [69]. 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. Fig. 8 describes the overall MSA ontology retrieved from this study.

<sup>3</sup> IntelliJ IDEA: <https://www.jetbrains.com/fr-fr/idea/>.

**Table 13**  
Relationships between ontology classes.

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

**Table 14**  
OWL-DL Query samples.

ID	Query
Q1	SecurityMechanism <b>and</b> (treats <b>value</b> Unauthorized_Access)
Q2	SecurityThreat <b>and</b> (treatedBy <b>value</b> Continuous_Monitoring)
Q3	SecurityMechanism <b>and</b> (applicableAt <b>value</b> Microservice)
Q4	SecurityThreat <b>and</b> (hasSource <b>value</b> Internal)

The ontology is developed using Protégé<sup>4</sup> and its consistency and coherence are checked using the Protégé Debugger option. The main concepts of the ontology are:

1. *SecurityMechanism*: 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 13.

For the usability of the ontology, OWL-DL queries can be used to investigate the ontology structure. Table 14 describes some of useful queries described in OWL-DL. In Table 14, Q1 can be used to retrieve the list of security 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.

## 7. Discussion

In this section, we highlight potential research gaps that require further investigations in the field. Those research gaps are basically identified through the analysis of the mapping results.

### 7.1. Bias towards external threats, user and data attacks

Although IBM X-Force [70] 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 considerable policy changes and continuous monitoring of internal traffics. This is owing to privileges awarded and sensitive data exposed to insiders. Moreover, the diversity of attacks is basically due to the adoption of Zero Trust model [71] 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.

### 7.2. Lack of innovative solutions for authorization and mitigation

The mapping results showed that much emphasis is put on authentication and authorization techniques. In fact, this is defensible since 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. On the contrary, 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) [72]. 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 vary 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. Therefore, more powerful and innovative solutions are required for authentication/authorization. Moreover, due to the lower rate of mitigation techniques and their applicability to existing microservice-based systems, we advocate more research on mitigation.

<sup>4</sup> <https://protege.stanford.edu/>.



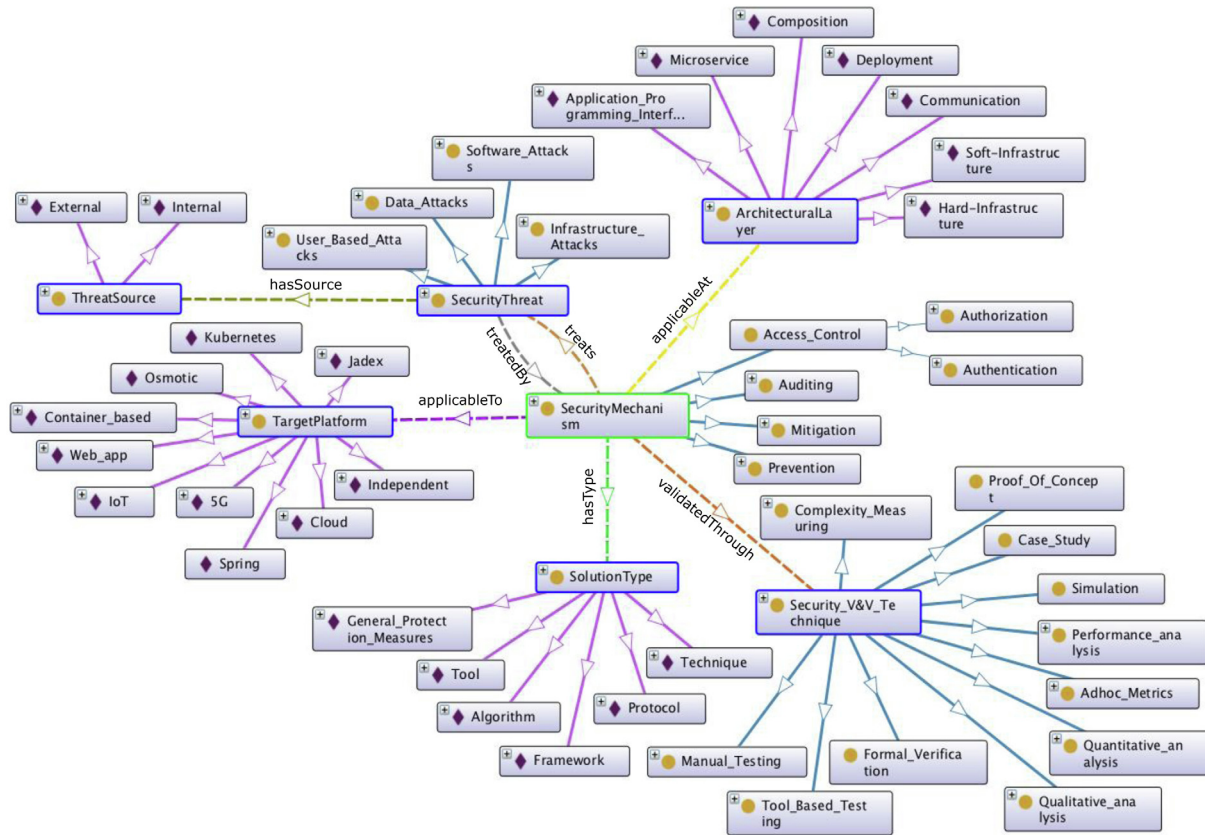


Fig. 8. Ontology for MSA security.

### 7.3. Lack of appropriate solutions to secure individual microservices

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.

### 7.4. Lack of appropriate solutions to emerging technologies

Cloud-focused and platform independent solutions are found within 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. However, more attention should be paid to 5G platforms. These are emerging technologies that requires specific attentions.

### 7.5. Absence of appropriate comparison techniques

The study revealed the adoption of several validation techniques. This is due to the nature of proposed solutions. For example, formal verification is adopted when formal specification of systems and their properties are described (e.g.; p20). In the other

side, performance analysis is very important for checking the suitability of proposed solutions to particular environments and platforms; it is also important for decision makers. Even though performance analysis is applicable to different proposals and found adopted with the higher rate (39.13%) not all the studies have adopted it. Note that the diversity and incompatibility of validation techniques make the comparison of proposed solutions harder. The adoption of performance analysis by all the studies may alleviate this problem.

## 8. Threats to validity

In this section we describe construct, internal, external and conclusion validity threats and how we mitigated their effects on the obtained results.

### 8.1. Construct validity

A construct validity threat to our study concerns the identification of primary studies from the large set of papers found in the literature. For this sake, we adopted the guidelines of Kuhrmann et al. [16] for the selection of search engines. Search keywords are carefully chosen; they are meant to be as general as possible and designed following the guidelines Petticrew and Roberts [17]. To avoid bias to search engines, we completed our search by snowballing technique [19] 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 where only peer-reviewed journal and conference papers are accepted 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 [73] were omitted.

### 8.2. Internal validity

An internal validity threat concerns the data extraction from the set of included studies. Both authors are incorporated in this process. Following the guidelines of Peterson et al. [7], a data extraction form is designed by the first author, discussed and updated after a deep discussion with the second author. Each author has filled the form independently; after solving the disagreements between the two authors, a unique and final form is adopted for the rest of the study. Following the guidelines of Kitchenham et al. [6], the kappa coefficient is estimated and found equal 0.94 which is evaluated as very good or almost perfect score according to [6].

### 8.3. External validity

An external validity threat concerns the generalization of the results of the study. The classification schemes adopted in this mapping are constructed based on retrieved papers. Future published studies may fit to the proposed scheme where many others may not. For this sake, we developed an ontology incorporating all the classification schemes. The constructed ontology is made freely available for researchers and practitioners for further updates when new studies are published and not fitting the proposed schemes.

### 8.4. Conclusion validity

A conclusion validity threat to our study concerns the adoption of taxonomies for security threats and mechanisms. In fact, several taxonomies are investigated [1,11,22], however, none of those taxonomies enables the proper classification of all the identified studies. Thus, we used open and selective coding from grounded theory [23] 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 are either used as they are or adapted to fulfill the context of our study.

## 9. 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 an ontology summarizing and gathering the retrieved results. The proposed ontology can be used as a guide to developers about 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 remains 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.

### Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References

- [1] T. Yarygina, A.H. Bagge, Overcoming security challenges in microservice architectures, in: 2018 IEEE Symposium on Service-Oriented System Engineering (SOSE), 2018, pp. 11–20, <http://dx.doi.org/10.1109/SOSE.2018.00011>.
- [2] S. Baškarada, V. Nguyen, A. Koronios, Architecting microservices: Practical opportunities and challenges, *J. Comput. Inf. Syst.* (2018) 1–9, <http://dx.doi.org/10.1080/08874417.2018.1520056>.
- [3] N. Dragoni, S. Giallorenzo, A.L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, L. Safina, *Microservices: Yesterday, Today, and Tomorrow*, Springer International Publishing, Cham, 2017, pp. 195–216, (Chapter 12), [http://dx.doi.org/10.1007/978-3-319-67425-4\\_12](http://dx.doi.org/10.1007/978-3-319-67425-4_12).
- [4] N. Alshuqayran, N. Ali, R. Evans, A systematic mapping study in microservice architecture, in: 2016 IEEE 9th International Conference on Service-Oriented Computing and Applications (SOCA), IEEE, 2016, pp. 44–51.
- [5] J. Bogner, J. Fritzsche, S. Wagner, A. Zimmermann, Microservices in industry: Insights into technologies, characteristics, and software quality, in: 2019 IEEE International Conference on Software Architecture Companion (ICSA-C), 2019, pp. 187–195, <http://dx.doi.org/10.1109/ICSA-C.2019.00041>.
- [6] B.A. Kitchenham, D. Budgen, P. Brereton, *Evidence-Based Software Engineering and Systematic Reviews*, Chapman & Hall/CRC, 2015.
- [7] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson, Systematic mapping studies in software engineering, in: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering, EASE'08, Swindon, UK, 2008, pp. 68–77.
- [8] K. Petersen, S. Vakkalanka, L. Kuzniarz, Guidelines for conducting systematic mapping studies in software engineering: An update, *Inf. Softw. Technol.* 64 (2015) 1–18, <http://dx.doi.org/10.1016/j.infsof.2015.03.007>.
- [9] A.P. Vale, G. Márquez, H. Astudillo, E.B. Fernandez, Security mechanisms used in microservices-based systems: A systematic mapping, in: XLV Latin American Computing Conference, 2019, pp. 1–10.
- [10] D. Yu, Y. Jin, Y. Zhang, X. Zheng, A survey on security issues in services communication of microservices-enabled fog applications, *Concurr. Comput.: Pract. Exper.* 31 (22) (2019) e4436, e4436 cpe.4436, <http://dx.doi.org/10.1002/cpe.4436>, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpe.4436.
- [11] L. de Aguiar Monteiro, W.H.C. Almeida, R.R. Hazin, A.C. de Lima, S.K.G. e Silva, F.S. Ferraz, A survey on microservice security—trends in architecture, privacy and standardization on cloud computing environments, *Int. J. Adv. Secur.* 11 (3–4) (2018) 201–213.
- [12] P. Nkomo, M. Coetzee, Software development activities for secure microservices, in: S. Misra, O. Gervasi, B. Murgante, E. Stankova, V. Korkhov, C. Torre, A.M.A. Rocha, D. Taniar, B.O. Apduhan, E. Tarantino (Eds.), *Proceedings of International Conference on Computational Science and its Applications, ICCSA 2019*, Springer International Publishing, Cham, 2019, pp. 573–585.
- [13] S. Sultan, I. Ahmad, T. Dimitriou, Container security: Issues, challenges, and the road ahead, *IEEE Access* 7 (2019) 52976–52996, <http://dx.doi.org/10.1109/ACCESS.2019.2911732>.
- [14] M. Bélair, S. Lanepce, J.-M. 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, ACM, New York, NY, USA, 2019, pp. 76:1–76:6, <http://dx.doi.org/10.1145/3339252.3340502>.
- [15] M. Felderer, J.C. Carver, Guidelines for systematic mapping studies in security engineering, in: *Empirical Research for Software Security: Foundations and Experience*, CRC Press, 2018, pp. 47–69.

- [16] M. Kuhrmann, D.M. Fernández, M. Daneva, On the pragmatic design of literature studies in software engineering: an experience-based guideline, *Empir. Softw. Eng.* 22 (6) (2017) 2852–2891, <http://dx.doi.org/10.1007/s10664-016-9492-y>.
- [17] M. Petticrew, H. Roberts, *Systematic Reviews in the Social Sciences: A Practical Guide*, John Wiley & Sons, Ltd, 2006.
- [18] C. 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*, Association for Computing Machinery, New York, NY, USA, 2014, pp. 1–10, <http://dx.doi.org/10.1145/2601248.2601268>.
- [19] C. Wohlin, Second-generation systematic literature studies using snowballing, in: *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering, EASE '16*, Association for Computing Machinery, New York, NY, USA, 2016, pp. 1–6, <http://dx.doi.org/10.1145/2915970.2916006>.
- [20] M.S. Farooq, S. Riaz, A. Abid, T. Umer, Y.B. Zikria, Role of iot technology in agriculture: A systematic literature review, *Electronics* 9 (2). <http://dx.doi.org/10.3390/electronics9020319>.
- [21] A. Fernandez, E. Insfran, S. Abrahão, Usability evaluation methods for the web: A systematic mapping study, *Inf. Softw. Technol.* 53 (8) (2011) 789–817, <http://dx.doi.org/10.1016/j.infsof.2011.02.007>.
- [22] OWASP, *Owasp Top 10: The Ten Most Critical Web Application Security Risks*, Tech. Rep., OWASP Foundation, 2017.
- [23] A.L. Strauss, J.M. Corbin, *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, Sage Publications, Thousand Oaks, Calif, 1998.
- [24] M. Ahmadvand, A. Pretschner, K. Ball, D. Eyring, Integrity protection against insiders in microservice-based infrastructures: From threats to a security framework, in: M. Mazzara, I. Ober, G. Salaün (Eds.), *Proceedings of Federation of International Conferences on Software Technologies: Applications and Foundations*, Springer International Publishing, Cham, 2018, pp. 573–588.
- [25] N. Surantha, F. Ivan, Secure kubernetes networking design based on zero trust model: A case study of financial service enterprise in indonesia, in: L. Barolli, F. Xhafa, O.K. Hussain (Eds.), *Proceedings of International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, Springer International Publishing, Cham, 2020, pp. 348–361.
- [26] S. Brenner, T. Hundt, G. Mazzeo, R. Kapitza, Secure cloud micro services using intel sgx, in: L.Y. Chen, H.P. Reiser (Eds.), *Proceedings of IFIP International Conference on Distributed Applications and Interoperable Systems*, Springer International Publishing, Cham, 2017, pp. 177–191.
- [27] C. Otterstad, T. Yarygina, Low-level exploitation mitigation by diverse microservices, in: F. De Paoli, S. Schulte, E. Broch Johnsen (Eds.), *Proceedings of the 6th IFIP WG 2.14 European Conference on Service-Oriented and Cloud Computing*, Springer International Publishing, Cham, 2017, pp. 49–56, [http://dx.doi.org/10.1007/978-3-319-67262-5\\_4](http://dx.doi.org/10.1007/978-3-319-67262-5_4).
- [28] T. Yarygina, C. Otterstad, A game of microservices: Automated intrusion response, in: S. Bonomi, E. Rivière (Eds.), *Proceedings of IFIP International Conference on Distributed Applications and Interoperable Systems*, Springer International Publishing, Cham, 2018, pp. 169–177, [http://dx.doi.org/10.1007/978-3-319-93767-0\\_12](http://dx.doi.org/10.1007/978-3-319-93767-0_12).
- [29] A. Nehme, V. Jesus, K. Mahbub, A. Abdallah, Fine-grained access control for microservices, in: N. Zincir-Heywood, G. Bonfante, M. Debbabi, J. Garcia-Alfaro (Eds.), *Proceedings of the International Symposium on Foundations and Practice of Security*, Springer International Publishing, Cham, 2019, pp. 285–300, [http://dx.doi.org/10.1007/978-3-030-18419-3\\_19](http://dx.doi.org/10.1007/978-3-030-18419-3_19).
- [30] A. Bánáti, E. Kail, K. Karóczkai, M. Kozlovsky, Authentication and authorization orchestrator for microservice-based software architectures, in: *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2018, pp. 1180–1184, <http://dx.doi.org/10.23919/MIPRO.2018.8400214>.
- [31] D. Nagothu, R. Xu, S.Y. Nikouei, Y. Chen, A microservice-enabled architecture for smart surveillance using blockchain technology, in: *2018 IEEE International Smart Cities Conference (ISC2)*, 2018, pp. 1–4, <http://dx.doi.org/10.1109/ISC2.2018.8656968>.
- [32] M. Pahl, F. Aubet, S. Liebold, Graph-based iot microservice security, in: *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*, 2018, pp. 1–3, <http://dx.doi.org/10.1109/NOMS.2018.8406118>.
- [33] Tran Quang Thanh, S. Covaci, T. Magedanz, P. Gouvas, 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)*, 2016, pp. 31–36, <http://dx.doi.org/10.1109/NETWKS.2016.7751149>.
- [34] Y. Sun, S. Nanda, T. Jaeger, Security-as-a-service for microservices-based cloud applications, in: *2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)*, 2015, pp. 50–57, <http://dx.doi.org/10.1109/CloudCom.2015.93>.
- [35] A. Buzachis, M. and 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)*, 2018, pp. 47–52, <http://dx.doi.org/10.1109/UCC-Companion.2018.00033>.
- [36] V.M. George, Q.H. Mahmoud, Claimsware: A claims-based middleware for securing iot services, in: *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, 1, 2017, pp. 649–654, <http://dx.doi.org/10.1109/COMPSAC.2017.85>.
- [37] A. Ranjbar, M. Komu, P. Salmela, T. Aura, Synaptic: Secure and persistent connectivity for containers, in: *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, 2017, pp. 262–267, <http://dx.doi.org/10.1109/CCGRID.2017.62>.
- [38] M. Ahmadvand, A. Ibrahim, Requirements reconciliation for scalable and secure microservice (de)composition, in: *2016 IEEE 24th International Requirements Engineering Conference Workshops (REW)*, 2016, pp. 68–73, <http://dx.doi.org/10.1109/REW.2016.026>.
- [39] K.A. Torkura, M.I.H. Sukmana, 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)*, 2018, pp. 932–939, <http://dx.doi.org/10.1109/BDCloud.2018.00137>.
- [40] H. Jin, Z. Li, D. Zou, B. Yuan, Dseom: A framework for dynamic security evaluation and optimization of mtd in container-based cloud, *IEEE Trans. Dependable Secure Comput.* (2019) 1–12, <http://dx.doi.org/10.1109/TDSC.2019.2916666>.
- [41] C. Gerking, 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)*, 2019, pp. 61–70, <http://dx.doi.org/10.1109/ICSA.2019.00015>.
- [42] A. Osman, P. Bruckner, H. Salah, F.H.P. Fitzek, T. Strufe, M. Fischer, Sandnet: Towards high quality of deception in container-based microservice architectures, in: *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–7, <http://dx.doi.org/10.1109/ICC.2019.8761171>.
- [43] M. Pahl, F. Aubet, All eyes on you: Distributed multi-dimensional iot microservice anomaly detection, in: *2018 14th International Conference on Network and Service Management (CNSM)*, 2018, pp. 72–80.
- [44] R. Ravichandiran, H. Bannazadeh, A. Leon-Garcia, Anomaly detection using resource behaviour analysis for autoscaling systems, in: *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, 2018, pp. 192–196, <http://dx.doi.org/10.1109/NETSOFT.2018.8460025>.
- [45] Z. Wen, T. Lin, R. Yang, S. Ji, R. Ranjan, A. Romanovsky, C. Lin, J. Xu, G. par: Dependable microservice orchestration framework for geo-distributed clouds, *IEEE Trans. Parallel Distrib. Syst.* (2019) 1–16, <http://dx.doi.org/10.1109/TPDS.2019.2929389>.
- [46] D. Lu, D. Huang, A. Walenstein, D. Medhi, A secure microservice framework for iot, in: *2017 IEEE Symposium on Service-Oriented System Engineering (SOSE)*, 2017, pp. 9–18, <http://dx.doi.org/10.1109/SOSE.2017.27>.
- [47] M. Pahl, L. Donini, Securing iot microservices with certificates, in: *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*, 2018, pp. 1–5, <http://dx.doi.org/10.1109/NOMS.2018.8406189>.
- [48] A. Nehme, V. Jesus, K. Mahbub, A. Abdallah, Securing microservices, *IT Prof.* 21 (1) (2019) 42–49, <http://dx.doi.org/10.1109/MITP.2018.2876987>.
- [49] C. Fetzter, Building critical applications using microservices, *IEEE Secur. Privacy* 14 (6) (2016) 86–89, <http://dx.doi.org/10.1109/MSP.2016.129>.
- [50] Q. Nguyen, O.F. Baker, Applying spring security framework and oauth2 to protect microservice architecture API, *JSW* 14 (6) (2019) 257–264.
- [51] X. He, X. Yang, Authentication and authorization of end user in microservice architecture, *J. Phys. Conf. Ser.* 910 (2017) 012060, <http://dx.doi.org/10.1088/1742-6596/910/1/012060>.
- [52] O. Baker, Q. Nguyen, A novel approach to secure microservice architecture from owasp vulnerabilities, in: *Proceedings of the 10th Annual CITREZ Conference (2019)*, ITx New Zealand's Conference of IT, Nelson, NZ, 2019, pp. 54–58.
- [53] J. Salibindla, *Microservices api security*, *Int. J. Eng. Res. Technol.* 7 (1) (2018) 277–281.
- [54] K. Jander, L. Braubach, A. Pokahr, Practical defense-in-depth solution for microservice systems, *J. Ubiquit. Syst. Pervasive Netw.* 11 (1) (2019) 17–25, <http://dx.doi.org/10.5383/juspn.11.01.003>.
- [55] K.A. Torkura, M.I. Sukmana, F. Cheng, C. Meinel, Cavas: Neutralizing application and container security vulnerabilities in the cloud native era, in: *International Conference on Security and Privacy in Communication Systems*, Springer, 2018, pp. 471–490.
- [56] J. Chen, H. Huang, H. 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*, ACM, New York, NY, USA, 2019, pp. 389–394, <http://dx.doi.org/10.1145/3318216.3363375>.

- [57] K.A. Torkura, M.I. Sukmana, C. Meinel, Integrating continuous security assessments in microservices and cloud native applications, in: Proceedings of The 10th International Conference on Utility and Cloud Computing, UCC '17, ACM, New York, NY, USA, 2017, pp. 171–180, <http://dx.doi.org/10.1145/3147213.3147229>.
- [58] S. Akkermans, B. Crispo, W. Joosen, D. 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, ACM, New York, NY, USA, 2018, pp. 59–68, <http://dx.doi.org/10.1145/3286978.3286997>.
- [59] D. Guija, M.S. 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, ACM, New York, NY, USA, 2018, pp. 46:1–46:10, <http://dx.doi.org/10.1145/3230833.3233255>.
- [60] X. Li, Y. Chen, Z. 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, ACM, New York, NY, USA, 2019, pp. 3–5, <http://dx.doi.org/10.1145/3342280.3342288>.
- [61] G. Márquez, H. 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, ACM, New York, NY, USA, 2019, pp. 123–129, <http://dx.doi.org/10.1145/3344948.3344996>.
- [62] A. Ibrahim, S. Bozhinoski, A. Pretschner, Attack graph generation for microservice architecture, in: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC '19, ACM, New York, NY, USA, 2019, pp. 1235–1242, <http://dx.doi.org/10.1145/3297280.3297401>.
- [63] D.M. Stallenberg, A. Panichella, Jcomix: A search-based tool to detect xml injection vulnerabilities in web applications, in: Proceedings of the 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE 2019, ACM, New York, NY, USA, 2019, pp. 1090–1094, <http://dx.doi.org/10.1145/3338906.3341178>.
- [64] M. Krämer, S. Frese, A. Kuijper, Implementing secure applications in smart city clouds using microservices, *Future Gener. Comput. Syst.* 99 (2019) 308–320, <http://dx.doi.org/10.1016/j.future.2019.04.042>.
- [65] K. Jander, L. Braubach, A. Pokahr, Defense-in-depth and role authentication for microservice systems, *Procedia Comput. Sci.* 130 (2018) 456–463, 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. <http://dx.doi.org/10.1016/j.procs.2018.04.047>.
- [66] S. Abidi, M. Essafi, C.G. Guegan, M. Fakhri, H. Witt, H.H.B. Ghezala, A web service security governance approach based on dedicated micro-services, in: Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 23rd International Conference KES2019, *Procedia Comput. Sci.* 159 (2019) 372–386, <http://dx.doi.org/10.1016/j.procs.2019.09.192>.
- [67] M. Elsayed, M. Zulkernine, Offering security diagnosis as a service for cloud saas applications, *J. Inf. Secur. Appl.* 44 (2019) 32–48, <http://dx.doi.org/10.1016/j.jisa.2018.11.006>.
- [68] V. Mavroudis, A. Cerulli, P. Svenda, D. Cvrcek, D. Klinec, G. 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, Association for Computing Machinery, New York, NY, USA, 2017, pp. 1583–1600, <http://dx.doi.org/10.1145/3133956.3133961>.
- [69] A.P. Vale, E.B. Fernandez, An ontology for security patterns, in: 2019 38th International Conference of the Chilean Computer Science Society (SCCC), 2019, pp. 1–8, <http://dx.doi.org/10.1109/SCCC49216.2019.8966393>.
- [70] IBM, An integrated approach to insider threat protection, 2016, URL <https://www.ibm.com/security/services/insider-threat-protection>.
- [71] J. Kindervag, S. Balaouras, K. Mak, Build Security Into Your Network's Dna: The Zero Trust Network Architecture, Tech. rep. Forrester Research, 2012.
- [72] R. Zhuang, S.A. DeLoach, X. Ou, Towards a theory of moving target defense, in: Proceedings of the First ACM Workshop on Moving Target Defense, MTD '14, Association for Computing Machinery, New York, NY, USA, 2014, pp. 31–40, <http://dx.doi.org/10.1145/2663474.2663479>.
- [73] D. Merkel, Docker: lightweight linux containers for consistent development and deployment, *Linux J.* 2014 (239) (2014) 2.