The Journal of Systems and Software 230 (2025) 112558

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

Containerization in multi-cloud environment: Roles, strategies, challenges, and solutions for effective implementation ☆

Muhammad Waseem [a,*], Aakash Ahmad [b], Peng Liang [c], Muhammad Azeem Akbar [d], Arif Ali Khan [e], Iftikhar Ahmad [f], Manu Setälä [g], Tommi Mikkonen [h]

[a] Faculty of Information Technology and Communication Sciences, Tampere University, Tampere, Finland
[b] School of Computing and Communications, Lancaster University Leipzig, Leipzig, Germany
[c] School of Computer Science, Wuhan University, Wuhan, China
[d] Software Engineering Department, Lappeenranta-Lahti University of Technology, Lappeenranta, Finland
[e] M3S Empirical Software Engineering Research Unit, University of Oulu, Oulu, Finland
[f] TietoEVRY Oy, Tampere, Finland
[g] Solita Oy, Tampere, Finland
[h] Faculty of Information Technology, University of Jyväskylä, Jyväskylä, Finland

ARTICLE INFO

Dataset link: https://zenodo.org/records/14692878

Keywords:
Containerization
Multi-cloud environment
Systematic mapping study
Cloud computing

ABSTRACT

Containerization in multi-cloud environments has received significant attention in recent years both from academic research and industrial development perspectives. However, there exists no effort to systematically investigate the state of research on this topic. The aim of this research is to systematically identify and categorize the multiple aspects of containerization in multi-cloud environment. We conducted the Systematic Mapping Study (SMS) on the literature published between January 2013 and July 2024. One hundred twenty one studies were selected and the key results are: (1) Four leading themes on containerization in multi-cloud environment are identified: 'Scalability and High Availability', 'Performance and Optimization', 'Security and Privacy', and 'Multi-Cloud Container Monitoring and Adaptation'. (2) Ninety-eight patterns and strategies for containerization in multi-cloud environment were classified across 10 subcategories and 4 categories. (3) Ten quality attributes considered were identified with 47 associated tactics. (4) Four catalogs consisting of challenges and solutions related to security, automation, deployment, and monitoring were introduced. The results of this SMS will assist researchers and practitioners in pursuing further studies on containerization in multi-cloud environment and developing specialized solutions for containerization applications in multi-cloud environment.

Contents

1. Introduction 2
2. Methodology 4
 2.1. Research questions 4
 2.2. Search string composition 4
 2.3. Study search and selection process 4
  2.3.1. Primary search 4
  2.3.2. Snowballing 6
  2.3.3. Quality assessment of selected studies 6
 2.4. Data extraction and analysis 7
3. Results 7
 3.1. Demographics, research aims, and contributions 7

☆ Editor: Dr. Dario Di Nucci.

* Corresponding author.
E-mail addresses: muhammad.waseem@tuni.fi (M. Waseem), a.ahmad13@lancaster.ac.uk (A. Ahmad), liangp@whu.edu.cn (P. Liang), azeem.akbar@lut.fi (M.A. Akbar), Arif.Khan@oulu.fi (A.A. Khan), iftikhar.ahmad@tietoevry.com (I. Ahmad), manu.setala@solita.fi (M. Setälä), tommi.j.mikkonen@jyu.fi (T. Mikkonen).

https://doi.org/10.1016/j.jss.2025.112558
Received 21 February 2025; Received in revised form 27 May 2025; Accepted 1 July 2025
Available online 24 July 2025

3.2. Container roles and implementation strategies (RQ1) 10
3.2.1. Container roles 10
3.2.2. Implementation strategies 12
3.3. Pattern and strategies (RQ2) 14
3.4. Quality attributes and tactics (RQ3) 17
3.5. Security challenges and solution framework (RQ4 and RQ5) 19
3.6. Automation challenges and solution framework (RQ4 and RQ5) 21
3.7. Deployment challenges and solution framework (RQ4 and RQ5) 24
3.8. Monitoring challenges and solution framework (RQ4 and RQ5) 27
3.9. Tools and frameworks (RQ6) 29
4. Discussion 30
4.1. Containers in multi-cloud: Roles and strategies (RQ1) 30
4.2. Patterns and strategies (RQ2) 31
4.3. Quality attributes and tactics (RQ3) 31
4.4. Automation challenges and solution framework (RQ4–RQ5) 32
4.5. Security challenges and solutions framework (RQ4 - RQ5) 33
4.6. Deployment challenges and solutions framework (RQ4 - RQ5) 33
4.7. Monitoring challenges and solutions (RQ4 - RQ5) 34
4.8. Tools and frameworks (RQ6) 35
5. Threats to validity 35
5.1. Internal validity 35
5.2. External validity 36
5.3. Construct validity 36
5.4. Conclusion validity 36
6. Related work 36
6.1. Challenges, solutions, and performance of multi-cloud containers 36
6.1.1. Challenges, applications, tools, and emerging trends 36
6.1.2. Performance and resource utilization 36
6.2. Container-based cloud orchestration, deployment, and security 37
6.2.1. Orchestration and deployment 37
6.2.2. Security of containerized clouds 37
6.3. Conclusive summary 37
7. Conclusions 38
CRediT authorship contribution statement 39
Declaration of AI Assistance 39
Declaration of competing interest 39
Acknowledgments 39
Data availability 39
References 39

1. Introduction

The use of containers in multi-cloud environment has been widespread in the industry for many years (Naik, 2016b). Containers are standalone and executable packages of software that include everything needed to run an application, such as code, system tools, libraries, and settings (Merkel et al., 2014). Containers allow the packaging of an application and its required dependencies, which makes it easy to move the application across different environments with minimal modification (Docker, 2021). On the other hand, multi-cloud environment is the distribution of cloud assets (Petcu, 2013). Using containers in multi-cloud environment allows organizations to achieve flexibility, agility, and cost-efficiency. Developers can build applications in a consistent environment and easily move them between multiple cloud platforms, thereby leveraging the unique strengths (e.g., extensive infrastructure, seamless integration, advanced data analytic) of each platform.

Containerization in multi-cloud is illustrated in Fig. 1. This figure provides an overview of applications deployed within various cloud configurations, including public, private, and hybrid models. Each cloud hosts multiple applications encapsulated in containers, which can support different types of applications, such as those in healthcare or finance, that require flexibility in deployment and scalability across cloud providers. These containers complete various tasks with their required binaries and libraries, emphasizing the portability and isolation that containerization provides. A container platform layer, which could be exemplified by systems like Kubernetes or Docker, manages and orchestrates these containers across the different cloud environments.

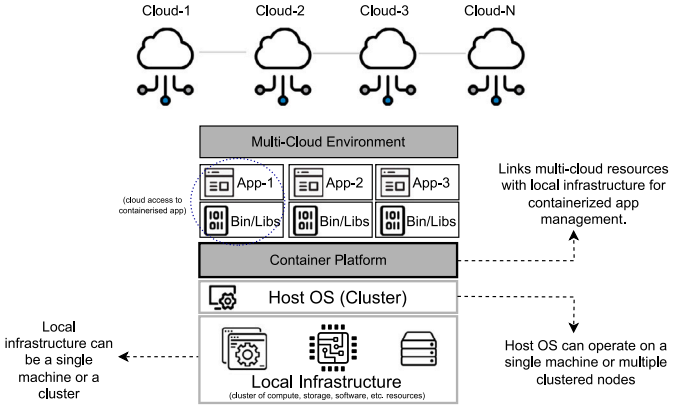

Fig. 1. Context: Containerization in multi-clouds.

For instance, a healthcare application can use a private cloud for processing sensitive data for compliance and a public cloud for data analysis to achieve cost-efficiency and scalability. This architecture enables a flexible and scalable approach, allowing efficient application deployment and operations in a multi-cloud setting, while maintaining consistency and resilience across platforms.

However, utilizing containerization in multi-cloud environment is not without challenges that organizations may face (Saif et al., 2022). These challenges may arise during different phases and activities of container-based application development including architectural design phase, system implementation, or when establishing an auto-

mated development infrastructure. Furthermore, the challenges can also present themselves during system testing, the coding process, and the deployment phase. For instance, during the architectural design phase, challenges may include ensuring seamless integration of containerized components with existing systems and selecting appropriate container orchestration patterns, strategies, and tools that can effectively manage containers across multiple clouds (Raj et al., 2018; Baby and Vysala, 2015), and while establishing an automated development infrastructure, organizations may encounter difficulties in creating efficient Continuous Integration/Continuous Deployment (CI/CD) pipelines that handle multi-cloud deployment scenarios while maintaining consistent performance and security standards (Debroy et al., 2018). Similarly, in the system implementation phase, challenges may include resolving potential compatibility issues between containers and various cloud platforms, as well as optimizing resource utilization to control costs effectively (Helali and Omri, 2021).

Motivation: Our study is part of the QLEAP project (2022–2024) funded by Business Finland, which investigates the use of containers in multi-cloud environment specifically for architecture design (Ahmad et al., 2025). The project brings together a consortium of four companies (Bittium,[1] M-Files,[2] Solita,[3] and Vaadin[4]), each with distinct requirements for containerization, and is further supported by industry leaders Nokia[5] and TietoEVRY.[6] This study fulfills an industry demand for optimized container strategies, addressing critical gaps in deployment consistency, security, and scalability that are necessary for practical applications in multi-cloud contexts. Despite the growing interest in containerization for multi-cloud environment, there remains a significant gap in systematically organized knowledge to address these challenges comprehensively. Our study not only fulfills an immediate industry demand but also aims to provide a structured view of containerization practices, addressing the needs of both practitioners and researchers in navigating this fragmented knowledge space. Recent research (e.g., see Selected Studies sheet in Waseem et al. (2025)) has highlighted the growing significance of container utilization in multi-cloud environment. These studies have explored various aspects, including container roles and strategies, architectural patterns, Quality Attributes (QAs), and the tools and frameworks employed. Additionally, these studies have explored the challenges associated with automation, deployment, monitoring, and security of containerization applications in multi-cloud environment. Despite the breadth of knowledge available, such valuable information is dispersed across different publications, spanning scientific research papers and gray literature. This fragmentation of knowledge about containerization applications in multi-cloud environment presents a significant navigational challenge for practitioners tasked with real-world implementations. They need to thoroughly scrutinize numerous aspects, such as patterns and strategies for containerization applications in multi-cloud environment, to locate specific information relevant to their use cases, whether it be a pattern, a challenge, or a solution to an existing problem. Consequently, this spreading of knowledge restricts the development of a holistic understanding of the containerization applications in multi-cloud environment, leaving practitioners underprepared to devise comprehensive, secure solutions.

Furthermore, this state of affairs of containerization in multi-cloud also poses challenges for the academic community. Researchers bear the obligation of navigating through this vast array of information to uncover the precise aspects they seek — be it a unique pattern, an unexplored challenge, or an innovative solution. Moreover, the distributed nature of this knowledge delays the formation of a cohesive and unified understanding of the subject matter. Recent studies (e.g., Bordeleau et al. (2020), Mihai et al. (2022) and Sarker (2022)) have notably highlighted that challenges in design, development, monitoring, and testing of containerization applications in multi-cloud environment are deeply interconnected with the software development life cycle.

To bridge this knowledge gap and support the practical demands of the QLEAP project consortium, our study systematically identifies and categorizes the various facets of container utilization in multi-cloud environment. These facets include container roles, implementation strategies, architectural patterns, and quality attributes. In addition, we explore the challenges and solutions associated with automation, deployment, monitoring, and security, providing practitioners and researchers with actionable insights to tackle these critical issues effectively. To achieve this objective, we employed a systematic approach to conduct a mapping study. We finally selected 86 studies that focus on containerized applications in multi-cloud environment for further analysis. We adopted the following Goal-Question-Metric (GQM) framework (Caldiera and Rombach, 1994):

- **Goal**:To enhance the understanding and efficiency of deploying and managing container-based applications in multi-cloud environment.
- **Questions**:To explore specific aspects such as container roles, implementation strategies, architectural patterns, quality attributes, challenges, and solutions.
- **Metrics**:To evaluate the effectiveness of our findings based on the identified roles, strategies, patterns, attributes, challenges, solutions, and tools.

The empirical findings and challenge-solution catalogs from this Systematic Mapping Study (SMS) provide significant benefits for developers of container-based applications. By examining implementation strategies, architectural patterns, and quality attributes, practitioners gain insights to improve practices. Detailed analyses of challenges and solutions in automation, deployment, monitoring, and security equip practitioners to address issues and enhance quality attributes. These challenge-solution catalogs facilitate knowledge sharing, training, and a shared understanding of containerized applications in multi-cloud environments, empowering developers to optimize their approaches and improve efficiency and quality. In response to the real-world needs identified in the QLEAP project and the broader research gap, our SMS presents the following **key contributions**:

- We systematically identified and categorized challenges and their solutions related to security, automation, deployment, and monitoring for containerization in a multi-cloud environment.
- We systematically identified and categorized an extensive list of patterns and strategies for container-based applications in multi-cloud environment.
- We systematically identified and categorized quality attributes and tactics for containerized applications in multi-cloud environment.
- We systematically identified and classified a wide range of tools and frameworks for containerized applications in multi-cloud environment.
- We have publicly released a dataset, which is available online (Waseem et al., 2025), to enable researchers and practitioners to access, replicate, and validate all collected data from our SMS. This dataset includes detailed hierarchies of the developed catalogs, identified and classified container implementation strategies, and the roles of containers in multi-cloud environment. It also encompasses patterns, strategies, quality attributes, tactics, tools, and frameworks for containerized-based applications in multi-cloud environment.

[1] https://www.bittium.com/.
[2] https://www.m-files.com/.
[3] https://www.solita.fi/en/.
[4] https://vaadin.com/.
[5] https://www.nokia.com/.
[6] https://www.tietoevry.com/.

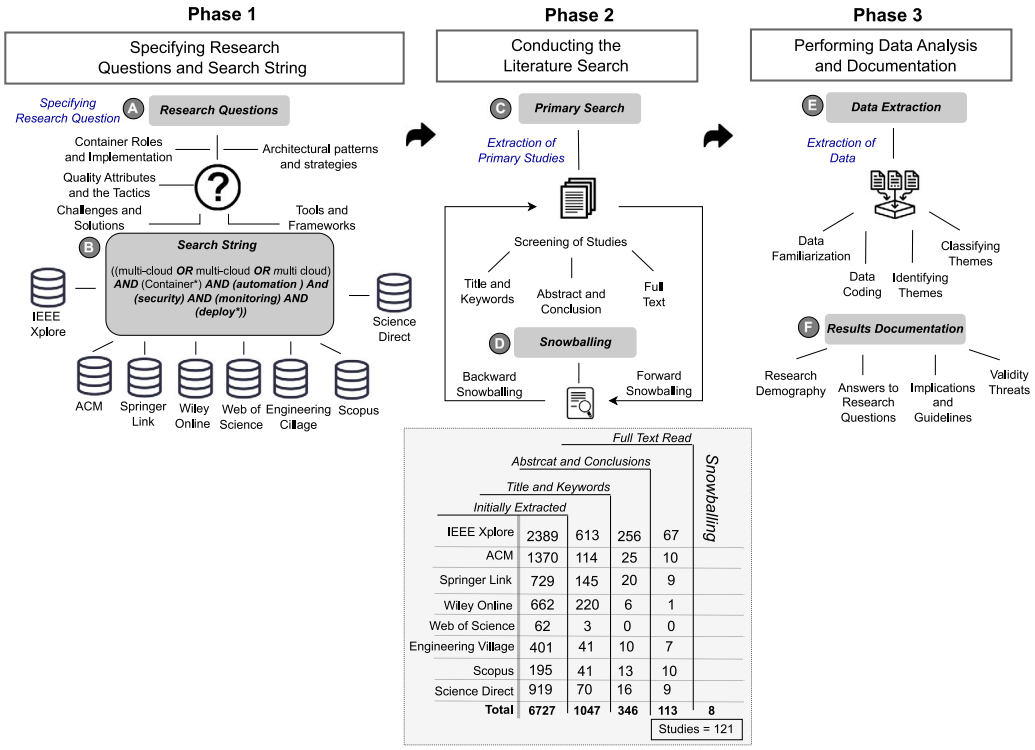

Fig. 2. Schematic representation of the research methodology implemented in this study.

The paper is structured as follows: Section 2 describes the research methodology employed. Section 3 presents the results and findings of this SMS. Section 4 discusses the implications of the SMS results. Section 5 clarifies the potential threats to the validity of this SMS. Section 6 reviews relevant prior work and highlights the differences between our work and previous research. Section 7 draws conclusions and outlines future research directions.

2. Methodology

We conducted a Systematic Mapping Study (SMS) by following the guidelines in Petersen et al. (2008) and augmenting them with SLR strategies (Keele et al., 2007). Our SMS consists of three phases: specifying research questions and search string, conducting the literature search, and performing data analysis and documentation. Fig. 2 illustrates the SMS execution process. Reliability and consistency are critical aspects of any empirical study. In this study, we followed established statistical approaches, including inter-rater agreement (IRA) computation, to ensure the reliability of our data coding and extraction processes. Additionally, we applied systematic quality assessment techniques to evaluate the studies included in our SMS.

2.1. Research questions

To conduct this SMS, we formulated six Research Questions (RQs) based on the GQM approach presented in the Introduction section (see Section 1) aligns with the objective of our SMS, as outlined in Table 1. The table provides a structured view of the RQs with their corresponding rationale and categorization.

These RQs are designed to address important gaps in the current understanding of container-based applications in multi-cloud environment. RQ1–RQ3 focus on understanding the roles of containers, implementation strategies, architectural patterns, and quality attributes. These questions aim to provide a comprehensive perspective, building on and organizing fragmented insights from prior studies. RQ4 and RQ5 focus on specific challenges and corresponding solutions related to security, automation, deployment, and monitoring in container-based multi-cloud environment. These questions aim to systematically address issues that are often only partially explored in existing works. Finally, RQ6 investigates tools and frameworks used for implementation, helping bridge the gap between academic research and practical applications. Together, these RQs provide a structured framework that enhances understanding and supports the development and management of container-based applications in multi-cloud environment, advancing the state-of-the-art in this domain

2.2. Search string composition

Initially, we considered the PICO (Population, Intervention, Comparison, Outcome) framework (Schardt et al., 2007; Ralph et al., 2020) for constructing our search string. However, given the complexity and breadth of our topic, applying PICO strictly required an extensive array of terms and logical operators, exceeding the query limits of databases like ScienceDirect. Therefore, we developed the search string for this study based on authors' knowledge and iterative trial searches. This approach allowed us to tailor the search strategy to the specific goal of our study, ensuring comprehensive coverage within practical constraints.

2.3. Study search and selection process

The study search and selection process for this study is divided into two phases. In the first phase, we conducted a primary search using a specific search string on various databases (see Table 2 and the Summary sheet in Waseem et al. (2025)). In the second phase, we applied the snowballing technique on primary studies that were selected from the first phase.

2.3.1. Primary search

The primary search involved querying digital databases (see Table 2) using customized search string. The search strings were executed concurrently on eight databases, as illustrated in Fig. 2, spanning between January 2013 and July 2024 when we started this SMS. We

Table 1
Research questions and their rationale.

#	Research questions	Rationale
Container roles and implementation		
RQ1	What roles do containers play, and what strategies can be employed for implementing container-based applications in multi-cloud environment?	To investigate the role of containers and explore implementation strategies in multi-cloud environment, with the objective of enhancing understanding and facilitating efficient deployment and management of applications across multiple cloud environments.
RQ2	What architectural patterns and strategies are utilized in the implementation of container-based applications in multi-cloud environment?	To explore and document the various architectural patterns and strategies that are utilized in implementing container-based applications in multi-cloud environment, thus providing a roadmap for efficient and scalable application development and deployment.
RQ3	What are the quality attributes and the tactics associated with their implementation in container-based applications in multi-cloud environment?	To examine and analyze the quality attributes required for container-based applications in multi-cloud environment and the corresponding tactics employed in their implementation, enabling the identification of best practices and guidelines for developing robust and reliable applications in such environments.
Challenges and solutions		
RQ4	Challenges in container-based applications in multi-cloud environment • **RQ4.1**:What are the challenges related to security in container-based applications in multi-cloud environment? • **RQ4.2**: What are the challenges related to automation in container-based applications in a multi-cloud environment? • **RQ4.3**: What are the challenges related to deployment in container-based applications in a multi-cloud environment? • **RQ4.4**: What are the challenges related to monitoring in container-based applications in a multi-cloud environment?	To comprehensively evaluate and gain insights into the challenges surrounding security (RQ4.1), automation (RQ4.2), deployment (RQ4.3), and monitoring (RQ4.4) concerning container-based applications in multi-cloud environment. These sub-RQs aim to provide valuable insights into the obstacles and intricacies that organizations encounter while implementing and managing container-based applications in a multi-cloud environment.
RQ5	Solutions to address challenges in container-based applications in multi-cloud environment • **RQ5.1**:How to address the challenges related to security in container-based applications within a multi-cloud environment? • **RQ5.2**: How to address the challenges related to automation in container-based applications within a multi-cloud environment? • **RQ5.3**: How to address the challenges related to deployment in container-based applications within a multi-cloud environment? • **RQ5.4**: How to address the challenges related to monitoring in container-based applications within a multi-cloud environment?	These sub-RQs seek to investigate solutions aimed at tackling the challenges of security (RQ5.1), automation (RQ5.2), deployment (RQ5.3), and monitoring (RQ5.4) within container-based applications operating in multi-cloud environment. By providing actionable guidance and recommendations, the goal is to elevate the overall management and performance of these applications in the context of multi-cloud environment.
Tools and frameworks		
RQ6	What tools and frameworks are used to implement container-based multi-cloud applications?	To investigate and identify the specific tools and frameworks utilized in the implementation of container-based multi-cloud applications, providing an overview of the technological landscape and aiding developers and organizations in making informed decisions when selecting appropriate tools for their application deployment and management.

Table 2
Search string and targeted search area for databases.

Search string	
((multicloud OR multi-cloud OR multi cloud) AND (container) AND (automation) And (security) AND (monitoring) AND (deploy*))*	
Databases	
Database	Targeted search area
ACM Digital Library	Paper title, abstract
IEEE Xplore	Paper title, keywords, abstract
Scopus	Paper title, keywords, abstract
SpringerLink	Paper title, abstract
ScienceDirect	Paper title, keywords, abstract
Wiley Online Library	Paper title, abstract
Engineering Village	Paper title, abstract
Web of Science	Paper title, keywords, abstract

chose 2013 as the starting point because it is the year when container technologies like Docker emerged (Docker, 2023), marking a key phase in containerization and cloud computing. This period also saw a rise in relevant research to containers in multi-cloud environment, making it ideal to understand the development of containerized applications in multi-cloud environment over the past decade. We followed the steps mentioned below to select the relevant studies.

- *Step 1: Extraction of Studies*: We ran custom search strings in the selected databases (see Table 2) to retrieve study titles, author names, publication years, venues, publication types, and abstracts. This initial search yielded 6727 studies from eight databases. Following initial retrieval, we did not apply Cohen's Kappa (Cohen, 1960, 1968) as this step is automated and does not involve subjective decision-making.
- *Step 2: Title and Keyword Screening*: This step involves reviewing the titles and keywords of the collected studies to assess their relevance to the research topic. To ensure alignment with our inclusion criteria, we manually reviewed study titles and abstracts to include only those that explicitly discuss containerization within multi-cloud contexts. Studies that focused solely on single-cloud containerization, general cloud orchestration without containers, or container use cases unrelated to multi-cloud environments were excluded—even if they matched keywords in the search string. Initially, we removed any duplicate studies obtained from different databases (e.g., IEEE Xplore, Scopus) by sorting them in ascending order. This resulted in the elimination of several thousand duplicate studies. We divided the remaining studies between two authors (R1, R2), who independently applied the inclusion and exclusion criteria to identify studies relevant to our research objective, as detailed in Table 3. To assess the consistency of study selection and mitigate bias, a Cohen's Kappa analysis (Cohen, 1960, 1968) was performed on a random sample of 400 studies (10% of the total dataset of 4000 studies), as illustrated in Formula (1):

Table 3
Inclusion and exclusion criteria for selecting primary studies in this SMS.

Selection criteria	Inclusion criteria	Exclusion criteria
Language	English	Non-English
Study type	Primary studies including peer-reviewed journal articles, book chapters, conference papers, workshops, and symposium papers	Secondary studies or non-peer-reviewed content (e.g., blogs, webpages, videos, white papers, technical reports, gray literature)
Study focus	Studies that explicitly address the intersection of containerization and multi-cloud environments. This includes studies that investigate the use of container technologies (e.g., Docker, Kubernetes) in the context of multi-cloud deployment, management, or architecture. Studies that focus solely on containerization without a multi-cloud context, or vice versa, are excluded.	Studies that address only traditional cloud computing, single-cloud environments, or containerization without reference to multi-cloud contexts, and vice versa. Specifically, studies that examine container technologies without situating them in multi-cloud scenarios, or studies focused on multi-cloud architectures without discussing container-based solutions, are excluded.
Study duration	Studies published between January 2013 and July 2024	Studies published before January 2013 or after July 2024

$$\kappa = \frac{P_o - P_e}{1 - P_e} \quad (1)$$

where P_o represents the relative observed agreement among authors, and P_e the hypothetical probability of chance agreement. The results, recorded in Table 4.a, indicated a Cohen's Kappa of $\kappa = 0.531$, which reflects moderate agreement between R1 and R2 (Sim and Wright, 2005). Specifically, R1 and R2 agreed on 360 studies being relevant ("YES") and 20 studies being irrelevant ("NO"). Disagreements occurred in 20 studies, which were systematically resolved through discussions among all authors, leading to a final consensus. Using a random subset is a common practice in systematic reviews to measure inter-rater agreement without overburdening resources, ensuring reliability while maintaining efficiency (Keele et al., 2007; Chandler et al., 2019). Following this process, the number of studies was reduced to 1047 for further analysis.

- *Step 3: Abstract-based Screening*: During this step, we conducted a thorough review of the abstracts of the collected studies to determine their relevance to our research topic. Two researchers (R1 and R2) independently examined each abstract, assigning a status of "relevant", "irrelevant", or "doubtful" with respect to our research goals. To reduce subjective bias and to quantify the consensus between the researchers, we performed a Cohen's Kappa analysis on a random subset of 105 studies (approximately 10% of the 1047 studies remaining after the Title and Keyword Screening phase).
  The outcomes of this assessment are depicted in Table 4.b. The analysis showed substantial agreement between R1 and R2, with 88 studies being unanimously considered relevant and 5 studies unanimously considered irrelevant. Disagreements occurred in 12 cases (7 studies where one researcher marked "relevant" while the other marked "irrelevant", and 5 studies initially classified as "doubtful"). The kappa value, calculated based on observed agreements and disagreements, was $\kappa = 0.521$, calculated based on observed agreements and disagreements, indicates moderate agreement (Sim and Wright, 2005; Keele et al., 2007).
  To compute Cohen's Kappa during the abstract-based screening phase, the original classifications of "relevant", "irrelevant", and "doubtful" were simplified into a binary scale ("YES" or "NO") due to the binary nature of Cohen's Kappa computation. Specifically:
  - Studies classified as "relevant" were mapped to "YES".
  - Studies classified as "irrelevant" were mapped to "NO".
  - Studies classified as "doubtful" were resolved through discussions between R1 and R2. If consensus was reached on "relevant", the study was marked as "YES"; otherwise, it was marked as "NO".

  Using subsets for kappa computation ensures that inter-rater reliability can be rigorously evaluated while optimizing the use of resources, as recommended by systematic review guidelines (Keele et al., 2007; Chandler et al., 2019). After resolving all disagreements through systematic discussions among the authors, a final consensus was reached for the subset. Consequently, upon completing this step, the number of studies was reduced to 346 for more detailed scrutiny.
- *Step 4: Full-Text Screening*: In this step, we reviewed the full texts of 346 studies that had been shortlisted through abstract-based screening. To ensure consistency in our review process, we randomly selected a subset of 35 studies (10% of the total 346 studies) for a Cohen's Kappa analysis to measure the agreement between two researchers (R1 and R2). The results, recorded in Table 4.c, indicated a Cohen's Kappa value of $\kappa = 0.578$, which represents moderate agreement between the two researchers (Sim and Wright, 2005). Specifically, R1 and R2 agreed on 30 studies as relevant ("YES") and 2 studies as irrelevant ("NO"). Disagreements occurred in 3 studies (2 marked as "YES" by one researcher and "NO" by the other, and 1 marked as "NO" by both but flagged as uncertain). All disagreements were systematically resolved through discussions among the authors, leading to a final consensus. This subset-based approach, coupled with systematic resolution of disagreements, provides confidence in the reliability of the full-text screening process (Chandler et al., 2019). At the end of this step, we obtained a total of 110 studies, which were deemed relevant for further analysis.

2.3.2. Snowballing

In Phase 2, we utilized the snowballing technique, as described in Wohlin (2014), to examine references within 110 primary studies to identify additional relevant studies. This strategy was augmented by forward snowballing, where we gathered studies that cited the selected studies, and backward snowballing, which involved using references within the selected studies. Notably, we encountered several dozen studies during the forward and backward snowballing that had been excluded in primary search. This phase resulted in the addition of 11 more studies, bringing the final count to 121.

2.3.3. Quality assessment of selected studies

We assessed the quality of the selected studies based on a set of predefined criteria, including relevance to multi-cloud containerization, clarity of research design, empirical validation (e.g., experiments, case studies), reporting quality (e.g., discussion of limitations and challenges), publication in peer-reviewed venues, and the generalizability of findings to multi-cloud environment. Each study was assessed using six predefined quality criteria: relevance to multi-cloud containerization, clarity of research design, data validity (e.g., empirical evidence), reporting quality (e.g., discussion of limitations), peer-review status, and generalizability of findings. Each criterion was scored using a 3-point scale: 1 (fully satisfied), 0.5 (partially satisfied), or 0 (not satisfied), giving a total score out of 6. The scoring was applied consistently across all studies based on observable features. For example, studies addressing both containerization and multi-cloud contexts were

Table 4
Kappa agreement across different screening steps.

(a) Title & Keyword screening (400 Studies)			(b) Abstract-based screening (105 Studies)			(c) Full-text screening (35 Studies)		
	R1 Yes	R1 No	R1 Yes	R1 No		R1 Yes	R1 No	
R2 Yes	360	20	R2 Yes	88	7	R2 Yes	30	3
R2 No	10	10	R2 No	5	5	R2 No	2	0
Total	370	30	Total	93	12	Total	32	3

scored 1 for relevance; studies with explicit research methods scored higher on research design; and those lacking empirical evidence were scored lower for data validity. Scores for all individual criteria are documented in the "QA of Selected Studies" sheet in our replication package, providing transparency and allowing independent verification of the quality assessment process. Studies scoring below 3 were flagged for further review and discussion but retained to ensure alignment with the study's objectives. This quality assessment was conducted after the study selection process to ensure consistency and thoroughness across all 121 studies. Detailed results are available in the replication package (Waseem et al., 2025), specifically in the sheet titled "QA of Selected Studies".

2.4. Data extraction and analysis

Data Extraction: The data extraction form was designed based on predefined data items (see Table 5) that were formulated to address the RQs specified in Table 1. To ensure the reliability of the extracted data items, the pilot data extraction was conducted on ten studies by the first author, and all the other authors assessed the extracted data. Subsequently, the first author employed a revised set of data items (e.g., D11, D12, D16), determined after evaluating the extracted data items, for the formal data extraction from the selected studies. To mitigate the personal bias and ambiguity, all authors engaged in discussions regarding the extracted data. The data items labeled as D1 to D8 present a summary of the demographics of the primary studies selected, while D9 to D22 are specifically employed to address RQ1 to RQ6. A concise description of each data item is presented in Table 5. Finally, Google Sheets were used to record and further analyze the extracted data.

Data Analysis: We employed descriptive statistics to analyze the quantitative data from data items D1, D5, D7, D8, and D9. For the remainder of the data, which primarily consist of qualitative free-text descriptions (e.g., study aim, roles of containers, challenges, and solutions), we conducted a thematic analysis in accordance with the guidelines outlined in Braun and Clarke (2006). Our thematic analysis process consists of the following steps:

1. **Data Familiarization**: We conducted a thorough review of the selected studies by repeatedly reading through them and meticulously noting key points related to study aims (D7), contributions (D8), roles of containers (D9), implementation strategies (D10), architectural patterns (D11), quality attributes (D12), tactics (D13), motivations (D14), and challenges and solutions (D15–D21) in automation, deployment, monitoring, and security challenges and solutions, as well as the tools, languages, and frameworks (D22) employed.
2. **Generation of Initial Codes**: After developing a thorough understanding of the data, we created an initial set of codes derived from the information extracted concerning the data items identified in the previous step.
3. **Identification of Types and Emerging Themes**: In this step, we conducted a two-tiered analysis. Initially, we examined the codes to ascertain their types. Subsequently, we developed subcategories based on these types, and then formulated overarching categories that encompass the related subcategories.
4. **Critical Evaluation of Types, Subcategories, and Categories**: All authors actively participated in rigorously reviewing and refining the coded data, including types, subcategories, and categories. During this collaborative process, we redefined, merged, or dropped certain themes based on collective input and discussion.
5. **Defining and Naming Categories**: At this point, we provided explicit definitions for each of the identified themes and further refined them, ensuring that the terminology used for the categories was precise and unambiguous.

Two researchers participated in the process to reduce personal bias. The most important activity of this process is brainstorming sessions that were mainly conducted while reviewing, defining, and naming the research themes. In these sessions, both researchers discussed and validated the research themes found.

Concerning data item D9 (Roles of Containers) and D10 (Container Implementation strategy), we used the open coding and constant comparison techniques from Grounded Theory (Glaser and Strauss, 1967) to analyze the qualitative data extracted from the selected studies.

Finally, we provided a replication package (Waseem et al., 2025) with the results of each phase of the study selection process (e.g., Phase 1, Phase 2) and detailed results (e.g., Contributions, Patterns, QAs, Challenges, Solutions) for verification and validation purposes of this SMS.

3. Results

This section presents the results of the SMS, derived from analyzing data from chosen studies. Section 3.1 provides a summary of the demographic data, classifications, and the mapping of research themes identified in the studies. Section 3.2 presents the results concerning container roles and implementation strategies. Section 3.3 reports on the patterns and strategies employed in container-based applications in multi-cloud environment. Section 3.4 presents the QAs and related tactics for containerized applications in multi-cloud environment. Section 3.5 presents the security challenges and solutions framework, while Section 3.6 presents the automation challenges and solutions framework. Section 3.7 presents the deployment challenges and solutions framework, and Section 3.8 presents the monitoring challenges and solutions framework. Finally, the tools and frameworks are reported in Section 3.9.

3.1. Demographics, research aims, and contributions

Yearly Distribution of Studies: Fig. 3-a illustrates the annual distribution of studies retrieved from eight databases between January 2013 and July 2024. We found only two primary studies on containerization in multi-cloud environment that meet our criteria in 2013. The bars display yearly study counts and trends. A peak in 2017, followed by a decrease and occasional fluctuations, suggests an early spike in interest followed by more targeted research. Notably, the reduction in publication numbers in 2020 and 2021 may be partially attributed to the global disruptions caused by the COVID-19 pandemic, which affected research outputs across various fields. However, there is a notable increase in studies in 2023 and continuing into 2024. This could be due to delayed research projects getting back on track and

Table 5
Data items to be extracted in this SMS.

Code	Data item	Description	RQ
D1	Index	The ID of the study	Demographics
D2	Publication Year	Publication year of the study	
D3	Publisher	The publisher of the study	
D4	Venue	The name of the publishing venue	
D5	Publication Type	Journal, conference, workshop, and book chapter	
D6	Authors' Affiliation	The affiliation of the authors	
D7	Study Aim	The aim of the paper	
D8	Study Contribution	The contributions made by the paper	
D9	Role of Containers	Role of containers in the study	RQ1
D10	Container Implementation strategy	Strategy used for container implementation	
D11	Architectural Pattern and Strategies	Architectural patterns and strategies reported in the study	RQ2
D12	Quality Attributes	Quality attributes discussed in the study	RQ3
D13	Tactics	Tactics used in the study	
D14	Automation Challenges	Challenges in automation discussed in the study	RQ4.2
D15	Automation Solutions	Solutions for automation challenges discussed in the study	RQ5.2
D16	Deployment Challenges	Challenges in deployment discussed in the study	RQ4
D17	Deployment Solutions	Solutions for deployment challenges discussed in the study	RQ5
D18	Monitoring Challenges	Challenges in monitoring discussed in the study	RQ4
D19	Monitoring Solutions	Solutions for monitoring challenges discussed in the study	RQ6
D20	Security Challenges	Challenges in security discussed in the study	RQ4,1
D21	Security Solutions	Solutions for security challenges discussed in the study	RQ5.1
D22	Tools, Languages, and Frameworks	Tools, programming languages, and frameworks	RQ6

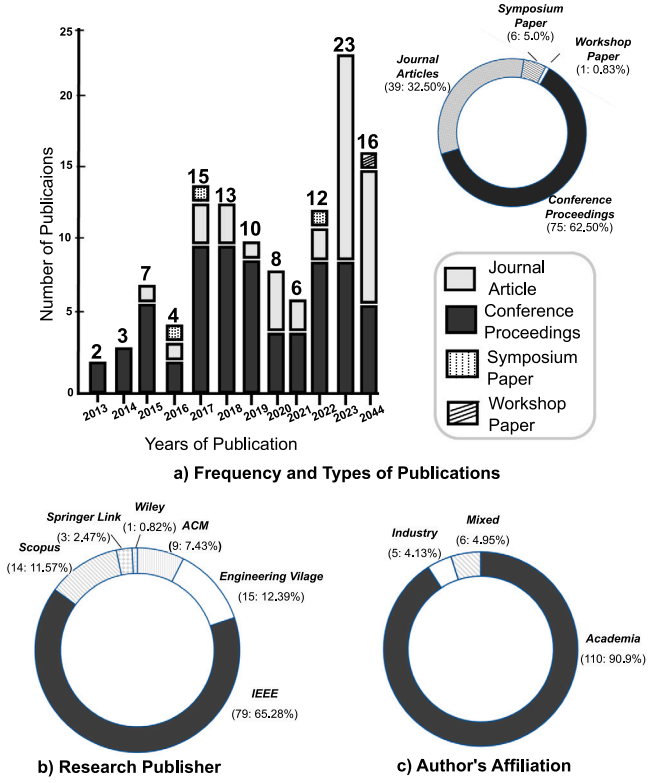

Fig. 3. Demographics of the selected studies, including: (a) Frequency and types of publications; (b) Research publisher; (c) Author's affiliation.

more funding in tech sectors after the pandemic, showing a renewed focus on improving containerization in multi-cloud environment.

Publishers Distribution: Fig. 3-b shows the distribution of studies by publisher. IEEE accounts for 65.29%, highlighting its major contribution. Scopus follows with 11.57%, while Engineering Village, ACM, and Springer Link represent smaller portions at 12.40%, 7.44%, and 2.48%, respectively. The chart emphasizes the dominance of IEEE in the identified studies, suggesting its importance as a primary source of relevant literature. This distribution also suggests to researchers where to primarily search and publish on containerization in multi-cloud environment.

Study Types: Fig. 3-a provides a breakdown of the types of selected studies. It is apparent that conferences dominate as the most common venue, comprising 62.5% of the total publications. Journals, often considered venues for mature and rigorously peer-reviewed research, make up 32.5%. This indicates a preference for conferences, likely due to faster publication and networking opportunities Symposiums and workshops represent for 5.0% and 0.83% respectively, suggesting that these are less common avenues, possibly due to their more specialized or focused nature.

Authors' Affiliations: Fig. 3-c illustrates the distribution of authors' affiliations in collected publications. 90.90% of the affiliations are associated with academia. This highlights the central role of academic institutions in producing and disseminating research. implying that industry professionals focus more on practical applications. The low industry affiliation may reflect the SMS's focus on academic databases, where industry participation is limited, indicating typical academic database trends and suggesting broader industry inclusion in future studies. Mixed affiliations, at 4.95%, indicate a small proportion of collaborations between academia and industry. This low rate of cross-affiliation collaboration may suggest that there is potential for increased synergy between academic and industrial entities in future research endeavors.

> **Takeaway 1**: Research on containerization in multi-cloud environments has gained more attention in recent years, especially after 2022. Most of the studies are published by IEEE and come from academic authors. This shows that the topic is mainly explored in academic settings. However, there is little contribution from industry, and most studies are presented at conferences rather than in journals. This suggests a need for more collaboration between academia and industry and more detailed research through journal publications.

Classification of Research Theme: The identified research themes and their notable trends are illustrated in Fig. 4 and detailed below. Fig. 4-(a) presents the taxonomical classification of existing research. This taxonomy systematically identifies, names, and represents various topics based on their similarities or distinctions. Fig. 4-(b) depicts the trends and temporal distribution of the themes over the years (2013–2024) based on the studies identified in our review.

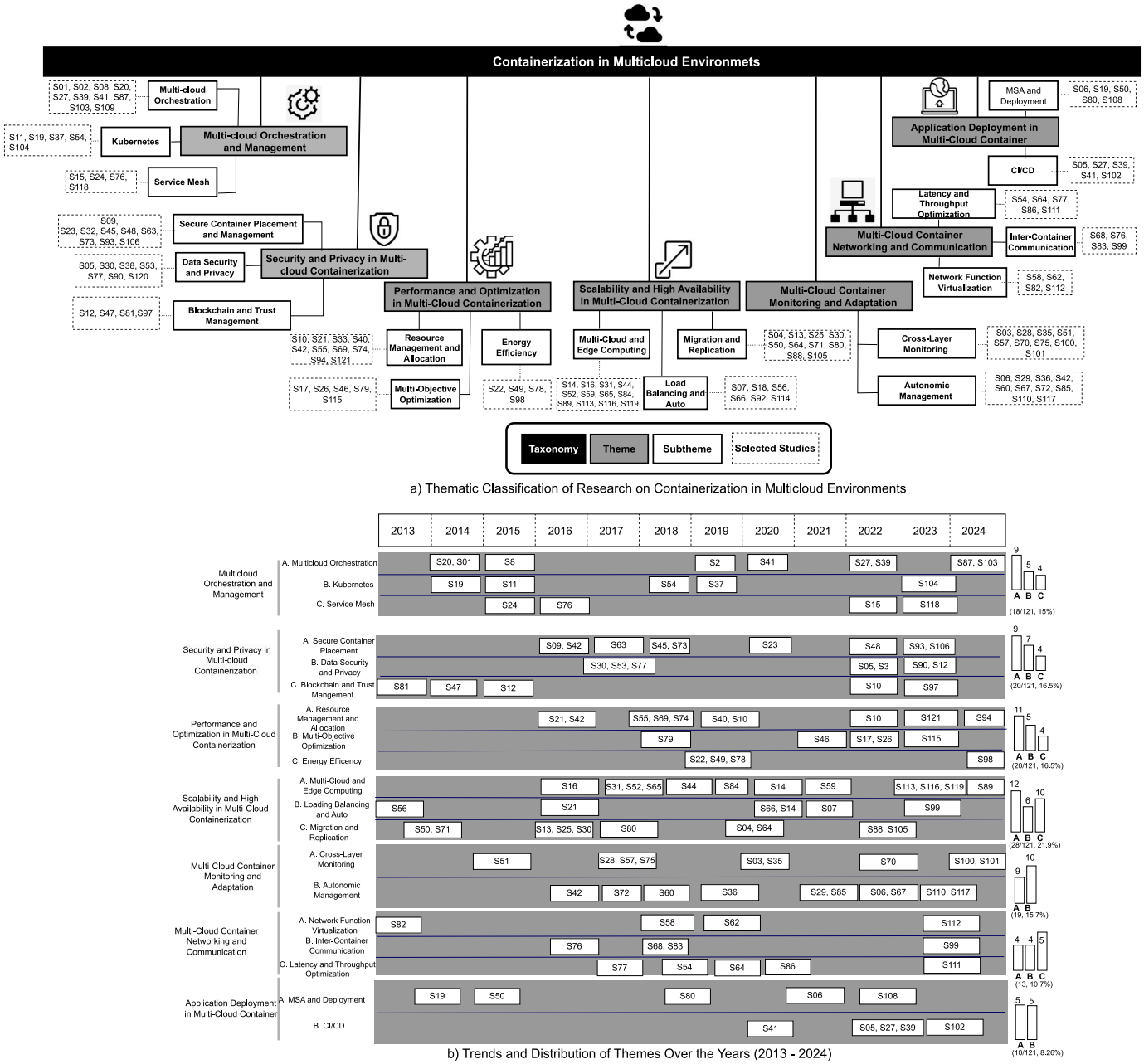

Fig. 4. Selected studies: (a) Thematic classification and (b) Trends and distribution of themes over the years.

- **Taxonomy of Research Themes**: The prominent research themes identified include "Scalability and High Availability" with 20 studies, and three other themes: "Security and Privacy" in "Multi-Cloud Container Monitoring and Adaptation", and "Performance and Optimization", each attracting attention in 15 studies. This highlights that the core areas of interest in multi-cloud environment are proficient container management, scalability, and performance optimization. "Security and Privacy" also stands as a considerable theme, as demonstrated by the 15 studies focusing on areas such as secure container handling, data security, and blockchain applications. This emphasizes the necessity of protecting data and applications in the context of containers spreading across multi-cloud environment. Moreover, it is noted that several studies encapsulate more than one theme. For instance, study S19 is cataloged under both 'Multi-cloud Orchestration and Management' and 'Application Deployment in Multi-cloud Container', signifying that it encompasses aspects of both Kubernetes and microservices architecture.

- **Trends and Temporal Distribution of Themes**: To highlight prominent trends and their progression over the years (2013–2024), we adopted the thematic and sub-thematic representation from Fig. 4(a) and illustrated them via a timeline and relative distribution of each theme in Fig. 4-(b). Specifically, Fig. 4-(b) demonstrates how certain research themes have progressed and matured (i.e., evolved as topics) over the years. For instance, the theme multi-cloud orchestration and management is addressed in a total of 18 studies, accounting for approximately 15% of the total reviewed studies. Our analysis indicates that one of the earliest research efforts on multi-cloud orchestration, published during 2014–2015, focused on the component-based and model-driven adaptation of applications for multi-cloud containers [S1, S20, S8]. Following this initial phase of application adaptation for multi-cloud orchestration, subsequent research explored techniques for benchmarking and enabling resource-efficient application orchestration [S2, S41, S39]. More recent studies, published

between 2022 and 2024, have investigated a variety of topics, including but not limited to DevOps-based development [S27] and monitoring orchestrated applications in multi-cloud environments [S39, S87]. Our review revealed no published studies on the theme of multi-cloud orchestration and management during 2013, 2017, or 2021. Two possible reasons for this are: (i) our literature search did not identify any relevant studies (see Section 2.3.1 - primary search), or (ii) the studies from those years did not meet the qualitative evaluation criteria (see Section 2.3.3 - quality assessment) for inclusion in the review.

Takeaway 2: The research themes show that when containerization is applied in multi-cloud environments, unique challenges emerge—such as the need for orchestration tools that can manage service consistency across diverse providers, and security mechanisms that adapt to varying cloud configurations. Studies have evolved from basic scalability and performance concerns to addressing how to efficiently monitor, deploy, and manage containerized applications across different cloud platforms using DevOps and federation-aware solutions. This highlights how multi-cloud settings demand more adaptable and interoperable container strategies than single-cloud systems.

Major Contributions of the Selected Studies: Table 6 provides an overview of the contributions of the selected studies. The contributions are primarily categorized into five main categories and 22 subcategories (see the Contributions sheet in Waseem et al. (2025)), through thematic analysis. The selected studies mainly focus on proposing and validating security solutions, reference architectures, cloud management frameworks, interoperability, orchestration, and tools for optimizing and deploying containerized applications in multi-cloud environment. Notably, the largest contributions come from the *Security Approaches* category, representing 27.25% of the total studies, which highlights the significant attention to cybersecurity & access control and cloud analysis approaches. Within this category, the *Cybersecurity and Access Control* subcategory alone accounts for 9.09%, emphasizing the importance of securing multi-cloud environment. The studies related to the *Reference Architectures and Models for Cloud* category, comprising 21.42% of the studies, focus on understanding and designing architectures specifically tailored for multi-cloud containerization. Within this category, **General Cloud Architectures** (4.92%) and **Fault Tolerance and Self-Healing** (4.13%) are key subcategories. These subcategories address critical challenges such as ensuring resilience and performance in cloud-native architectures. The studies on **Optimization and Deployment Approaches and Tools** category accounts for 18.16% of the studies, indicating a strong research focus on optimizing container deployment and resource scheduling. Within this category, **Optimization & Scheduling Approaches** (8.26%) represent the second-largest subcategory across all studies, highlighting the practical need for effective resource allocation in multi-cloud systems. Finally, **Cloud Management and Interoperability Frameworks**, accounting for 12.38%, primarily aim to improve resource utilization, service management, monitoring, and deployment in multi-cloud environments. This includes contributions from **Resource and Service Management Frameworks**, which represent 4.95% of the studies.

3.2. Container roles and implementation strategies (RQ1)

In this section, we present the roles that containers play in multi-cloud environments and the strategies used to implement them. Based on thematic analysis of the selected studies, we classified the findings into six major roles and five strategic categories. Each category is supported with examples to help readers relate to real-world use cases. The goal is to provide a holistic understanding of how containers are applied and managed across clouds. Readers can interpret the tables as a catalog of practical use scenarios and corresponding strategic techniques.

3.2.1. Container roles

We classify the roles of containers in a multi-cloud environment into six categories, as shown in Table 7. Each category is briefly reported below.

Takeaway 3: The contributions of the reviewed studies highlight how multi-cloud environments introduce specific demands on containerization, particularly in areas such as cross-platform security, cloud-aware architecture design, and adaptive resource scheduling. Unlike generic solutions, these studies propose container orchestration frameworks, interoperability tools, and security models tailored to the challenges of managing distributed containerized applications across heterogeneous cloud providers.

1. Container-Based Resource and Service Management: This category includes studies discussing the use of containers for resource management and hosting. The studies underline how the use of containers enables an efficient multi-tenancy solution, sharing of resources, and reduction in complexity due to efficient management of computational resources. It also supports OS-level virtualization and multi-tenant service management, enhancing general resource utilization. In the context of hosting services, containers provide scalable and isolated environments for application components and services. They support microcloud scaling, ensure efficient service delivery, and are vastly used in edge computing for hosting applications closer to end users. Each of these capabilities together shows why containers are very important for optimizing resource management and offering flexible and reliable hosting of services in cloud environments.

2. Container-based Application Deployment: Containers provide a way to package and distribute applications, along with their dependencies, across different environments. This category reports on the use of containers for deploying various types of applications, such as web apps, Platform as a Service (PaaS) solutions, data-intensive applications, and IoT apps.

3. Containerization in App Development and IoT: This category focuses on application development and IoT integration using containers in multi-cloud environment. Key areas include improving deployment speed, enhancing DevOps integration for better portability and efficiency, and enabling standardized software units. Containers are also use to address issues like privacy, compliance, and vendor lock-in. Notable highlights include their role in orchestrating container placement in fog computing and integration with tools like OpenVAS and Chef for enhanced security and automation.

4. Performance and Efficiency through Containers: In this category, we classified studies that discuss how containers improve performance and efficiency. The selected studies highlights their role in enabling scalable application development and enhancing performance isolation for SaaS. Additionally, studies reported that containers contribute to cloud platform optimization, boosting overall cloud performance, and improving cost efficiency specifically while using microservices. Further discussions include evaluating the performance impact of containers across various systems.

5. Container Technology Features: This category also gathers the studies that report the special characteristics and advantages of

Table 6
Selected studies' contributions on containerization in multi-cloud environment.

Category	Subcategory	Study #
Reference Architectures and Models for Cloud	General Cloud Architectures (6, 4.92%)	Watanabe et al. (2022), Sami and Mourad (2020), Carpio et al. (2020), Baresi et al. (2016), Pop et al. (2016) and Alonso et al. (2023a)
	IoT & Service Mesh Architecture (5, 4.13%)	Gattobigio et al. (2022), Rodigari et al. (2021), Merlino et al. (2024), Hossain et al. (2023) and Solayman and Qasha (2023)
	Fault Tolerance & Self-Healing (5, 4.13%)	Bisht and Kaur (2022), Bolivar et al. (2018), Tang (2021), Mfula and Nurminen (2018) and Nagarajan et al. (2023)
	Migration & Testing Architecture (3, 2.47%)	Slominski et al. (2015), Serhiienko and Spillner (2018) and Roy and Kalafatis (2023)
	Optimization & Reliability Model (3, 2.47%)	Bhamare et al. (2015), Aral et al. (2020) and Huang and Wang (2023)
	Domain-Specific Language (2, 1.65%)	Quint and Kratzke (2018) and Ferry et al. (2013)
	Performance Modeling (2, 1.65%)	Barna et al. (2017) and Rahman and Lama (2019)
Security approaches	Cybersecurity & Access Control (11, 9.09%)	He et al. (2022), Sheridan et al. (2017), Rios et al. (2017), Graupner et al. (2015) and Guechi and Maamri (2018)
	Cloud Analysis Approaches (7, 5.78%)	Pai and Kumar (2021), Pahl (2015), Dimitrijevic et al. (2019), Zeck and Bouroudjian (2017), Kiss (2018) and Gundu et al. (2020)
	Security and Performance Solution (6, 4.95%)	Sabbioni et al. (2020), Wang et al. (2020), Pandey et al. (2022) and Ranjbar et al. (2017)
	Security Approaches (5, 4.13%)	Deng et al. (2022), Gao et al. (2017), Bélair et al. (2019) and Wahab et al. (2016)
	Provisioning & Restoration Approach (4, 3.30%)	Pellegrini et al. (2017), Lazuka et al. (2022), Wang et al. (2019) and Kritikos et al. (2019)
Cloud Management and Interoperability Framework	Resource & Service Management Framework (6, 4.95%)	Saif et al. (2023), Harwalkar et al. (2019b), Libardi et al. (2014), Pham and Pham (2015), Sitaram et al. (2018) and Barletta et al. (2024)
	Monitoring & Deployment Framework (5, 4.13%)	Alhamazani et al. (2014), Trihinas et al. (2015), Fadda et al. (2019), Pham et al. (2015), Horchulhack et al. (2024) and Marques et al. (2024)
	Evaluation & Modeling Framework (4, 3.30%)	Ferry et al. (2014), Bou Ghantous and Gill (2021), Tundo et al. (2024) and Ramesh et al. (2023)
	Interoperability Framework (2, 1.65%)	Hua et al. (2020) and Challita et al. (2017)
Orchestration and Deployment Approach	Benchmarking & Optimization Approach (6, 4.13%)	Almeida et al. (2014), Jha et al. (2019), Shi et al. (2020), Buyya and Barreto (2015), Guerrero et al. (2018) and Petrovski and Gusev (2024)
	Service Management Approach (4, 3.30%)	Zhang et al. (2021), Chen et al. (2022), Gupta et al. (2021) and Benmerar et al. (2023)
	Simulation & Integration Approach (3, 2.47%)	Alonso et al. (2019), Mittal and Risco-Martín (2017) and del Castillo et al. (2013)
	Storage & Services Solution (3, 2.47%)	Harwalkar et al. (2019a), Shen et al. (2017) and Orzechowski et al. (2023)
Optimization and Deployment Approaches and Tool	Optimization & Scheduling Approach (10, 8.26%)	Wong et al. (2019), Al-shammari and Alwan (2018), Lorenzo et al. (2019), Altran et al. (2022), da Silva et al. (2023), Dogani et al. (2023), Pandi et al. (2023), Sofia et al. (2023) and Nguyen and Chien (2023)
	Multi-Cloud Platform (6, 4.95%)	Verdugo et al. (2017), Ouyang et al. (2022), Dreibholz et al. (2019), Janarthanan et al. (2018), Alaluna et al. (2017), Wong et al. (2019), Al-shammari and Alwan (2018), Lorenzo et al. (2019), Altran et al. (2022) and Santos et al. (2023)
	Deployment Approach and Tool (6, 4.95%)	Cui et al. (2016), Casola et al. (2017), Tricomi et al. (2017), Di Modica et al. (2019), Liu and Qin (2024) and Tiwari and Sharma (2023)

container technology, to be specific: lightweight virtualization, where the containers improve resource efficiency in edge computing, and applications are brought closer to end users. Additionally, their compatibility with orchestration tools like Kubernetes simplifies container management and deployment across diverse environments.

6. Other Container Uses: This category report several roles of container technology beyond traditional use cases. Key roles include container-based data storage, support for system architecture, and enhancing blockchain reliability in multi-cloud environment. Additionally, containers are used for evaluating scheduling algorithms and enabling virtual network composition, demonstrating their versatility in addressing a wide range of technical challenges.

Takeaway 4: The reviewed studies show that containers take on distinct roles in multi-cloud environments, particularly in addressing challenges such as provider heterogeneity, cross-cloud orchestration, and decentralized application deployment. Containers are used for multi-tenant resource management, cross-cloud service hosting, and workload migration. They also support edge and fog computing, compliance, and vendor lock-in prevention—roles less prominent in single-cloud contexts. This shows that multi-cloud scenarios introduce unique architectural and operational needs that shape how containers are implemented.

Table 7
Roles of containers in multi-cloud environment.

Category	Role of containers	ID
Container-Based Resource and Service Management	Container-Based Multi-Tenancy Solution	Ramos et al. (2021)
	Container-Driven Resource Sharing & Multi-Tenancy	Deng et al. (2022)
	Multi-Tenant Service Management via Containers	Slominski et al. (2015)
	Container-Enabled OS-Level Virtualization	Bélair et al. (2019)
	Container-Based Complexity Reduction	Shi et al. (2020)
	Containerized Service Hosting	Pham and Pham (2015)
	Multi-Cloud Workload Migration via Containers	Janarthanan et al. (2018) and Zeck and Bouroudjian (2017)
	Seamless Workload Migration via Containers	Zeck and Bouroudjian (2017)
	Container-Based Multi-Cloud Platform Execution	Serhiienko and Spillner (2018)
	Container-Driven MicroCloud Scaling & Isolation	Baresi et al. (2016)
Container-based Application Deployment	Containerized Web-App Deployment	Almeida et al. (2014)
	Container-Driven App Deployment	Gattobigio et al. (2022) and Pham et al. (2015)
	Containerized PaaS Prototyping	Verdugo et al. (2017)
	Data-Intensive App Deployment via Containers	Barna et al. (2017)
	Containerized App Management	Gupta et al. (2021)
	Container-Based Aneka Service Deployment	Buyya and Barreto (2015)
	Container-Driven Microservices Deployment & Scaling	Rahman and Lama (2019)
	Container-Based Multi-Cloud App Deployment & Scaling	Mfula and Nurminen (2018)
	Containerized IoT App Deployment	Bou Ghantous and Gill (2021)
Containerization in App Development & IoT	App Development & IoT Integration via Containers	Ouyang et al. (2022)
	Container-Based OpenVAS & Integration of Chef	Sheridan et al. (2017)
	Containers as Standardized Software Units	Al-shammari and Alwan (2018)
	Container Placement & Orchestration in Fog Computing	Sami and Mourad (2020)
	Container Deployment Speed & DevOps Integration	Lorenzo et al. (2019)
	Containerizing Apps for Privacy & Compliance	Dimitrijevic et al. (2019)
	Container-Driven Portability & Efficiency Improvement	Altran et al. (2022)
	Preventing Vendor Lock-In via Containers	Pellegrini et al. (2017)
	Container-Based Extensible App Development	Pop et al. (2016)
Performance & Efficiency via Containers	Scalable App Building via Containers	Pai and Kumar (2021)
	Container-Driven SaaS Performance Isolation	Wang et al. (2020)
	Cloud Performance Enhancement via Containers	Saif et al. (2023)
	Evaluating Performance Impact of Containers	Rodigari et al. (2021)
	Cost Efficiency of Microservices via Containers	Guerrero et al. (2018)
	Container-Based Cloud Platform Optimization	Kiss (2018)
Container Technology Features	Container Usage in Edge Computing	Wong et al. (2019)
	Container-Enabled Lightweight Virtualization	Pahl (2015)
	Key Features of Container Technology	Gao et al. (2017)
	Container-Based Edge Computing Virtualization	Carpio et al. (2020)
	Container Leverage in Kubernetes	Lazuka et al. (2022)
Other Container Uses	Container-Based Azure Data Storage	Bucur et al. (2018)
	Container & VM Support in Architecture	Bolivar et al. (2018)
	Blockchain-Based Multi-Cloud Reliability via Containers	Aral et al. (2020)
	Evaluating Scheduling Algorithm via Containers	Tang (2021)
	Container-Based Virtual Network Composition	Alaluna et al. (2017)

3.2.2. Implementation strategies

Container implementation strategies in multi-cloud environment refer to the various approaches and techniques used to deploy, manage, and orchestrate containerized applications across multiple cloud platforms. These strategies ensure efficient, secure, and scalable management of containerized workloads. Table 8 provides an overview of these strategies, categorized into subcategories. Overall, we identified 76 strategies classified into 11 subcategories and 5 categories. Notably, these strategies were identified from only 76 out of 121 selected studies (62.81%). The remaining studies do not explicitly discuss the strategies, or it is unclear what strategies they employed. One possible reason for this could be that many studies focus on high-level discussions of container technology without detailing specific implementation strategies. Additionally, in some cases, the strategies might have been implicitly integrated into broader architectural frameworks, making them difficult to extract or clearly identify.

1. Container Deployment and Management: This category focuses on how containers are deployed and managed in multi-cloud environment. We identified 23 strategies within this category, grouped into three subcategories: *Deployment Strategies*, *Container Lifecycle Management*, and *Data Management*. Deployment strategies, with 10 strategies identified, focus on optimizing the deployment of containers in multi-cloud environment, including approaches such as Kubernetes for containerized applications and Docker image management. Nine strategies related to container lifecycle management address resource monitoring, image optimization, and privacy, with examples like multi-cloud container resource usage monitoring and container image reorganization. The data management subcategory, comprising 4 strategies, focuses on the storage and management of container data in multi-cloud environment, featuring solutions such as Firebase Cloud for real-time databases and blockchain-based container log management. While deployment strategies are well-represented, fewer strategies focus on data management, indicating potential areas for further research in managing data across multi-cloud platforms.

2. Container Orchestration: We identified a total of 20 strategies in this category, classified into two subcategories: *Cloud Orchestration* and *Microservices Orchestration*. The *Cloud Orchestration* subcategory, comprising 13 strategies, focuses on managing containers

Table 8
Container implementation strategies in multi-cloud environment.

Category	Subcategory	Implementation Strategies	ID
Container Deployment and Management	Deployment Strategies	Automated Orchestration for Web Applications	Jha et al. (2019)
		Kubernetes for Containerized APPs	Gattobigio et al. (2022)
		Multi-cloud App Development Framework	Zhang et al. (2021)
		Modular Approach to Container Implementation	Pham et al. (2015)
		Deployment with Docker Images	Verdugo et al. (2017)
		Docker Image Creation and Management	Mittal and Risco-Martín (2017)
		Real-time Kubernetes workloads in multi-cloud	Barletta et al. (2024)
		AWS multi-cloud container app deployment	Kataru et al. (2023)
		Multi-cloud container policy enforcement	Liu and Qin (2024)
		Probes and orchestration for cloud containers	Tundo et al. (2024)
	Container Lifecycle Management	Container Image Optimization	Al-shammari and Alwan (2018)
		Efficient Image Reorganization	Lorenzo et al. (2019)
		Orchestrating with Docker and Kubernetes	Dimitrijevic et al. (2019)
		Executable Images on Multi-cloud Platforms	Serhiienko and Spillner (2018)
		Privacy monitoring for containerized clouds	Barati et al. (2023)
		Multi-cloud container resource usage monitoring	Horchulhack et al. (2024)
		Container ecosystem for multi-cloud deployment	Wong et al. (2023)
		Containers in AWS ECS for cloud services	Petrovski and Gusev (2024)
		Container management for multi-cloud MEC	Santos et al. (2023)
		IoT container deployment in multi-cloud	Solayman and Qasha (2023)
	Data Management	Firebase Cloud for Real-Time Database	Carpio et al. (2020)
		High Availability with SpyStorage MCSS	Shen et al. (2017)
		Data Storage with Object Store Service	Pop et al. (2016)
Container Orchestration	Cloud Orchestration	SLA-Based Container Strategies	Ramos et al. (2021)
		Task Division and Scaling in Container Strategy	Bisht and Kaur (2022)
		Location-Aware Service Brokering	Chen et al. (2022)
		Docker Containers for Multi-tenant Services	Slominski et al. (2015)
		Resource Acquisition for Aneka Containers	Buyya and Barreto (2015)
		Cross-Cloud Container and VM Management	Janarthanan et al. (2018)
		Container Orchestration Technologies Overview	Sami and Mourad (2020)
		Multi-cloud dynamic resource adaptation	Struhár et al. (2024)
		AI/ML orchestration in multi-cloud containers	Sofia et al. (2024)
		AI-driven container orchestration across clouds	Benmerar et al. (2023)
		Cross-cloud container orchestration tools	Alonso et al. (2023a)
		Kubernetes CI/CD for multi-cloud setups	Pandi et al. (2023)
	Microservices Orchestration	Microservices Architecture in IoT Cloud Apps	Ouyang et al. (2022)
		ADAPT Deployment of Microservices	Alonso et al. (2019)
		MSA Benchmark with Docker and Kubernetes	Rahman and Lama (2019)
		Kubernetes for power-efficient multi-cloud	Centofanti et al. (2024)
		Edge computing with Docker and Kubernetes	Hossain et al. (2023)
		Kubernetes for edge-cloud orchestration	Roy and Kalafatis (2023)
		Kubernetes orchestration for multi-cloud containers	Sofia et al. (2023)
Container Security	Container Placement	Secure Placement with Deep Learning	Deng et al. (2022)
		Container Placement Strategies	Altran et al. (2022)
		Dynamic microservice relocation across clouds	da Silva et al. (2023)
	Security Policies	SLA-Oriented Performance Isolation	Wang et al. (2020)
		Access Control and Defense for Container Security	Gao et al. (2017)
		HIP Integration for Docker Security	Ranjbar et al. (2017)
		Secure Connectivity in Multi-tenant Environment	Alaluna et al. (2017)
		Signed URLs for Access Control in Multi-cloud	Graupner et al. (2015)
Performance and Scaling Strategies	Performance Optimization	LXC Virtual Cluster for Performance	Cui et al. (2016)
		Scalable Multi-layer Architecture with Containers	Barna et al. (2017)
		Reactive Auto-scaling and Resource Provisioning	Saif et al. (2023)
		Istio for Control Plane Impact and Latency Reduction	Rodigari et al. (2021)
		Auto-scaling containers across clouds	Dogani et al. (2023)
		Multi-cloud resource monitoring and provisioning	Marques et al. (2024)
		AI resource allocation in multi-cloud containers	Nagarajan et al. (2023)
	Scaling Strategies	Edge Data Streaming and Real-Time Database	Carpio et al. (2020)
		MicroCloud Architecture for Efficient Scaling	Baresi et al. (2016)
		Distributed Workloads with Kubernetes	Lazuka et al. (2022)
		Hybrid Abstractions for Container and Host	Bélair et al. (2019)
Cloud Service and Networking	Cloud Service Management	Multi-Tenant Service with Docker and Cloudant	Slominski et al. (2015)
		Abstraction Layer for Cloud Service Standardization	Pellegrini et al. (2017)
		OpenStack and IoT in multi-cloud	Merlino et al. (2024)
		Data indexing for multi-cloud exposure	Orzechowski et al. (2023)
	Networking	Edge Scheduling with Location-Awareness	Wong et al. (2019)
		Openstack Networking with Kuryr	Bolivar et al. (2018)
		Federated Framework for Cloud Storage	Harwalkar et al. (2019a)
		Host Identity Protocol for Docker Networking	Ranjbar et al. (2017)

and resources across multi-cloud environment. These strategies include AI/ML-driven container orchestration, cross-cloud orchestration tools, and Kubernetes CI/CD for multi-cloud setups. The *Microservices Orchestration* subcategory, consisting of 7 strategies, emphasizes the orchestration of microservices using technologies like Kubernetes and Docker. Key strategies include Kubernetes orchestration for multi-cloud containers and edge computing with Docker and Kubernetes. These approaches are essential for enabling scalable and efficient management of containerized applications, with *Cloud Orchestration* strategies focusing on adaptive resource management, while *Microservices Orchestration* supports effective multi-cloud microservices deployment.

3. Container Security: We identified a total of 11 strategies within the *Container Security* category, classified into two subcategories: *Security Policies* and *Container Placement*. The Security Policies subcategory consists of eight strategies and mainly focuses on enhancing container security through mechanisms such as access control, secure connectivity, and deep learning-based policy generation. Key strategies include implementing defense mechanisms, integrating HIP for Docker security, and managing access in multi-cloud environments using signed URLs. These approaches help address security challenges, safeguard containerized applications, and ensure compliance in multi-cloud infrastructures. The Container Placement subcategory consists of three strategies and mainly focuses on optimizing container placement to balance security and efficiency. These strategies include secure placement using deep learning and the dynamic relocation of microservices across clouds. By enabling secure and adaptive placement, these strategies enhance resource utilization, reduce latency, and maintain consistent application performance across cloud environments.

4. Performance Optimization and Scaling Strategies: We identified a total of 12 strategies within the *Performance Optimization and Scaling Strategies* category, divided into two subcategories: Container *Performance Optimization and Scaling Strategies*. The *Container Performance Optimization* subcategory, consisting of seven strategies, focuses on improving container performance in multi-cloud environments. Key approaches include reactive auto-scaling and resource provisioning, AI-driven resource allocation, and multi-cloud resource monitoring. These strategies aim to enhance performance by optimizing resource usage and reducing latency across cloud platforms. The Scaling Strategies subcategory, with five strategies, centers on improving the scalability of containerized applications. Notable approaches include managing distributed workloads with Kubernetes, utilizing hybrid abstractions for container scaling, and implementing MicroCloud architectures for efficient scaling. Together, these strategies support the dynamic adjustment of containerized applications to fluctuating resource demands in multi-cloud setups while maintaining performance and efficiency.

5. Cloud Service and Networking: We identified a total of 10 strategies within the *Cloud Service and Networking* category, classified into two subcategories: *The Cloud Service Integration* subcategory, with five strategies, focuses on integrating services across multi-cloud platforms. Key approaches include multi-cloud runtime environments, abstraction layers for cloud service standardization, and the integration of OpenStack and IoT in multi-cloud setups. These strategies aim to simplify and streamline the seamless integration of services across different cloud platforms. The *Multi-Cloud Networking Strategies* subcategory, also comprising five strategies, addresses networking challenges in multi-cloud environments. Notable strategies include edge scheduling with location awareness, federated frameworks for cloud storage, and TOSCA-based deployment models for multi-cloud setups. These approaches ensure efficient and secure communication between clouds, supporting smooth service operation and data transfer across diverse cloud infrastructures.

Takeaway 5: The reviewed studies identify a wide range of implementation strategies that reflect the specific challenges of containerizing applications in multi-cloud environments. These include orchestrating containers across heterogeneous platforms, ensuring secure placement and access control under varying provider policies, and optimizing performance and scaling through AI-driven resource provisioning and hybrid abstractions. Unique to multi-cloud settings are strategies for managing distributed services, networking across clouds, and integrating edge and IoT services.

3.3. Pattern and strategies (RQ2)

Table 9 presents a thematic classification of patterns and strategies employed in container-based applications within multi-cloud environments. These patterns were identified through qualitative coding and grouped into multiple categories and subcategories based on their architectural, management, security, resilience, and migration focus. Each row corresponds to a pattern extracted from the literature, and studies that reported multiple distinct patterns appear in more than one subcategory (e.g., Pahl (2015), Graupner et al. (2015), Shen et al. (2017) and Pham and Pham (2015)). Table 9 is structured to allow readers to quickly identify and compare solution strategies across different technical domains. To improve interpretability, we provide a brief overview of each category below.

1. Cloud Architectural Patterns and Models: We identified 48 patterns within this category, grouped into four subcategories: *Architecture Patterns*, *Communication and Networking Patterns*, *Deployment Patterns*, and *Service Models*. The Architecture Patterns subcategory, with 22 patterns, is the largest, focusing on designing scalable and flexible cloud systems. Leading patterns include *Microservice Architecture* and *Service-Oriented Architecture (SOA)*, which emphasize modularity and independence of services in multi-cloud environment. Emerging patterns such as *AI-Driven Edge-Cloud Architecture* and *Blockchain-Based Architecture* reflect the incorporation of advanced technologies in cloud design. The Communication and Networking Patterns subcategory consists of 11 patterns, with notable strategies such as *Service Mesh* and *Service Chaining (SC)* providing essential frameworks for ensuring secure and efficient communication between distributed services across cloud environments. These patterns are critical for enabling seamless integration and operation of cloud services in multi-cloud setups. The Deployment Patterns subcategory, containing 8 patterns, focuses on strategies for deploying applications across multiple cloud platforms. Key patterns include *black-Green Deployment*, which supports smooth application updates by alternating between two environments, and *Object Store Service*, which ensures efficient storage of unstructured data across clouds. Advanced orchestration patterns like *Decentralized Orchestration Architecture* and *Policy-Driven Orchestration Architecture* further enhance deployment efficiency by automating resource management across clouds. Lastly, the Service Models subcategory comprises 7 models that define various cloud service delivery mechanisms. The leading models, *Platform-as-a-Service (PaaS)* and *Software-as-a-Service (SaaS)*, provide essential platforms for building, deploying, and managing applications in the cloud without the need for managing underlying infrastructure. Other models, like *Function-as-a-Service (FaaS)* and *Serverless Container-Oriented Architecture*, emphasize serverless computing, allowing developers to focus on code rather than infrastructure management.

2. Cloud Management and Resource Allocation Strategies: We identified 30 strategies within this category, grouped into four subcategories: *Multi-Cloud Management Strategies*, *Container Management Strategies*, *Edge Computing and IoT Strategies*, and *Resource Management Strategies*. The *Multi-Cloud Management Strategies* subcategory, with 8 strategies, focuses on managing and optimizing resources across multiple cloud environments. Leading strategies include the *Multi-cloud*

Table 9
Patterns and strategies for container-based applications in multi-cloud environment.

Categories	Subcategories	Patterns	ID
Cloud Architectural Patterns and Models	Architecture Patterns	Microservice Architecture	Jha et al. (2019), Verdugo et al. (2017), Ouyang et al. (2022), Ranjbar et al. (2017), Quint and Kratzke (2018), Pahl (2015), Mittal and Risco-Martín (2017), Sami and Mourad (2020), Guerrero et al. (2018), Centofanti et al. (2024), Hossain et al. (2023) and Tiwari and Sharma (2023)
		Service-Oriented Architecture (SOA)	Buyya and Barreto (2015), Casola et al. (2017), Gundu et al. (2020) and Merlino et al. (2024)
		Multitier Architecture, Layered Architecture	Jha et al. (2019), Ramos et al. (2021), Bucur et al. (2018), Barati et al. (2023), Horchulhack et al. (2024), Liu and Qin (2024) and Tundo et al. (2024)
		Client–Server Architecture	Hua et al. (2020) and Bucur et al. (2018)
		Component-based Architecture	Wong et al. (2019)
		Event-driven Architecture	Wurster et al. (2018)
		Multi-agent Architecture	Mfula and Nurminen (2018) and Saif et al. (2023)
		Master-worker Nodes Architecture	Sami and Mourad (2020)
		Netflix Zuul-based Seeker Component	Slominski et al. (2015)
		Multi-tenant Cloud Service Architecture	Slominski et al. (2015)
		Stateful Engine Architecture	Slominski et al. (2015)
		Distributed Multi-Cloud Native Architecture	Alonso et al. (2023a)
		Plugin-Based Deployment Automation Architecture	Harzenetter et al. (2023)
		Cloud-Edge Integrated Robotics Architecture	Huang and Wang (2023)
		Hybrid Malware Detection Architecture	Wang et al. (2023)
		Cloudfront-Enhanced Container Architecture	Kataru et al. (2023)
		AI-Driven Edge-Cloud Architecture	Benmerar et al. (2023)
		Metadata-Driven Architecture	Orzechowski et al. (2023)
		Blockchain-Based Architecture	Ahmad and Aujla (2023)
		Cross-Layered Edge-Cloud Architecture	Sofia et al. (2023) and Santos et al. (2023)
	Communication and Networking Patterns	Service Mesh	Gattobigio et al. (2022)
		Service Chaining (SC)	Bhamare et al. (2015)
		Multi-Cloud APIs	Graupner et al. (2015)
		Third-Party APIs	Graupner et al. (2015)
		Service Request Broker	Zhang et al. (2021)
		Publish and Subscribe Communication Protocol	Trihinas et al. (2015)
		Synchronous Network Communication Protocols	Almeida et al. (2014)
		Asynchronous Network Communication Protocols	Almeida et al. (2014)
		Network Function Virtualization (NFV)	Bhamare et al. (2015)
		Service Discovery	Pop et al. (2016)
		Blockchain-Based Identity and Access Management (IAM) Architecture	Fugkeaw (2023)
		Federated Cloud-Edge Architecture	Roy and Kalafatis (2023)
	Deployment Pattern	Black-Green deployment	Alonso et al. (2019) and Sheridan et al. (2017)
		Object store service	Pop et al. (2016)
		Multi-cloud Deployment Model	Sabbioni et al. (2020)
		Distributed Deployment Model	Sabbioni et al. (2020)
		Multi-Cloud Deployment Architecture	Petrovski and Gusev (2024) and Pandi et al. (2023)
		Decentralized Orchestration Architecture	Sofia et al. (2024)
		Policy-Driven Orchestration Architecture	Lee and Nam (2023)
		DevOps-Oriented Architecture	Solayman and Qasha (2023)
	Service Models	Platform-as-a-Service (PaaS)	Pahl (2015) and Shi et al. (2020)
		Software-as-a-Service	Serhiienko and Spillner (2018)
		Infrastructure-as-a-Service (IaaS)	Pahl (2015)
		Function-as-a-Service (FaaS)	Pahl (2015) and Ramesh et al. (2023)
		Multi-Cloud Orchestration Architecture	Pandi et al. (2023)
		Serverless Container-Oriented Architecture	Shi et al. (2020)

(*continued on next page*)

Table 9 (*continued*).

Cloud Management and Resource Allocation Strategies	Multi-Cloud Management Strategies	Multi-cloud computing strategy	S78, S81
		Hybrid cloud architecture	Wang et al. (2019)
		Hybrid/Multi-cloud Approach	Gattobigio et al. (2022) and Wahab et al. (2016)
		Introducing multi-cloud Middleware	Casola et al. (2017) and Acquaviva et al. (2017)
		MAPE-K control loop	Fadda et al. (2019) and Saif et al. (2023)
		Connecting to multiple cloud service providers	Pop et al. (2016)
		Multi-cloud load balancing	Wang et al. (2020) and Pop et al. (2016)
		Multi-Cloud BP provisioning	Kritikos et al. (2019)
		Cross-level orchestration of cloud services	Kritikos et al. (2019)
		Cross-level monitoring and adaptation of BPs	Kritikos et al. (2019)
	Container Management Strategies	Linux Container (LXC) project	Pahl (2015)
		Container engine	Pahl (2015)
		Docker Container Images	Pahl (2015)
		Lightweight virtualization	Pahl (2015)
		Portable application packaging	Pahl (2015)
		One-container-per-app approach	Pahl (2015)
		Data volumes and data volume containers	Pahl (2015)
	Edge Computing and IoT Strategies	Mobile Edge Computing (MEC)	Wong et al. (2019), Lazuka et al. (2022), Dreibholz et al. (2019) and Bucur et al. (2018)
		Fog computing	Al-shammari and Alwan (2018)
		Edge Services	Carpio et al. (2020)
		Connecting IoT Edge Devices to the Cluster	Gattobigio et al. (2022)
		Cloud of Things (CoT)	Gattobigio et al. (2022)
		Multi-Access Edge Computing (MEC) Architecture	Santos et al. (2023)
		Cross-Layered Edge-Cloud Architecture	Sofia et al. (2023)
		Cloud-Edge Integrated Robotics Architecture	Huang and Wang (2023)
		AI-Driven Edge-Cloud Architecture	Benmerar et al. (2023)
	Resource Management Strategies	Resource allocation based on different SLA levels	Wang et al. (2020)
		Container Orchestration, Kubernetes, Minikube	Pahl (2015), Gupta et al. (2021) and Sami and Mourad (2020)
		Rate-Based Stream Processing Architecture	Nguyen and Chien (2023)
		Multi-Agent Resource Optimization Architecture	Nagarajan et al. (2023)
Cloud Security and Resilience Strategies	Security and Resiliency Patterns	Security-by-Design approach	Buyya and Barreto (2015)
		Encryption of files	Graupner et al. (2015)
		Standardized APIs for file transfer	Graupner et al. (2015)
		Centralized/External ACM Service	Graupner et al. (2015)
		Distributed ACM Service	Graupner et al. (2015)
		Dynamic Switching between Authentication Methods	Graupner et al. (2015)
		ACM based on Signed URLs	Graupner et al. (2015)
		Attribute-Based Encryption and Signature	Shen et al. (2017)
		Secure data sharing	Shen et al. (2017)
		Data access control	Shen et al. (2017)
		Local encryption and signing	Shen et al. (2017)
		Authorized Tokens	Shen et al. (2017)
		Byzantine quorum protocol	Shen et al. (2017)
		Blockchain-Driven SLA Management Architecture	Neeraj et al. (2023)
		Hybrid Malware Detection Architecture	Wang et al. (2023)
	Fault-tolerance Strategies	Fault-tolerance with Redundant Engines	Slominski et al. (2015)
		Multi-cloud systems fault-tolerant workflow	Tang (2021)
		RAFT Consensus Algorithm	Naik (2016a)
Cloud Migration Strategies		Service-oriented Migration	Pham and Pham (2015)
		Application-centric Migration	Pham and Pham (2015)
		Image-based Migration	Pham and Pham (2015)
		Migration to a Virtualized Container	Pham and Pham (2015)

Computing Strategy, *Hybrid/Multi-cloud Approach*, and *MAPE-K control loop*, all of which emphasize seamless integration and resource optimization across cloud platforms. The *Container Management Strategies* subcategory includes 7 approaches that focus on effectively managing containers in cloud environments. Key examples are *Docker Container Images* and *Lightweight Virtualization*, both of which help simplify container operations and improve resource efficiency in multi-cloud settings. The *Edge Computing and IoT Strategies* subcategory consists of 8 strategies that explore how edge computing and IoT devices can be integrated with cloud systems. Noteworthy strategies such as *Mobile Edge Computing (MEC)* and *Fog Computing* aim to bring computation and storage closer to IoT devices, enhancing system responsiveness and performance. Lastly, the *Resource Management Strategies* subcategory comprises 7 strategies dedicated to optimizing resource usage in cloud environments. Prominent approaches like *Resource Allocation based on SLA levels*, *Container Orchestration, Kubernetes, Minikube*, and *AI-Driven Edge-Cloud Architecture* focus on smart resource distribution and coordination across cloud and edge platforms.

3. Cloud Security and Resilience Strategies: We identified 20 strategies within this category, divided into two subcategories: *Security and Resiliency Patterns* and *Fault-tolerance Strategies*. The *Security and Resiliency Patterns* subcategory, comprising 14 strategies, focuses on enhancing security and data resilience in cloud environments. Prominent strategies include *Security-by-Design approach*, *Encryption of*

files, and *Centralized/External ACM Service*, which aim to secure data sharing and access control. Emerging patterns like *Blockchain-Driven SLA Management Architecture* and *Hybrid Malware Detection Architecture* highlight the use of advanced technologies to ensure resilient cloud infrastructures. The *Fault-tolerance Strategies* subcategory contains 6 strategies that focus on ensuring system reliability in case of failures. Key strategies include *Fault-tolerance with Redundant Engines* and *Multi-cloud Systems Fault-tolerant Workflow*, both of which emphasize maintaining uninterrupted service in multi-cloud environment by building in redundancy and fault-tolerant processes. The *RAFT Consensus Algorithm* ensures consistency across distributed systems, further improving the resilience of containerized applications in a multi-cloud environment.

4. Cloud Migration Strategies: We identified 4 strategies within this category, which focus on various approaches to migrating applications and services to the cloud. These strategies include *Service-oriented Migration*, *Application-centric Migration*, *Image-based Migration*, and *Migration to a Virtualized Container*. Each of these strategies provides a unique method for transferring workloads and data from on-premises or legacy systems to cloud environments. The emphasis is on ensuring a smooth transition with minimal disruption to services, whether through the migration of individual services, entire applications, or the use of virtualized containers. These strategies are essential for organizations looking to leverage cloud technologies while preserving the integrity of their existing systems during the migration process.

> **Takeaway 6**: The studies reveal that when containerized applications are deployed in multi-cloud environments, architecture and management patterns must be adapted to support distributed, heterogeneous infrastructure. Patterns like Microservices and SOA are extended with multi-cloud orchestration and federation capabilities, while service mesh and chaining are tailored for secure, cross-cloud communication. Deployment and fault-tolerant patterns—such as black-Green deployment, decentralized orchestration, and blockchain-driven resilience.

3.4. Quality attributes and tactics (RQ3)

Table 10 presents a thematic classification of QAs and their associated implementation tactics for containerized applications in multi-cloud environments. These QAs and tactics were derived through thematic analysis of data extracted from 121 primary studies and are organized according to the ISO 25 010 standard. This standard defines high-level software quality characteristics and their sub-attributes, which served as a guiding framework for our classification. The table includes nine core QAs, namely Performance (Efficiency), Security, Compatibility, Scalability, Reliability, Portability, Flexibility, Maintainability, and Usability. For each QA, the table lists representative related terms, example study references, and commonly applied tactics. Due to space constraints, only five example tactics per QA are shown in Table 10, while the complete list of 70 tactics is available in the replication package (Waseem et al., 2025). These tactics are distributed across multiple studies and highlight recurring practices for achieving specific quality goals. To help readers navigate this classification, we briefly explain below how each QA is addressed in the literature and what patterns emerge in terms of implementation strategies. **Performance (Efficiency)**: Performance or efficiency is the most dominant QA, frequently discussed in 60 (49.58%) of the selected studies, along with related terms or characteristics such as time behavior, resource utilization, and capacity. We also listed several tactics that can be used to achieve performance (or efficiency) in containerized applications, such as efficient resource optimization and allocation, use of machine learning techniques for performance optimization, utilization of container technology, location-aware service brokering, and performance-oriented Service Level Agreements (SLA). For instance, optimizing resources ensures that the system uses the minimum possible resources while delivering the required output. Using ML techniques for performance optimization can enhance system performance by learning and adjusting the operational parameters. Similarly, utilizing container technology can help maintain optimal performance by encapsulating the application and its environment. These results also indicate that performance is a critical QA, with a strong focus in containerized applications, and the majority of studies advocate for advanced tactics to achieve optimal performance.

Security: This QA has been discussed in 38 studies along with various tactics. We also identified several characteristics of security that have been highlighted in the selected studies, such as confidentiality, authenticity, access control, authorization, privacy, trustworthiness, and integrity. Some of the identified tactics to improve the security of containerized applications from the selected studies include encryption and strong authentication mechanisms, implementation of firewalls, VPNs, or SDNs for network security, utilization of ML techniques to detect and prevent security threats and attacks, deployment of applications on diverse cloud providers, and consideration of users' security specifications. For example, encryption provides a secure way of transmitting data, and ML can detect unusual behavior that could signal a threat or attack.

Compatibility: This QA has been reported in 15 studies, along with characteristics such as interoperability and co-existence. Several tactics that can be employed to achieve compatibility include the implementation of lightweight communication protocols and modes, standardization of interfaces for seamless integration, componentization and modular design for interoperability, prototyping and exploring interoperability approaches for multi-cloud deployment, and interoperability standardization and federation between clouds. Standardizing interfaces can ensure that different software components can interact with each other seamlessly, enhancing compatibility. The results suggest that compatibility is essential for ensuring smooth integration and operation in multi-cloud environment.

Scalability: Scalability refers to the ability to expand capacity or performance without losing the efficiency or functionality of a software system. We identified 40 studies that report scalability along with other characteristics like capacity, extensibility, elasticity, and throughput. We also identified ten tactics, five of which are listed in Table 10, to achieve scalability. For example, scalability can be achieved through tactics like combining containerization and microservices for enhanced scalability, implementing elastic resource allocation, enabling migration between multi-cloud services, combining container-based deployment with runtime monitoring and optimization, and leveraging orchestration, federated networks, and geographic placement. Elastic resource allocation ensures that the system can seamlessly scale up or down in response to changing demands.

Reliability: This QA is about the system's correct operation and consistent performance without failure over a specified period, while ensuring the availability of the system. We identified 33 studies that report on reliability, along with characteristics such as maturity, availability, and fault tolerance. We identified nine tactics overall (see Table 10 and the QA and Tactics Sheet in Waseem et al. (2025)) to achieve reliability from the selected studies. These include deploying redundant engines for fault tolerance, distributing resources and replicating applications for improved response time, enforcing redundancy and distributed services for availability, deploying parallel search and multi-cloud distribution, and building fault-tolerant systems to enhance system reliability.

Portability: Portability refers to a system's ability to function correctly across different platforms (e.g., operating systems, hardware configurations). We identified 6 studies that report on portability, along

Table 10
Quality attributes and tactics for containerized applications in multi-cloud environment.

Quality attribute	Related terms	ID	Tactics
Performance (Efficiency)	Time behavior, Resource utilization, Capacity, Throughput, Response time	Almeida et al. (2014), Jha et al. (2019), Hua et al. (2020), Deng et al. (2022), Verdugo et al. (2017), Chen et al. (2022), Pham and Pham (2015), Ouyang et al. (2022), Ferry et al. (2014), Cui et al. (2016), Wang et al. (2020), Gao et al. (2017), Alonso et al. (2019), Gupta et al. (2021), Rios et al. (2017), Carpio et al. (2020), Lorenzo et al. (2019), Dimitrijevic et al. (2019), Pandey et al. (2022), Altran et al. (2022), Shi et al. (2020), Dreibholz et al. (2019), Wurster et al. (2018), Libardi et al. (2014), Harwalkar et al. (2019b), Casola et al. (2017), Bucur et al. (2018), Tricomi et al. (2017), Sitaram et al. (2018), Rodigari et al. (2021), Janarthanan et al. (2018), Rahman and Lama (2019), Harwalkar et al. (2019a), Zeck and Bouroudjian (2017) and Aral et al. (2020)	Efficient resource optimization and allocation Utilizing machine learning techniques for performance optimization Container technology utilization Location-aware service brokering Performance-oriented Service Level Agreements (SLA)
Security	Confidentiality, Authenticity, Access control, Authorization, Privacy, Trustworthiness, Integrity	Almeida et al. (2014), Gattobigio et al. (2022), Deng et al. (2022), Verdugo et al. (2017), Pham and Pham (2015), Wang et al. (2020), Pahl (2015), Gao et al. (2017), Gupta et al. (2021), Rios et al. (2017), Dimitrijevic et al. (2019), Pandey et al. (2022), Bélair et al. (2019), Wurster et al. (2018), Libardi et al. (2014), Casola et al. (2017), Bucur et al. (2018), Bolivar et al. (2018), Tricomi et al. (2017), Rodigari et al. (2021), Di Modica et al. (2019), Harwalkar et al. (2019a), Zeck and Bouroudjian (2017), Aral et al. (2020), Kiss (2018), Graupner et al. (2015), Alaluna et al. (2017), Guechi and Maamri (2018), Ranjbar et al. (2017) and Gundu et al. (2020)	Encryption and strong authentication mechanisms Implementation of firewalls, VPNs, or SDNs for network security Utilizing machine learning techniques to detect and prevent security threats and attacks Deployment of cloud applications on diverse cloud providers Consideration of users' security specifications and addressing CSP incompatibilities
Compatibility	Interoperability, Co-existence	Pham and Pham (2015), Ouyang et al. (2022), Cui et al. (2016), Pahl (2015), Alonso et al. (2019), Baresi et al. (2016), Wurster et al. (2018), Trihinas et al. (2015), Harwalkar et al. (2019b), Pellegrini et al. (2017), Harwalkar et al. (2019a), Graupner et al. (2015), Ranjbar et al. (2017), Quint and Kratzke (2018) and Ferry et al. (2013)	Implementing lightweight communication protocols and modes Standardization of interfaces for seamless integration Componentization and modular design for interoperability Prototyping and exploring interoperability approaches for multi-cloud deployment Interoperability standardization and federation between clouds
Scalability	Capacity, Extensibility, Elasticity, Throughput, Responsiveness	Almeida et al. (2014), Pham et al. (2015), Verdugo et al. (2017), Chen et al. (2022), Pham and Pham (2015), Ouyang et al. (2022), Slominski et al. (2015), Pai and Kumar (2021), Alhamazani et al. (2014), Cui et al. (2016), Wong et al. (2019), Pahl (2015), Gao et al. (2017), Alonso et al. (2019), Gupta et al. (2021), Mittal and Risco-Martín (2017), Carpio et al. (2020), Shi et al. (2020), Baresi et al. (2016), Wurster et al. (2018), Trihinas et al. (2015), Bucur et al. (2018), Tricomi et al. (2017), Rahman and Lama (2019), Aral et al. (2020), Kiss (2018), Alaluna et al. (2017), Mfula and Nurminen (2018), Pop et al. (2016), Ranjbar et al. (2017), Ferry et al. (2013) and Kritikos et al. (2019)	Combining Containerization and Microservices for enhanced scalability Implementing Elastic Resource Allocation Enabling Migration between Multi-Cloud Services Combining Container-based Deployment with Run-time Monitoring and Optimization Leveraging Orchestration, Federated Network, and Geographic Placement
Reliability	Maturity, Availability, Fault tolerance, Recoverability	Almeida et al. (2014), Jha et al. (2019), Ramos et al. (2021), Sabbioni et al. (2020), Naik (2016a), Slominski et al. (2015), Alhamazani et al. (2014), Ferry et al. (2014), Wong et al. (2019), Gao et al. (2017), Alonso et al. (2019), Al-shammari and Alwan (2018), Sami and Mourad (2020), Carpio et al. (2020), Dimitrijevic et al. (2019), Wurster et al. (2018), Libardi et al. (2014), Buyya and Barreto (2015), Acquaviva et al. (2017), Bucur et al. (2018), Bolivar et al. (2018), del Castillo et al. (2013), Tricomi et al. (2017), Janarthanan et al. (2018), Harwalkar et al. (2019a), Zeck and Bouroudjian (2017), Aral et al. (2020), Tang (2021), Graupner et al. (2015), Guechi and Maamri (2018), Mfula and Nurminen (2018), Shen et al. (2017) and Ferry et al. (2013)	Deploying Redundant Engines for Fault-Tolerance Distributing Resources and Replicating Applications for Improved Response Time Enforcing Redundancy and Distributed Service for Availability Deploying Parallel Search and Multi-Cloud Distribution Building Fault-Tolerant Systems
Portability	Installability, Replaceability, Adaptability	Pai and Kumar (2021), Pahl (2015), Gao et al. (2017), Mittal and Risco-Martín (2017), Rahman and Lama (2019) and Ranjbar et al. (2017)	Virtual Machine-Based Packaging Docker Image-Based Packaging Cloud-Portable Containerization Dynamic Cross-Level Adaptation and Provisioning Adaptive Rule-Based System Modification

(*continued on next page*)

Table 10 (*continued*).

Flexibility	Functional completeness, Functional correctness, Functional appropriateness	Ouyang et al. (2022), Cui et al. (2016), Alonso et al. (2019), Mittal and Risco-Martín (2017), Shi et al. (2020), Di Modica et al. (2019), Janarthanan et al. (2018), Lazuka et al. (2022), Pop et al. (2016) and Kritikos et al. (2019)	Service-oriented Architecture Dynamic allocation of resources across different cloud platforms based on demand Container orchestration Model-Driven Development, Risk Analysis Cross-cloud service orchestration,
Maintainability	Modularity, Reusability, Analyzability, Modifiability, Testability	Pham et al. (2015), Slominski et al. (2015), Gao et al. (2017), Alonso et al. (2019), Rahman and Lama (2019), Mfula and Nurminen (2018) and Ferry et al. (2013)	Microservices, easier maintenance Infrastructure Provisioning as Code (IaC) Containerization and Image Management Configuration Management and Templating Version Control and Change Management Continuous Integration and Deployment Automation
Usability	Appropriateness recognizability, Learnability, Operability, User interface aesthetics, Accessibility	Verdugo et al. (2017), Slominski et al. (2015), Alonso et al. (2019,?), Dimitrijevic et al. (2019) and Mfula and Nurminen (2018)	Feedback-loop controller for multi-cloud Infrastructure Container Orchestration (e.g., Kubernetes) User-Centric Interface Design Responsive and Adaptive User Experience Accessible Design and Compliance Error Handling and Feedback Mechanisms

with characteristics such as installability, replaceability, and adaptability, as well as 5 tactics that can help achieve portability. For example, portability can be achieved using tactics such as virtual machine-based packaging, Docker image-based packaging, cloud-portable containerization, dynamic cross-level adaptation and provisioning, and adaptive rule-based system modification. These tactics ensure that an application can be easily transferred from one computing environment to another.

Flexibility: This QA represents the system's capacity to accommodate new features based on user requirements with minimal effort. We identified 10 studies (see Table 10 and the QA and Tactics Sheet in Waseem et al. (2025)) that report this QA, along with five tactics that can help achieve it. For example, flexibility can be improved through tactics such as service-oriented architecture, on-demand dynamic allocation of resources across different cloud platforms, containers, container orchestration, model-driven development, risk analysis, and cross-cloud service orchestration. Using multi-cloud environment, for instance, offers the flexibility to choose services from different providers as per specific needs.

Maintainability: This QA has been reported in 7 studies, along with several characteristics such as modularity, reusability, and analyzability. We also identified 6 tactics from these studies that can help achieve maintainability. For example, tactics like microservices for easier maintenance, Infrastructure Provisioning as Code (IaC), containerization and image management, configuration management and templating, version control and change management, and continuous integration and deployment automation can enhance maintainability. For example, IaC allows developers to manage infrastructure more efficiently and minimize human errors, thus increasing maintainability.

Usability: This QA refers to the ease with which users can interact with a system to perform various operations. We identified 6 studies that report on usability, along with several characteristics such as learnability, and operability, as well as 6 other tactics. According to the selected studies, usability can be enhanced by employing tactics such as a feedback-loop controller for multi-cloud infrastructure, container orchestration (e.g., Kubernetes), user-centric interface design, responsive and adaptive user experience, accessible design and compliance, and error handling and feedback mechanisms. For instance, a feedback-loop controller is effective in a multi-cloud context as it dynamically adjusts resources across different cloud platforms based on user interactions, ensuring a seamless and responsive user experience tailored to the multi-cloud setup.

Takeaway 7: QAs and their associated tactics from the reviewed studies reveal that ensuring performance, security, reliability, and other QAs in containerized applications requires context-specific adaptations when operating in multi-cloud environments. Unlike single-cloud setups, these environments introduce heterogeneity in infrastructure, distributed control, and varying security models across providers. As a result, tactics such as SLA-aware resource optimization, federated access control, decentralized fault-tolerance mechanisms, and cross-platform provisioning are essential to achieving QA goals.

3.5. Security challenges and solution framework (RQ4 and RQ5)

This section presents a framework for addressing security challenges in containerized applications in multi-cloud environment. Based on a detailed review, it identifies seven key categories of challenges and solutions, outlined in Table 11. Each category is briefly reported below:

Data Security in Container-based Applications category covers multiple challenges related to data security in container-based applications, including data protection, database security, data compliance, and secure data transfer in a multi-cloud environment. Out of the 121 reviewed studies, 11 specifically highlighted these issues and provided corresponding solutions (see Table 11 and the Security Challenges-Solution sheet in Waseem et al. (2025)). The identified solutions focus on implementing robust encryption techniques, secure container orchestration, and compliance with regulatory standards such as GDPR and HIPAA.

The solutions also include using Role-Based and Attribute-Based Access Controls (RBAC and ABAC) to manage secure access effectively. Standardization efforts like Secure CSP Interoperability and OpenStack API integration aim to mitigate risks of unauthorized access and vendor lock-in. Furthermore, comprehensive frameworks like DRA emphasize secure data management, ensuring privacy, fault tolerance, and secure data transfers across different cloud platforms.

Access and Communication Control category reports on multiple challenges related to maintaining secure multi-cloud connectivity and managing user access control. Out of the 121 reviewed studies, 9 specifically identified issues and proposed solutions addressing secure communication, provider-agnostic operations, and decentralized identity management (see Table 11 and the Security Challenges-Solution

Table 11
Identified security challenges and solutions for containerized applications in multi-cloud environment.

ID	Challenge	Solution
Category 1: Data security		
Slominski et al. (2015)	Data Protection	Multi-Cloud Data Protection through Secure Containers
Gupta et al. (2021)	Database Security	Enhanced multi-cloud Database Security via Containerization
Dimitrijevic et al. (2019)	Data Compliance	Multi-cloud Legal Compliance and Data Protection
Pandey et al. (2022)	Data Security	Secure CSP Interoperability and Selection in multi-cloud
Acquaviva et al. (2017)	Data Ownership and Privacy	Multi-cloud Security via NoMISHAP Service Abstraction
Sitaram et al. (2018)	Data Transfer Security	Secure multi-cloud Integration via OpenStack Standardization
Bou Ghantous and Gill (2021)	Cloud Data Management	Multi-cloud Data Security via DRA Framework
Ahmad and Aujla (2023)	Data leaks and access risks	Cryptography, smart contracts, encryption
Sofia et al. (2024)	Decentralized data security challenge	Secure data handling with decentralized AI
Neeraj et al. (2023)	SLA data integrity challenge	Blockchain for SLA security and integrity
Santos et al. (2023)	Multi-cloud data privacy issue	Stringent security controls for data protection
Category 2: Access and communication Control		
Gattobigio et al. (2022)	Connectivity and Access Control	Multi-cloud Containerized Application Security
Dreibholz et al. (2019)	Secure Communication	Middleware Secure Deployment for multi-cloud
Buyya and Barreto (2015)	Provider-Agnostic Operations	Aneka-Based Security and Performance for Containerized multi-cloud Applications
Hossain et al. (2023)	Microservices privacy and access control	Encryption, authentication, service mesh for security
Fugkeaw (2023)	Decentralized IAM security issue	Blockchain for IAM security and data integrity
Roy and Kalafatis (2023)	Secure communication across edge-cloud environments	WireGuard for secure edge-cloud communication
Benmerar et al. (2023)	Multi-domain edge communication security	Secure communication and AI anomaly detection
Huang and Wang (2023)	Secure robot-cloud communication	Secure communication and encryption for data integrity
Category 3: Container security		
Deng et al. (2022)	Co-resident Security	Secure Placement Strategy via Deep RL
Gao et al. (2017)	Container Isolation	Power and Leakage Management for multi-cloud Security
Bélair et al. (2019)	Container and Host Security	Host-Container Cooperation for multi-cloud Defense
Wang et al. (2023)	Container security and malware detection	DockerWatch for malicious activity detection
Tiwari and Sharma (2023)	Multi-tenant serverless security challenge	Security best practices for serverless deployments
Category 4: Infrastructure and deployment management security		
Ferry et al. (2014)	Data Deployment	Secure Private Data Deployment Strategy
Wurster et al. (2018)	Infrastructure Orchestration	Standard Interface Implementation for Serverless Security
Libardi et al. (2014)	Storage Security	Multi-cloud Storage Security with MSSF
Quint and Kratzke (2018)	Migration Security	Infrastructure-Aware and Agnostic Secure Migration
Solayman and Qasha (2023)	Distributed infrastructure security challenge	Orchestration security for IoT data protection
Kataru et al. (2023)	Securing AWS app deployments	AWS best practices for CI/CD security
Category 5: Security monitoring and breach prevention		
Sheridan et al. (2017)	System Vulnerabilities	Automated Security Evaluation and Bootstrapping
Trihinas et al. (2015)	Breach Detection and Prevention	Metric Filtering and Fault Recovery for multi-cloud Security
Lee and Nam (2023)	Network policy misconfiguration risks	Automated network policy discovery and verification
Nguyen and Chien (2023)	Stream processing security risks	RBAM framework for secure data handling
Category 6: Trust and compliance management		
Casola et al. (2017)	Application Security and Compliance	Risk Analysis and Security SLAs with MUSA Framework
Harwalkar et al. (2019b)	Data Security and Standardization	OpenStack API-based multi-cloud Security and Interoperability
Tricomi et al. (2017)	Network Security	Inter-cloud Communication Protection with SFC
Wahab et al. (2016)	Trust and Community Formation	Trust-based Hedonic Coalitions for multi-cloud Security
Kritikos et al. (2019)	Orchestration and Adaptation	Multi-cloud Service Orchestration and Cross-Level Adaptation
Petrovski and Gusev (2024)	Multicloud compliance and security issue	AWS security measures for multicloud environments
Alonso et al. (2023a)	Cross-cloud security and compliance	Security frameworks for multi-cloud environment
Category 7: Placement strategy		
Sami and Mourad (2020)	Placement Strategy	Security Model for Volunteering Fog Services
Pandi et al. (2023)	Multi-cloud security concerns	Security groups, VPCs, and infrastructure security
Ramesh et al. (2023)	Multicloud environment security issue	Multicloud security for risk diversification
Sofia et al. (2023)	Edge-cloud security concerns	Enhanced security protocols for decentralized environments
Harzenetter et al. (2023)	Cross-technology update security risks	Integrated security in workflows for management
Nagarajan et al. (2023)	Resource allocation security risks	Security measures in resource allocation framework

sheet in Waseem et al. (2025)). Advanced encryption and authentication mechanisms, e.g., TLS, JWT, and OAuth 2.0, were identified as solutions to secure communications across cloud and edge environments. Secure deployment of applications using middleware solutions was also discussed, which make use of SSL/TLS and access controls to enhance security attributes. Provider-agnostic middleware, e.g., the one based on Aneka, enables runtime provider selection and thus facilitates seamless operations across multi-cloud environment. Solutions for decentralized IAM security relate to blockchain utilization for improving data integrity and confidentiality. Secure communications in multi-domain edge environments are achieved through frameworks like WireGuard, backed by strong service mesh systems that maintain connectivity and protect against unauthorized access. Security monitoring for anomaly detection and policy enforcement was also noted as important for sustaining secure operations.

Container Security category reports on challenges such as co-resident security, container isolation, host security, and multi-tenant serverless security in multi-cloud environment. Among the 121 reviewed studies, 5 identified these issues and proposed solutions to address them (see Table 11 and the Security Challenges-Solution sheet in Waseem et al. (2025)). The solutions identified include the use of secure placement strategies via deep reinforcement learning to manage secure container placement in multi-cloud platforms. Security measures for container isolation include the technique of Power and Leakage

Management, followed by Linux kernel-level isolation using SELinux. In order to enhance security on both container and host sides, solutions such as Host-Container Cooperation and Docker-based detection tools, like DockerWatch, enhance the defenses and malicious activity detection. Finally, with a view to further securing multi-tenant serverless environments, the implementation of best practices regarding security in the CLI tool used for managing serverless deployments was underlined.

Infrastructure and Deployment Management Security category reports on challenges related to securing data deployment, infrastructure orchestration, storage security, and migration security in multi-cloud environment. Out of the reviewed 121 studies, 6 specifically addressed these types of challenges and proposed solutions (see Table 11 and the Security Challenges-Solution sheet in Waseem et al. (2025)). The identified solutions include the use of a Secure Private Data Deployment Strategy with encryption techniques like AES-256 and secure protocols such as SFTP and HTTPS to safeguard data. In the case of infrastructure orchestration, the focus is on the implementation of standard interfaces, such as OpenStack APIs, to ensure security and interoperability during deployments. For storage security, the solutions introduced the Multi-Cloud Storage Framework (MSSF), providing secure storage management through the implementation of RBAC and data encryption, along with periodic audits. The solutions also introduced measures for secure migration through both Infrastructure-Aware and Agnostic Secure Migration strategies, emphasizing image signing and data encryption under the least-privilege principle. Additionally, the reviewed studies proposed orchestration security for IoT distributed data and operations, as well as AWS best practices to help protect CI/CD pipelines running in multi-cloud environment.

Security Monitoring and Breach Prevention category reports on challenges related to system vulnerabilities, breach detection and prevention, and network policy misconfiguration risks in multi-cloud environment. Out of the 121 reviewed studies, 4 specifically addressed these issues and proposed corresponding solutions (see Table 11 and the Security Challenges-Solution sheet in Waseem et al. (2025)). The identified solutions suggest implementing automated security evaluations and bootstrapping mechanisms for Virtual Machines with security configurations using tools like Terraform, Chef, Ansible, or Puppet. The studies also highlighted several other tools for vulnerability assessment, such as OpenVAS and Nessus, which can be used to address delays in patching and reduce system vulnerabilities. In terms of breach detection and prevention, the reviewed studies emphasized the need to enhance security through metric filtering, fault recovery, and the automated deployment of SIEM, IDS, and IPS tools. Log analysis solutions were considered necessary for proactive breach management. As a result, automated network policy discovery and verification mechanisms, such as KUNERVA, were proposed, incorporating rigorous verification to prevent most potential breaches in network policy management.

Trust and Compliance Management category reports on solutions related to ensuring security and compliance in multi-cloud environment. Among the 121 reviewed studies, 7 specifically discussed these issues and proposed corresponding solutions (see Table 11 and the Security Challenges-Solution sheet in Waseem et al. (2025)). The identified solutions emphasize the need for security controls and standardized practices, such as risk analysis and secure coding using MUSA. The use of standardized API-based cloud interfaces, like the OpenStack API, was also emphasized to ensure both data security and interoperability between different cloud platforms. Establishing trust-based coalitions for community security was another important approach, supported by IAM systems and secure communication frameworks like SFC to enable secure interactions and collaborations within multi-cloud settings.

Placement Strategy category highlights the need for secure service deployment on volunteering fog nodes in multi-cloud environment. Of the 121 reviewed studies, 6 proposed solutions for these challenges (see Table 11 and the Security Challenges-Solution sheet in Waseem et al. (2025)). The solutions involve developing a security model that integrates SSL/TLS protocols for communication and certificate-based access controls to protect deployed services on fog nodes. The strategies include implementing a combination of security groups, VPCs, and infrastructure-level measures to distribute risks and prevent unauthorized access. The reviewed studies recommend adaptive methods for handling security in dynamic and decentralized edge-cloud environments by proposing cross-technology security updates and resource allocation frameworks to avoid conflicts and vulnerabilities.

> **Takeaway 8**: Security in containerized multi-cloud environments poses unique challenges due to the combined complexity of container orchestration and distributed cloud platforms. The reviewed studies show that addressing these issues requires specific strategies—such as secure container placement, decentralized identity management, and container-aware compliance frameworks—not typically needed in single-cloud or non-containerized setups. These solutions highlight the distinct security needs that arise only when containerization is applied across multiple clouds.

3.6. Automation challenges and solution framework (RQ4 and RQ5)

Table 12 presents an overview of automation challenges and corresponding solutions for containerized applications in multi-cloud environments. The framework is organized into eight categories, each addressing a distinct aspect of automation—such as orchestration, deployment, resource management, and standardization. These categories capture recurring challenges and the solution strategies proposed across the selected studies.

Multi-Cloud Automation category reports on challenges and solutions related to automating processes for combining and administering various cloud computing environments. Out of the 121 reviewed studies, 13 identified these challenges and provided associated solutions (see Table 12 and Automation Challenges-Solutions sheet in Waseem et al. (2025)). Key challenges involve the integration, provisioning, orchestration, and adaptation of multiple clouds. The solutions focus on utilizing proxies for environment interfaces to ease management across multiple cloud platforms and employing model-driven engineering techniques to manage provisioning and adaptation more efficiently. Dynamic on-demand fog computing was identified as an essential approach for handling multi-cloud environment dynamically. At the level of cloud redundancy and resource management driven by security policies, smart and user-friendly automation techniques, such as MSSF, were recommended. For managing the complexity of multi-cloud orchestration, serverless deployment methodologies were proposed, using models like TOSCA-based orchestration to improve scalability and coordination between different cloud stacks. Recommendations also include deploying application-level resource managers to support multi-cloud operations and adopting secure abstraction models to create uniform interfaces for managing a wide range of cloud products. MUSA Deployer tools were highlighted for their key role in automating deployment and monitoring activities, ensuring overall security compliance, and simplifying the deployment pipeline across federated clouds.

Automation in Deployment and Scaling category reports on challenges and solutions related to automating deployment and scaling processes for containerized applications across multi-cloud platforms. Out of the 121 reviewed studies, 12 identified these challenges and proposed corresponding solutions (see Table 12 and the Automation Challenges-Solutions sheet in Waseem et al. (2025)). Key challenges include intelligent container placement, application runtime switch

automation, and service deployment issues. The proposed solutions emphasized intelligent algorithms for container placement, using machine learning for multi-cloud workload balancing. Runtime switch techniques are also being automated with methods such as service drivers, which are designed to be lightweight and enable seamless application migration and replication. DevOps methodologies were frequently mentioned in the literature in relation to managing deployment and scaling through continuous development and error reduction. Security during deployment is ensured through automated security measures integrated into the pipeline, aiming for a secure setup with reduced vulnerabilities. The automation strategies include elastic resource provisioning using predictive analytics to handle fluctuating workloads in real time. Optimization models were involved in solutions for virtual function placement and container deployment to enhance efficiency. This emphasis on automating and optimizing deployment strategies led to the proposal of comprehensive multi-cloud orchestration tools that facilitate the deployment of containers and VMs, aimed at improving agility and scalability within diverse cloud environments.

Resource Management Automation category reports on challenges and solutions related to automating the management of resources in multi-cloud environment. Out of the 121 studies reviewed, 9 specifically addressed these challenges and provided appropriate solutions (see Table 12 and the Automation Challenges-Solutions sheet in Waseem et al. (2025)). These challenges include resource overbooking management, resource management for containerized applications, and resource commissioning and decommissioning.

The proposed solutions include machine learning-based resource overbooking detection for efficiently managing service containers across multiple cloud platforms. To manage containerized applications, custom Kubernetes schedulers, such as label-affinity schedulers, were proposed to optimize orchestration and allocation. Additionally, a GA-based algorithm was suggested to reduce the search space of VM types in data centers, simplifying the resource allocation problem within multi-cloud environment. For resource commissioning and decommissioning, proposals such as the MicroCloud architecture enabled fine-grained resource allocation and coordinated adaptation workflows. The solutions also addressed cloud federation and inter-platform portability challenges, utilizing middleware platforms for security enhancement, cost management, and performance optimization. The JCatascopia automated modular monitoring framework was introduced to monitor runtime configurations and detect elasticity actions. Finally, SLA compliance testing automation was proposed to manage resource allocation processes and enforce corrective actions, ensuring that service level agreements are met across cloud environments.

Data and Application Migration Automation: category reports challenges and solutions related to the automation of testing and benchmarking processes for containerized applications in multi-cloud environment. Among the 121 studies reviewed, 3 identified these challenges along with their proposed solutions (see Table 12 and the Automation Challenges-Solutions sheet in Waseem et al. (2025)). The main challenges include ensuring data integrity during the migration process and enabling the smooth migration of legacy Web applications to cloud services. The proposed solutions involve introducing autonomous management cycles to reduce complexity and ensure effective deployment across multi-cloud environment. Other solutions include reusable architectural patterns to accelerate the modernization of legacy applications in containerized multi-cloud environment. Additionally, the CloudSME Simulation Platform was highlighted for its ability to accelerate the porting of existing applications to cloud infrastructures, with support for more complex tasks such as billing and resource management.

Testing and Benchmarking Automation category highlights the challenges and solutions related to automating the testing and benchmarking processes for containerized applications in multi-cloud environment. Out of the 121 studies reviewed, 3 studies specifically addressed these challenges and provided relevant solutions (see Table 12 and the Automation Challenges-Solutions sheet in Waseem et al. (2025)). The main challenges revolve around automating user-oriented testing and benchmarking across various cloud platforms. The solutions range from just-in-time deployment and streamlined multicomponent prototyping to rapid testing, enabled by container-based lightweight virtualization. Additionally, Smart Docker Benchmarking Orchestrators were proposed for automating benchmarking in multi-cloud environment, thereby enhancing the overall efficiency of testing procedures. The application of Computational Intelligence (CI) was also suggested to improve dependability in automated testing for containerized multi-cloud systems.

Standardization and Interoperability Challenges category reports on the challenges and solutions related to standardization and interoperability in the automation of containerized applications across multiple cloud environments. Out of the 121 reviewed studies, 6 specifically identified these challenges and proposed corresponding solutions (see Table 12 and the Automation Challenges-Solutions sheet in Waseem et al. (2025)). The main challenges involve standardizing automation processes, selecting cloud providers, and addressing the lack of unified templates for microservices deployment. The proposed solutions include service mesh networks and multi-cloud cluster federation to standardize the automation processes for containerized applications. AI-driven tools for optimal cloud provider selection were also introduced, which help in selecting the most suitable provider and node based on factors such as cost-effectiveness and runtime efficiency in multi-cloud environment. To address the lack of common templates for microservices, a Service Model approach was proposed to automate the deployment of microservices across multiple cloud platforms, thereby reducing operational complexity. For cloud interoperability, an AI-driven platform, such as multi-cloud Harmony, was developed to facilitate seamless data transfer and communication between disparate cloud environments. In terms of cloud application standardization, the solutions emphasized promoting the adoption of global standards and interfaces to improve interoperability between cloud service providers and middleware platforms. Finally, automation frameworks for package creation and scaling, such as those using machine-readable definition files, were proposed to efficiently manage both virtual machines and Docker containers across multi-cloud environment.

Application and Service Management category reports challenges and solutions related to the automation of application and service management processes in containerized multi-cloud environment.Out of the 121 reviewed studies, 2 specifically addressed these challenges and proposed appropriate solutions (see Table 12 and the Automation Challenges-Solutions sheet in Waseem et al. (2025)). The key challenges include the selection of 5G ecosystem components and end-to-end tail latency management in microservice architectures. The proposed solutions involve using open-source software components to customize the orchestration of containers and virtual machines in modern multi-cloud deployments. Additionally, performance modeling approaches were suggested to evaluate the performance of container-level and VM-level data in multi-cloud environment, enabling more efficient latency management and optimized overall performance.

Runtime and Service Discovery category organizes the challenges and solutions related to runtime management and services discovery in multi-cloud environment, particularly in managing the complexity of network control and administration for containerized applications. Out of the 121 reviewed studies, 4 specifically tackled these challenges (see Table 12 and the Automation Challenges-Solutions sheet in Waseem et al. (2025)). The proposed solutions include automating network management to address the increasing complexity of distributed applications across multi-cloud topologies. Additionally, automated discovery and selection algorithms were proposed to tag and manage social network services within containerized multi-cloud systems. Autonomic computing systems were introduced to handle the growing complexity of applications by facilitating knowledge transfer, enabling more effective containerization management through automation.

Table 12
Identified automation challenges and solutions for containerized applications in multi-cloud environment.

ID	Challenge	Solution
Category 1: Multi-cloud automation		
Sabbioni et al. (2020)	Integrating multi-cloud environment	Implementing Environment Proxies
Ferry et al. (2014)	Automating Multi-Cloud Provisioning	Adopting Model-Driven Engineering
Sami and Mourad (2020)	Creating On-Demand Fog in Multi-Cloud	Enabling On-Demand Fog Computing
Libardi et al. (2014)	Enhancing User-Friendly Automation	Using Multi-Cloud Storage Selection
Wurster et al. (2018)	Orchestrating Multi-Stack Environments	Deploying with TOSCA-based Approach
Rios et al. (2017)	Securing Multi-Cloud Application Creation	Applying MUSA Framework
Dimitrijevic et al. (2019)	Unifying Resource Abstraction	Managing with Resource Managers
Casola et al. (2017)	Integrating Multi-Cloud Deployments	Automating MUSA Deployment
Tricomi et al. (2017)	Coordinating Federated Cloud Deployments	Orchestrating Subsystem Deployment
Sitaram et al. (2018)	Setting Up Hybrid multi-cloud environment	Standardizing Interoperability
Quint and Kratzke (2018)	Ensuring Multi-Cloud Interoperability	Enabling Multi-Cloud Support
Challita et al. (2017)	Managing Multi-Cloud Brokerage Systems	Managing Cloud Services
Kritikos et al. (2019)	Implementing Multi-Cloud Orchestration	Migrating Manually to Cloud
Bou Ghantous and Gill (2021)	Establishing DevOps in Multi-Cloud	Automating Agile Development
da Silva et al. (2023)	Addressing Cost and Migration Challenges	Managing Complexity
Pandi et al. (2023)	Overcoming Multi-Cloud Automation Barriers	Integrating Jenkins and Kubernetes
Ramesh et al. (2023)	Improving Multi-Cloud Cost Efficiency	Optimizing Deployment Costs
Petrovski and Gusev (2024)	Simplifying Multi-Cloud Deployment Complexity	Adopting AWS Deployment Strategies
Kataru et al. (2023)	Maintaining Multi-Cloud Applications	Automating Resource Allocation
Category 2: Automation in deployment and scaling		
Alonso et al. (2019)	Automating Multi-Cloud Deployment Monitoring	Adopting Multi-Cloud Strategy
Deng et al. (2022)	Optimizing Container Placement	Implementing Intelligent Placement
Zhang et al. (2021)	Automating Runtime Application Switching	Deploying Service Driver Model
Pham and Pham (2015)	Enabling Fine-Grained Component Automation	Replicating and Migrating Services
Barna et al. (2017)	Streamlining DevOps Automation	Developing with DevOps Approach
Gupta et al. (2021)	Automating Service Deployment	Using Automated Scripts
Sheridan et al. (2017)	Securing Deployment Time	Securing with Automation Tools
Mittal and Risco-Martín (2017)	Automating Distributed Simulations	Adopting DevOps Methodologies
Al-shammari and Alwan (2018)	Improving Software Deployment Speed	Optimizing Container Images
Lorenzo et al. (2019)	Automating Container Startup	Profiling Container Execution
Pandey et al. (2022)	Dynamically Scaling Resources	Automating Scaling with CSP Capability
Pahl (2015)	Scaling Container Deployment	Scaling with Kubernetes
Saif et al. (2023)	Provisioning Elastic Resources	Provisioning Resources Elastically
Bhamare et al. (2015)	Automating VF Placement	Optimizing VF Placement
Guerrero et al. (2018)	Managing Container and VM Deployment	Orchestrating Multi-Cloud Deployments
Category 3: Resource management automation		
Ramos et al. (2021)	Handling Resource Overbooking	Detecting Overbooking with ML
Cui et al. (2016)	Managing Virtual Clusters	Managing with ClaaS Model
Altran et al. (2022)	Scheduling Containerized Applications	Scheduling with Kubernetes
Shi et al. (2020)	Selecting Optimal VM Types	Using GA-based Algorithms
Baresi et al. (2016)	Automating Resource Commissioning	Adopting MicroCloud Architecture
Dreibholz et al. (2019)	Supporting Cloud Federation Automation	Optimizing with Middleware Platform
Trihinas et al. (2015)	Automating Elastic Resource Detection	Automating with JCatascopia
Buyya and Barreto (2015)	Automating Resource Acquisition	Managing Resources with Aneka
Mfula and Nurminen (2018)	Enforcing SLA Compliance	Testing for Compliance
Category 4: Data and application migration automation		
Almeida et al. (2014)	Safeguarding Data Integrity in Migration	Maintaining Integrity Autonomously
Slominski et al. (2015)	Migrating Legacy Web Applications	Reusing Architectural Patterns
Kiss (2018)	Porting Applications to Cloud	Using CloudSME Platform
Category 5: Testing and benchmarking automation		
Verdugo et al. (2017)	Automating User-Oriented Testing	Virtualizing with Lightweight Containers
Jha et al. (2019)	Streamlining Benchmarking	Orchestrating Docker Benchmarking
Naik (2016a)	Enhancing Automation Dependability	Enhancing Dependability with CI
Category 6: Standardization and interoperability challenges		
Gattobigio et al. (2022)	Standardizing Automation Processes	Integrating Service Mesh
Lazuka et al. (2022)	Optimizing Cloud Provider Selection	Selecting Optimal Providers
Ouyang et al. (2022)	Unifying Microservices Deployment	Adopting Service Chain Models
Harwalkar et al. (2019a)	Addressing Cloud Interoperability	Managing with Multi-Cloud Harmony
Pellegrini et al. (2017)	Standardizing Cloud Applications	Standardizing Interfaces
Pai and Kumar (2021)	Automating Package Creation and Scaling	Automating Package Frameworks
Category 7: Application and service management		
Bolivar et al. (2018)	Selecting 5G Ecosystem Components	Utilizing Open-Source Components
Rahman and Lama (2019)	Managing End-to-End Tail Latency	Modeling Performance Efficiently
Serhiienko and Spillner (2018)	Handling Distributed Application Complexity	Managing Distributed Applications
Category 8: Runtime and service discovery		
Alaluna et al. (2017)	Addressing Network Complexity	Managing Network Complexity
Wahab et al. (2016)	Improving Social Network Service Discovery	Automating Service Discovery
Pham et al. (2015)	Administering Complex Applications	Building Autonomic Systems

Takeaway 9: Automation challenges in containerized multi-cloud environments are not merely generic cloud or container issues; they arise from the complex interplay between container portability and cloud heterogeneity. Studies reveal that managing orchestration, scaling, and resource allocation across diverse cloud platforms requires specialized solutions, such as TOSCA-based orchestration, intelligent placement algorithms, and DevOps-integrated pipelines, highlighting how containerization uniquely amplifies automation demands in multi-cloud settings.

3.7. Deployment challenges and solution framework (RQ4 and RQ5)

Table 13 presents a comprehensive overview of the challenges and corresponding solutions faced during the deployment of containerized applications in multi-cloud environment. As organizations increasingly adopt containerization and multi-cloud strategies to enhance scalability, flexibility, and resource utilization, they encounter various obstacles that require effective solutions. The figure consists of nine categories, each representing a specific deployment challenge, along with the corresponding challenge description and its solution. These categories encompass a wide range of issues, including deployment complexity and Orchestration, access and communication control, security and compliance management, multi-cloud deployment coordination and Integration, monitoring and scalability challenges, microservices architecture and containerization, and network connectivity and hybrid cloud integration challenges.

Deployment Complexity and Orchestration category outlines various challenges and solutions related to the complexity and orchestration of container deployment in multi-cloud environment. Out of the 121 studies reviewed, 18 focused on these specific challenges and their respective solutions. Key challenges include Performance Testing Deployment, where standardized benchmarking and automation ensure consistency in performance testing. Deployment Validation is addressed by the Testing Process Management System, which enhances validation to ensure all criteria are met before deployment. Cost-Effective Deployment is achieved through Containerization and Microservice Architecture, enabling agile and efficient deployment. For Resource Management, a User Preference-Based Resource Brokering system optimizes resource allocation across clouds. Additional prominent solutions include orchestration using TOSCA, automation tools like Jenkins and Terraform, and Blockchain-based SLA Monitoring for immutable and auditable SLA tracking. Dynamic Adaptation is handled through Context-Aware Resource Allocation, while AWS Wavelength for Edge Cloud Integration addresses regional infrastructure consistency issues.

Access and Communication Control category organizes challenges and solutions related to maintaining secure connectivity in multi-cloud environment and managing user access control. Out of the 121 reviewed studies, 8 focused on these challenges. These studies report solutions include ensuring secure access to resources, secure communication between containers, and maintaining secure connectivity across multiple clouds. The solutions involve the development of encryption and authentication mechanisms to protect data and ensure authorized access. Secure communication protocols were recommended for interactions within containers and across cloud environments. It was also suggested that security measures should be provider-agnostic, ensuring that they are not tied to any single cloud provider.

Security and Compliance Management in Multi-cloud Deployment category focuses on security and compliance challenges and their solutions faced during multi-cloud container deployment. One challenge, Vendor Lock-in Prevention Deployment, aims to prevent vendor lock-in and ensure modular and loosely-coupled deployment in multi-cloud environment. The solution proposed is the Cloud Modelling Framework, which introduces a framework for specifying provisioning and deployment to enhance compatibility with existing ACSs and cloud solutions. Another challenge in this category is Security Deployment in Multi-container Environment, which addresses security and privacy concerns during multi-container deployment on the same OS kernel in multi-cloud environment. The solution is Cross-Container Isolation, which enforces cross-container isolation to enhance security.

Multi-cloud Deployment Coordination and Integration: This category identifies and classifies the challenges and solutions related to coordination and integration during the multi-cloud deployment phase. Of the 121 reviewed studies, 5 addressed these challenges. One such challenge is Multi-cloud PaaS Deployment, which seeks to overcome high entry barriers for deploying a PaaS infrastructure across multiple clouds. The proposed solution is Lightweight Proxies for PaaS Integration, which allows for the seamless integration of different PaaS services through lightweight proxies. One other challenge includes IoT Application Deployment, which presents issues when deploying IoT applications on multi-cloud platforms. The solution for this challenge is the DRA Framework and CI Broker for Multi-Cloud Deployment, which ensure smooth deployment and resource coordination across multiple clouds. Additionally, Data Transfer and Service Compatibility issues are managed using FaaS and storage services, ensuring that data transfer and compatibility between different cloud providers are not problematic. RBAM supports heterogeneous cloud and edge integration, offering frictionless management and operational flexibility for cloud–edge configurations. Lastly, private cloud integration is facilitated by the CLI tools that enable easy integration and the establishment of serverless environments.

Monitoring and Scalability Challenges category identifies and classifies various issues and solutions related to monitoring, performance, and scalability in the context of multi-cloud container deployment. Of the 121 reviewed studies, 8 specifically addressed these issues. One challenge is Legacy Code Migration Deployment, which involves managing the migration of legacy code, tenant engine separation, and composition modeling during container deployment across multiple clouds. The proposed solution is a Reusable Architectural Pattern with Docker and Cloudant, which enables the reuse of an architectural pattern using Docker for containerization and Cloudant for the persistence layer. Another challenge is Tenant Performance Deployment, which focuses on managing tenant service performance and competition during multi-cloud containerized deployments.

A proposed solution is the Multi-tenant and Multi-instance Hybrid Deployment scheme, based on container technology, which improves performance efficiency across tenants. A critical challenge is the monitoring of deployments, where existing tools have limitations in multi-cloud environment. The solution involves implementing JCatascopia, a platform-independent and interoperable monitoring system, which enhances the monitoring of multi-cloud setups and helps mitigate these challenges. Additional challenges include Resource Contention and QoS Degradation. The solutions include Dynamic CPU Adaptation through Linux patches, along with Kubernetes and performance monitoring tools. These measures ensure optimal resource usage and maintain performance consistency across cloud environments.

Microservices Architecture and Containerization category classifies the challenges and solutions related to microservices architecture and containerization in multi-cloud environment. Of the 121 reviewed studies, 5 explicitly addressed challenges in this area. One such challenge involves the deployment of multilayer and multitier Web architectures in multi-cloud environment, referred to as Autonomic System Deployment. The proposed solution is a Self-Tuning Performance Model and Autonomic Management, which introduces a self-tuning performance model along with an autonomic management system to optimize deployment and management. Another challenge focuses on Microservice Deployment, which explores how microservice architectures can be deployed in multi-cloud environment. The proposed solution is a Metrics and Requirements Framework for Microservices, emphasizing

Table 13
Identified deployment challenges and solutions for containerized applications in multi-cloud environment.

ID	Challenge	Solution
Category 1: Deployment complexity and orchestration challenges		
Almeida et al. (2014)	Performance Testing Deployment	Standardized Benchmarking and Automation
Verdugo et al. (2017)	Deployment Validation	Testing Process Management System
Jha et al. (2019)	Cost-effective Deployment	Containerization and Microservice Architecture
Pandey et al. (2022)	Resource Management Deployment	User Preference-Based Multi-Cloud Resource Brokering
Bhamare et al. (2015)	NFV Site Deployment	Site Selection and VF Allocation Algorithms
Serhiienko and Spillner (2018)	Cloud Management Platform Evaluation	Standardized Output Formats and Evaluation Criteria
Dimitrijevic et al. (2019)	Standardized Requirement Deployment	Application-Level Resource Managers for Multi-Cloud
Baresi et al. (2016)	Heterogeneous Node Deployment	Technology-Independent Multi-Level Adaptation
Casola et al. (2017)	Deployment Plan Generation	Cloud Services Selection and Deployment Planning
Tricomi et al. (2017)	TOSCA-based Deployment	Brokered Multi-Cloud Deployment with TOSCA
Sitaram et al. (2018)	Standardization Deployment	Proxy Cloud Virtualization for OpenStack
Barletta et al. (2024)	Criticality Constraints in Kubernetes Deployment	Kubernetes for Critical Deployments
da Silva et al. (2023)	Automation Tools Deployment Issue	Accurate Cost and Resource Tracking
Pandi et al. (2023)	Multi-cloud Deployment Challenge	Jenkins for Multi-cloud Automation
Sofia et al. (2024)	Decentralized App Deployment Issue	CODECO for Edge-Cloud Deployment
Neeraj et al. (2023)	SLA Monitoring Deployment Complexity	Blockchain for SLA Data Integrity
Petrovski and Gusev (2024)	Container and Serverless Architecture Deployment Issue	AWS ECS and Lambda Deployment
Solayman and Qasha (2023)	Heterogeneous Environment Deployment Challenge	Edge-cloud Orchestration Management
Sofia et al. (2023)	Dynamic Adaptation Deployment Issue	Context-aware Resource Allocation Strategies
Santos et al. (2023)	Regional Infrastructure Deployment Consistency Issue	AWS Wavelength for Edge Cloud Integration
Roy and Kalafatis (2023)	Robotic App Deployment Challenge	Hybrid Edge-Cloud Architecture
Alonso et al. (2023a)	Multi-cloud Deployment Consistency Problem	Multi-cloud Deployment Efficiency Models
Harzenetter et al. (2023)	Cross-Technology Deployment Integrity Issue	Workflow Automation for Cross-Technology Deployment
Huang and Wang (2023)	Industrial Robot Control System Deployment Problem	Seamless Control between Cloud and Robots
Category 2: Security and compliance management in multi-cloud deployment		
Ferry et al. (2014)	Vendor Lock-in Prevention Deployment	Cloud Modelling Framework
Gao et al. (2017)	Security Deployment in Multi-container Environment	Cross-Container Isolation
Rios et al. (2017)	Security Deployment	Security Control Specification and Deployment Framework
Alaluna et al. (2017)	Virtual Network Security Deployment	Network Virtualization Platform for Multi-Cloud
Wahab et al. (2016)	Malicious Service Management Deployment	Collusion-Resilient Trust Aggregation Technique
Barati et al. (2023)	Secure Personal Data Deployment Challenge	Docker, Kubernetes, Blockchain for GDPR Compliance
Fugkeaw (2023)	IAM Deployment across Cloud Platforms	Blockchain for Decentralized IAM Deployment
Wang et al. (2023)	Non-intrusive Malware Detection Deployment	DockerWatch for Non-intrusive Container Integration
Category 3: Multi-cloud deployment coordination and integration		
Sabbioni et al. (2020)	Multi-cloud PaaS Deployment	Lightweight Proxies for PaaS Integration
Bou Ghantous and Gill (2021)	IoT Application Deployment	DRA Framework and CI Broker for Multi-Cloud Deployment
Ramesh et al. (2023)	Data Transfer and Service Compatibility	FaaS and Storage for Multi-Cloud Deployment
Nguyen and Chien (2023)	Heterogeneous Cloud and Edge Integration	RBAM for Cloud-Edge Integration
Tiwari and Sharma (2023)	Private Cloud Integration Problem	CLI Tool for Serverless Setup
Category 4: Monitoring, performance, and scalability challenges		
Slominski et al. (2015)	Legacy Code Migration Deployment	Reusable Architectural Pattern with Docker and Cloudant
Wang et al. (2020)	Tenant Performance Deployment	Multi-Tenant and Multi-Instance Hybrid Deployment
Trihinas et al. (2015)	Monitoring Deployment	Platform-Independent Monitoring System
Struhár et al. (2024)	Resource Contention and Performance Problem	Linux Patch for Dynamic CPU Adaptation
Horchulhack et al. (2024)	QoS Degradation Detection Problem	Kubernetes with Performance Monitoring Tools
Marques et al. (2024)	Monitoring Integration in Hybrid Cloud Issue	Custom Scaling with Kubernetes
Kataru et al. (2023)	AWS Deployment Performance Consistency	AWS Model Comparison for Performance
Benmerar et al. (2023)	Cloud-Edge Deployment Performance Issue	AI-driven Orchestration for Dynamic Resource Management
Category 5: Microservices architecture and containerization challenges		
Barna et al. (2017)	Autonomic System Deployment	Self-Tuning Performance Model and Autonomic Management
Fadda et al. (2019)	Microservice Deployment	Metrics and Requirements Framework for Microservices
Pop et al. (2016)	Multi-cloud Application Configuration Deployment	Runtime Environment with Object Store and Artifact Repository
Quint and Kratzke (2018)	Elastic Container Platform Dependency Deployment	Separation of Elastic Platform and Cloud Application
Hossain et al. (2023)	Microservices Deployment in Edge Environments	Docker, CI/CD, Service Mesh for Microservices
Category 6: Network connectivity and hybrid cloud integration		
Carpio et al. (2020)	Inter-edge Bandwidth Deployment	Modular Edge Cloud Computing Architecture
Shi et al. (2020)	Network Latency Deployment	VM Selection for Composite Applications
Dreibholz et al. (2019)	Latency Reduction Deployment	Edge-Cellular Hybrid Infrastructure Provisioning
Pellegrini et al. (2017)	Vendor Lock-in Prevention Deployment	Interoperability Layer Above Cloud Infrastructure
Lee and Nam (2023)	Dynamic Network Policy Deployment Problem	KUNERVA for Network Policy Automation

(*continued on next page*)

Table 13 (*continued*).

Category 7: Cloud deployment constraints and challenges		
Pham et al. (2015)	ACS Compatibility Deployment	Scalable Multi-Cloud Deployment Framework
Sheridan et al. (2017)	Manual Intervention Risk Deployment	Automated Security Measures at Deployment
Wurster et al. (2018)	Multi-provider Deployment	TOSCA-Based Deployment Modeling Approach
Challita et al. (2017)	Cloud Provider Transition Deployment	Semantic Interoperability in Multi-Clouds
da Silva et al. (2023)	Automation Tools Deployment Issue	Accurate Cost and Resource Tracking
Santos et al. (2023)	Regional Infrastructure Deployment Consistency Issue	AWS Wavelength for Edge Cloud Integration
Category 8: Infrastructure provisioning and container deployment challenges		
Al-shammari and Alwan (2018)	Fog Computing Deployment	Reorganized Container Images and Docker Deployment
Sami and Mourad (2020)	Fog Device Deployment	On-Demand Fog and Microservices Deployment
Lorenzo et al. (2019)	Fog Computing Container Deployment	FogDocker for Container Deployment
Altran et al. (2022)	Distributed Compute Node Deployment	Label-Based Scheduling Strategy
Bélair et al. (2019)	Kernel Security Deployment	Enhanced Container Security
Barletta et al. (2024)	Criticality Constraints in Kubernetes Deployment	Kubernetes for Critical Deployments
Centofanti et al. (2024)	Power Tool Compatibility Problem	Docker, Kubernetes for Power Monitoring Integration
Ramesh et al. (2023)	Data Transfer and Service Compatibility	FaaS and Storage for Multi-Cloud Deployment
Category 9: Application deployment workflow and orchestration challenges		
Alonso et al. (2019)	API Management Deployment	DevOps Approach for Multi-Cloud Applications
Gupta et al. (2021)	Web Service Deployment	Kubernetes-Based Containerized Deployment
Buyya and Barreto (2015)	Complex Application Deployment	Aneka Platform for Distributed Applications
Acquaviva et al. (2017)	High-availability Deployment	Middleware Support for High Availability in Multi-Cloud PaaS
Kiss (2018)	Complex Workflow Deployment	CloudSME Simulation Platform
Kritikos et al. (2019)	Business Process Provisioning Deployment	Multi-Cloud Service Orchestration Frameworks
Roy and Kalafatis (2023)	Robotic App Deployment Challenge	Hybrid Edge-Cloud Architecture
Sofia et al. (2023)	Dynamic Adaptation Deployment Issue	Context-aware Resource Allocation Strategies

the need for a structured framework to develop requirements and relevant metrics for deploying microservice-based applications. For Multi-cloud Application Configuration, the challenges involve deployment, configuration, and operation across multi-cloud environment. The one solution involves deploying a Runtime Environment with an Object Store and Artifact Repository to effectively manage configurations across different clouds. The Elastic Container Platform Dependency Deployment solution addresses the challenge of vendor lock-in and dependency on specific container platforms. The solution is the Separation of Elastic Platform and Cloud Application Definitions, which separates platform definitions from cloud application definitions to increase flexibility of application deployment. Additionally, deploying microservices in an edge environment can help resolve integration and communication issues in heterogeneous edge computing environments. The use of Docker, CI/CD Pipelines, and Service Mesh for Microservices enables the automated and secure deployment and communication of microservices.

Network Connectivity and Hybrid Cloud Integration category classifies the challenges and solutions related to ensuring proper network connectivity in multi-cloud deployments and hybrid cloud integration. Out of the 121 studies reviewed, 5 specifically address these issues. One challenge, Inter-edge Bandwidth Deployment, deals with interconnecting distributed localities and managing bandwidth at the edge. The solution is a Modular Edge Cloud Computing Architecture, using containerization for better bandwidth management. A key challenge is Network Latency Deployment, which focuses on minimizing latency during multi-cloud deployments. The proposed solution is VM Selection for Composite Applications, optimizing virtual machine selection to reduce latency. Latency Reduction Deployment involves improving end-to-end performance through an Edge-Cellular Hybrid Infrastructure, combining edge and cellular networks. To address Vendor Lock-in, an Interoperability Layer Above Cloud Infrastructure enables smoother transitions between cloud providers.

Finally, Dynamic Network Policy Deployment tackles the challenge of adapting network policies to container workloads, with the solution being Automation of Network Policies using tools like KUNERVA.

Cloud Deployment Constraints and Challenges category identifies constraints and challenges in multi-cloud deployment, with 6 out of 121 studies addressing these issues. One challenge, ACS Compatibility Deployment, involves ensuring container compatibility with existing ACS systems, and the solution is a Scalable Multi-Cloud Deployment Framework. Other challenges include Manual Intervention Risk Deployment, which addresses risks from manual intervention. The solution is Automated Security Measures at Deployment to reduce vulnerabilities. The challenge of Multi-Provider Deployment involves managing multiple cloud providers. The proposed solution is the TOSCA-Based Deployment Modeling Approach, which provides a unified framework. The challenge of Cloud Provider Transition Deployment lies in addressing feature incompatibilities. The proposed solution is Semantic Interoperability in Multi-Clouds, which ensures smooth transitions between providers. Automation Tools Deployment challenge is solved with tools like Jenkins and Terraform for accurate cost and resource management.

Infrastructure Provisioning and Container Deployment Challenges category classifies the challenges and solutions associated with infrastructure provisioning and container deployment in multi-cloud settings. Of the 121 reviewed studies, 8 focused on these challenges. One such challenge is Fog Computing Deployment, which deals with issues related to fog computing hardware and container deployment time. The proposed solution is Reorganized Container Images and Docker Deployment, suggesting that container images should be reorganized, and the Docker deployment process is modified to handle fog computing more effectively, enabling better resource management and optimization. Another challenge is Fog Device Deployment, which focuses on managing fog device availability during multi-cloud container deployments. The proposed solution is On-Demand Fog and Microservices Deployment, allowing for the on-demand creation of fog nodes and the on-demand deployment of microservices via Docker and Kubeadm. Distributed Compute Node Deployment involves challenges related to containerized application management in distributed compute nodes. Kernel Security Deployment addresses the security risks of kernel sharing and isolation. The proposed solution includes a Label-Based Scheduling Strategy and Enhanced Container Security for secure and efficient deployment.

Application Deployment Workflow and Orchestration Challenges category classifies the challenges and solutions related to application deployment workflows and orchestration in multi-cloud environment. Of the 121 reviewed studies, 7 focused on these challenges. One such challenge, API Management Deployment, managing the APIs and interfaces provided by different cloud providers in multi-cloud container deployments. The proposed solution is a DevOps Approach for Multi-Cloud Applications, which suggests using DevOps to manage application lifecycles in multi-cloud environment. Another challenge,

Web Service Deployment, involves ensuring proper and efficient deployment in heterogeneous computing environments. The solution is Kubernetes-Based Containerized Deployment, which uses Kubernetes to efficiently manage and deploy containerized environments. Complex Application Deployment and High-availability Deployment focus on deploying complex applications while ensuring high availability. The solutions include the Aneka Platform for Distributed Applications and Multi-cloud PaaS Middleware Support for High Availability, which ensure the availability and distribution of applications across multi-cloud environment.

Takeaway 10: Reviewed studies show that deployment challenges in containerized multi-cloud environments emerge from the tight coupling of container orchestration with heterogeneous cloud infrastructures. Solutions such as TOSCA-based modeling, cross-container isolation, and context-aware adaptation specifically address complexities introduced when containers span multiple clouds, highlighting the unique integration, security, and coordination demands identified through this SMS.

3.8. Monitoring challenges and solution framework (RQ4 and RQ5)

Table 14 presents a categorized overview of monitoring challenges and corresponding solutions for containerized applications in multi-cloud environments. The framework captures a range of challenges related to performance variability, resource utilization, observability, and integration. Each of the six categories highlights a specific aspect of monitoring and summarizes relevant solutions proposed in the literature.

Performance, Consistency, and Variability Monitoring category identifies the challenges and solutions that ensure consistent performance monitoring in dynamically changing multi-cloud environment. Nine of the reviewed studies addressed these challenges. Performance Monitoring focuses on maintaining consistent performance tracking of applications. The solution, Performance Variation Monitoring, uses benchmarking systems to track and analyze runtime changes in performance, ensuring optimized performance throughout. Another challenge is Benchmarking for Performance and Cost Efficiency, which balances performance against cost. The solution, SBDO Optimization, optimizes resource utilization in real time to maximize efficiency and minimize costs. A related challenge, Website Performance Monitoring, is managed through Kubernetes Monitoring, which ensures uptime and performance across widely distributed cloud environments. Monitoring User Experience is handled by collecting CSP QoS Metrics, allowing cloud service providers to gather Quality of Service data, anticipate variations in performance, and manage the overall user experience effectively. Lastly, low-latency performance optimization in multi-cloud systems is addressed by Tail Latency Prediction, which uses predictive models to minimize network delays and ensure smooth application performance.

Resource and Infrastructure Monitoring category classifies the challenges and corresponding solutions related to monitoring resource and infrastructure management for containerized applications in multi-cloud environment. Of the reviewed studies, 8 specifically addressed these challenges. One key challenge is Network Communication Monitoring, which involves managing service-to-service communication across cloud networks. The proposed solution is the implementation of a Service Mesh, which enhances visibility and security for network services. Another challenge is System Image and Configuration Monitoring, which tracks software updates, configuration changes, and performance across platforms. The solution proposed is Docker Server Deployment, which allows for easy updates to the containerized system. For Vcluster and Framework Management Monitoring, the solution is the implementation of LXC (Linux Containers), which manages application isolation and performance using lightweight containers. Multi-Cloud Resource Utilization Monitoring involves tracking the performance of resources, such as compute, storage, and network, used across multiple clouds. The solution uses Prometheus and Grafana for monitoring resource usage and availability, with an emphasis on optimization. Similarly, Dynamic Resource Management Monitoring solutions, such as the Aneka Platform, dynamically add or release resources based on demand. The challenge of cost tracking in resource utilization across multiple cloud environments, such as AWS and Azure, is addressed by the COSTA system, which monitors both resource utilization and cost efficiency. Finally, Prometheus is used for Real-Time Dynamic Resource Monitoring, allowing real-time data collection and analysis to optimize performance based on resource usage fluctuations.

Application Optimization and Adaptability Monitoring: This category identifies and discusses challenges related to optimizing and adapting applications in multi-cloud environment. Out of the 121 reviewed studies, 8 specifically addressed these challenges with corresponding solutions. Key challenges include Real-Time Application Monitoring, which focuses on taking immediate corrective actions using the DRA Framework, and Microservice Optimization Monitoring, which enhances monitoring across multiple clouds through MiCADO Optimization. Other challenges include Partial Download Execution Monitoring, which ensures the correct execution of partially downloaded files. The proposed solution for runtime accuracy is the FogDocker Implementation. Another challenge, Event-Driven Application Behavior Monitoring, involves monitoring event-driven application behavior. The TOSCA Event Modeling solution efficiently handles application events. For Adaptive System Design and State Monitoring, continuous design and dynamic system adaptability are managed using the Cloud Modelling Framework. Additionally, the challenge of Container Application Adaptability Monitoring is addressed through advanced orchestration techniques, which provide real-time views of distributed applications. The solution involves using AWS tools to ensure high performance for different deployment models, addressing challenges in Serverless and Containerized Monitoring. AWS tools are recommended to ensure high performance for different deployment models, addressing challenges in Serverless and Containerized Monitoring.

System Complexity and Standardization Monitoring category identifies and addresses the challenges of managing system complexity and standardization in monitoring solutions across multi-cloud environment. Of the 121 studies reviewed, 4 specifically focused on these challenges and proposed corresponding solutions. A key challenge is the standardization of proprietary monitoring solutions, for which the proposed solution is Pervasive Monitoring. Pervasive Monitoring focuses on implementing pervasive monitoring techniques for containerized applications within multi-cloud PaaS platforms. Another significant challenge is Kernel Resource Isolation Mechanism Monitoring, with the proposed solution being Leakage Defense. This solution addresses in-container leakage channels by implementing a two-stage isolation mechanism in Linux-based systems. Additionally, the challenge of Multi-Cloud Application Monitoring Complexity is addressed by the DECIDE DevOps Expansion, which extends the framework for analyzing multi-cloud containerized applications. The Autonomic Management Framework further enhances monitoring through Monitoring Agent Coordination, improving the coordination of monitoring agents using a performance model and adaptive actions for better system behavior within containerized applications.

Multi-cloud Coordination and Integration Monitoring category identifies and addresses the challenges related to coordinating and integrating monitoring data across multiple cloud environments. Out of the 121 studies reviewed, 5 focused on these challenges. One key challenge is Multi-cloud Monitoring Improvement, and the proposed solution is the use of Self-Healing Techniques to autonomously detect and correct issues within cloud infrastructures. Another challenge is Cross-Level Monitoring, which is addressed through Business Process Monitoring Integration to ensure that business processes are effectively

Table 14
Identified monitoring challenges and solutions for containerized applications in multi-cloud environment.

ID	Challenge	Solution
Category 1: Performance monitoring consistency and variability		
Almeida et al. (2014)	Performance Monitoring	Performance Variation Monitoring
Jha et al. (2019)	Benchmarking Performance Variation and Cost Efficiency Monitoring	SDBO Optimization
Gupta et al. (2021)	Website Performance and Availability Monitoring	Kubernetes Monitoring
Pandey et al. (2022)	User Experience Variability Monitoring	CSP QoS Metrics Collection
Dreibholz et al. (2019)	Independent Performance Metrics Monitoring	Performance Evaluations
Bhamare et al. (2015)	Comprehensive Network Performance Measurement Monitoring	Server Selection Scheme
Rahman and Lama (2019)	Microservice Performance Degradation Monitoring	Tail Latency Prediction
Struhár et al. (2024)	Container Performance Metrics Tracking	CLM and OSLM for Resource Adjustments
Horchulhack et al. (2024)	Container Performance and Resource Usage Monitoring	Metric and Container Monitoring Tools
Hossain et al. (2023)	Microservice Performance Monitoring Challenge	Prometheus, Grafana, Kubernetes for Monitoring
Ramesh et al. (2023)	Cross-Cloud Service Performance Monitoring	FaaS and Storage Services for AI Monitoring
Nguyen and Chien (2023)	Stream Processing Performance Monitoring	RBAM for Real-Time Performance Insights
Kataru et al. (2023)	AWS Application Performance Monitoring	AWS CloudWatch for Performance Monitoring
Alonso et al. (2023a)	Multi-Cloud Performance Visibility Issue	Monitoring Tools for Multi-Cloud Visibility
Harzenetter et al. (2023)	Cross-Technology Performance Monitoring Issue	TOSCA-based Workflows for Monitoring Applications
Huang and Wang (2023)	Industrial Robot System Monitoring Challenge	Real-Time System Performance Tracking Tools
Category 2: Resource and infrastructure management monitoring		
Gattobigio et al. (2022)	Network Communication Monitoring	Service Mesh Implementation
Verdugo et al. (2017)	System Image and Configuration Monitoring	Docker Server Deployment
Cui et al. (2016)	Vcluster and Framework Management Monitoring	LXC Implementation
Pahl (2015)	Multi-Cloud Resource Utilization Monitoring	Prometheus and Grafana Monitoring
Buyya and Barreto (2015)	Dynamic Resource Management Monitoring	Aneka Platform
da Silva et al. (2023)	Cost and Resource Usage Monitoring	COSTA for Cost and Resource Monitoring
Dogani et al. (2023)	Real-Time Dynamic Resource Monitoring	Prometheus for Real-Time Monitoring
Wong et al. (2023)	Resource Usage and Performance Issue	Prometheus for Real-Time Data Collection
Marques et al. (2024)	Real-Time Container Monitoring	Prometheus with Forecasting for Resource Management
Santos et al. (2023)	Distributed Cloud Resource Monitoring Issue	AWS CloudWatch and Azure Monitor for Insights
Nagarajan et al. (2023)	Resource Allocation and Energy Monitoring Issue	Tools for Resource, Energy, and SLA Monitoring
Category 3: Security, compliance, and trust management monitoring		
Ramos et al. (2021)	Provider Host Information Access Monitoring	ML Monitoring
Rios et al. (2017)	Continuous Security Control Monitoring	MUSA Platform
Casola et al. (2017)	Multi-Cloud Security Compliance Monitoring	MUSA Security Compliance
Acquaviva et al. (2017)	Service Continuity and Data Privacy Monitoring	NoMISHAP Implementation
Wahab et al. (2016)	Trust Relationship and Network Contact Monitoring	Trust Bootstrapping
Fugkeaw (2023)	Access and Authentication Monitoring Issue	Blockchain for Secure Access Event Logging
Wang et al. (2023)	Malware Behavior and System Monitoring	DockerWatch for Container Behavior Monitoring
Neeraj et al. (2023)	Log Data Integrity Monitoring Issue	Blockchain-based Logs for SLA Monitoring
Category 4: Application optimization and adaptability monitoring		
Ferry et al. (2014)	Adaptive System Design and State Monitoring	Cloud Modelling Framework
Lorenzo et al. (2019)	Partial Download Execution Monitoring	FogDocker Implementation
Wurster et al. (2018)	Event-Driven Application Behavior Monitoring	TOSCA Event Modeling
Guerrero et al. (2018)	Microservice Optimization Monitoring	MiCADO Optimization
Bou Ghantous and Gill (2021)	Real-Time Application Monitoring	DRA Framework
Sofia et al. (2023)	Container Application Adaptability Monitoring	Advanced Monitoring with Orchestration Integration
Petrovski and Gusev (2024)	Serverless and Containerized Monitoring Challenge	AWS Tools for Performance Monitoring
Category 5: System complexity and standardization monitoring		
Sabbioni et al. (2020)	Proprietary Monitoring Solution Standardization	Pervasive Monitoring
Gao et al. (2017)	Kernel Resource Isolation Mechanism Monitoring	Leakage Defense
Alonso et al. (2019)	Multi-Cloud Application Monitoring Complexity	DECIDE DevOps Expansion
Barna et al. (2017)	Monitoring Agent Coordination	Autonomic Management
Sami and Mourad (2020)	Heterogeneous Fog Resource Management Monitoring	Kubernetes Utility Architecture
Baresi et al. (2016)	Application Topology and Dependency Monitoring	MicroCloud TOSCA Library
Trihinas et al. (2015)	Elastic Cloud Service Monitoring Research	JCatascopia Monitoring
Fadda et al. (2019)	Microservice Monitoring Diversity	QoM Definition
Harwalkar et al. (2019b)	Compute and Network Resource Utilization Monitoring	Time Series Resource Utilization
Ahmad and Aujla (2023)	Real-Time Data Operations Monitoring	Smart Contracts for GDPR Monitoring
Centofanti et al. (2024)	Power Consumption Monitoring Issue	Prometheus with Power Monitoring Integration
Tiwari and Sharma (2023)	Serverless Function Monitoring Complexity	CLI Tool for Function Monitoring
Category 6: Multi-cloud coordination and integration monitoring		
Sitaram et al. (2018)	Integrated Multi-Cloud Management Monitoring	Gnocchi Integration
Mfula and Nurminen (2018)	Multi-Cloud Monitoring Challenges	Self-Healing Techniques
Kritikos et al. (2019)	Cross-Level Monitoring Improvement	Business Process Monitoring
Lee and Nam (2023)	Real-Time Network Policy Monitoring Challenge	KUNERVA for Real-Time Network Policy Monitoring
Sofia et al. (2024)	Decentralized System Observability Issue	CODECO for Orchestration and Data Observability

monitored across multiple cloud platforms. Lastly, Integrated Multi-cloud Management Monitoring requires advanced solutions, such as Integration with Gnocchi, to efficiently gather and monitor metrics from diverse cloud resources.

Security, Compliance, and Trust Monitoring: category addresses the challenges and solutions related to monitoring security, compliance, and trust management for containerized applications in multi-cloud environment. Out of the 121 reviewed studies, 9 specifically deal with these concerns. One challenge is Provider Host Information Access Monitoring, where limitations in accessing host data are overcome using ML Monitoring, which employs machine learning to track system performance and usage patterns.

Another challenge, Continuous Security Control Monitoring, is addressed by the MUSA platform, which enforces and continuously monitors security in real time for containerized applications. Multi-Cloud Security Compliance Monitoring ensures continuous security compliance across multi-cloud environment. This is achieved through the MUSA Security Compliance Solution, a platform for continuous compliance management. The challenges of maintaining data privacy, fault tolerance, and service continuity in multi-cloud environments are addressed under Service Continuity and Data Privacy Monitoring. The NoMISHAP Implementation is proposed to handle these monitoring issues. Additionally, Trust Bootstrapping establishes trust through a chain of endorsement and decision tree classification. Access and Authentication Monitoring uses blockchain technology to enable secure and transparent logging of authentication activities. For Malware Behavior and System Monitoring, DockerWatch monitors container behavior with minimal performance overhead.

Takeaway 11: Reviewed studies reveal that monitoring containerized applications in multi-cloud environments introduces unique challenges, such as inconsistent performance metrics across cloud providers, real-time trust and security event tracking, and fragmented observability due to container orchestration spanning heterogeneous infrastructures. These challenges are addressed through specialized solutions like cross-cloud Prometheus-Grafana stacks, blockchain-based SLA logging, and service mesh integrations, reflecting the need for tightly coupled monitoring mechanisms tailored specifically to the hybrid nature of containerized multi-cloud deployments.

3.9. Tools and frameworks (RQ6)

In response to RQ6 regarding tools and frameworks for developing containerized multi-cloud applications, we identified 87 distinct tools and frameworks, classified into 5 subcategories (see Table 15 and the Tools and Frameworks sheet in Waseem et al. (2025)). Below, we provide an overview of each category, along with key tools and frameworks, highlighting their functionality, adoption, and distinctive features.

Container Orchestration and Management Tools: are essential for automating the deployment, scaling, and management of containerized applications across distributed and multi-cloud environment. These tools assist developers in managing large-scale container deployments and efficiently allocating resources across different cloud providers. Among the most popular orchestration and management tools, *Kubernetes* is widely adopted for orchestrating containers at scale, providing high availability, and facilitating load balancing across nodes. *Docker* and its variants, such as *Docker Swarm*, offer more convenient container orchestration but are less scalable than *Kubernetes*. *Mesos* is another orchestration tool that supports not only containers but also other workloads, such as big data processing tasks. Tools like *LXC* and *runC* provide the runtime environment to run a single container with fine-grain control. Overall, this category covers key orchestration and container management tools that address diverse deployment needs in multi-cloud environment.

Multi-cloud and Hybrid Cloud Platforms are designed to support organizations in deploying and managing applications across different cloud providers smoothly. In this category, we identify and organize tools like *Cloudify* and *CMP2*, which offer end-to-end multi-cloud orchestration capabilities, along with platforms such as *Cloudcheckr* and *MistIO*, which provide monitoring and cost management across various cloud services. Similarly, *AWS SDK* and the *CloudSME Simulation Platform (CSSP)* are widely adopted for interacting with cloud APIs and simulating cloud environments, respectively. Moreover, hybrid platforms like *OpenStack* and *VMware vSphere* offer a blend of public and private cloud deployment options, which are essential for organizations pursuing a hybrid cloud strategy. This category highlights the tools that enable integration, scalability, and cost management for applications running across multiple clouds.

Networking, Security, and Access Management Tools: This category represents the tools that help to minimize security risks and ensure connectivity across containerized workloads in multi-cloud systems. Advanced networking tools like Open vSwitch (OvS) allow developers to manage virtual networks across multiple cloud platforms as multi-cloud deployments grow more complicated. By implementing stringent access controls and sandboxing, security tools like Seccomp-BPF and Linux Security Modules (LSM) offer vital container protection. Proper authentication and authorization for cloud users and services are ensured by identity management solutions like Host Identity Protocol (HIP) and access control policies.

Monitoring, Management, and Automation Tools: Tools in this category help manage large-scale cloud applications in an easy and automated way. They make sure everything runs smoothly by constantly checking and improving how resources are used. For example, *Prometheus* and *Grafana* help system administrators by showing important information like CPU use, memory load, and network traffic in clear charts and dashboards. Tools like *Terraform* and *Puppet* make it possible to write and reuse code to set up servers and other infrastructure automatically across different cloud services. This saves time and reduces errors. In big systems using OpenStack, tools such as *Ceilometer* and *Gnocchi* collect data about how the system is working, which helps with tracking performance and solving issues.

Development, APIs, and Cloud Storage Solutions: This category organizes the main tools that support software development, API integration, and data storage in containerized multi-cloud environment. The tools include *REST API* and *YAML*, which enable services and applications deployed across multiple cloud providers to communicate for data sharing and inter-service interactions. General programming languages such as *Java*, *Go*, and *Python* are commonly used to develop applications based on cloud-native architecture, alongside tools for version control, such as *GitHub*, and project management using *Maven*. On the data storage side, tools like *gcloud storage* and *azblob* provide scalable and fault-tolerant systems for managing large datasets in cloud environments.

Takeaway 12: The reviewed studies reveal that tools and frameworks commonly used in general container or cloud settings (e.g., Kubernetes, Docker, Terraform, Prometheus) require specific adaptations or configurations to operate effectively in containerized multi-cloud environments. This includes challenges such as consistent orchestration across heterogeneous cloud APIs, maintaining observability across isolated infrastructures, and enabling secure, policy-compliant automation at scale. Moreover, hybrid-enabling tools like OpenStack, Cloudify, and MistIO emerged as critical for unifying management across cloud boundaries — highlighting how the combination of containerization and multi-cloud imposes unique operational and integration demands not encountered when using either paradigm alone.

Table 15
Tools and frameworks for containerized applications in multi-cloud environment.

Category	Tool, Framework, and Language
Container Orchestration and Management	Kubernetes (Pai and Kumar, 2021; Wong et al., 2019; Gupta et al., 2021; Bolivar et al., 2018; del Castillo et al., 2013; Rodigari et al., 2021; Rahman and Lama, 2019; Pellegrini et al., 2017; Quint and Kratzke, 2018; Barletta et al., 2024; Struhár et al., 2024; Barati et al., 2023; Horchulhack et al., 2024; Dogani et al., 2023; Wong et al., 2023; Marques et al., 2024; Orzechowski et al., 2023; Merlino et al., 2024; Ahmad and Aujla, 2023; Centofanti et al., 2024; Hossain et al., 2023; Liu and Qin, 2024; Tundo et al., 2024; Pandi et al., 2023; Sofia et al., 2024; Lee and Nam, 2023; Sofia et al., 2023; Nguyen and Chien, 2023; Tiwari and Sharma, 2023; Santos et al., 2023; Roy and Kalafatis, 2023; Kataru et al., 2023; Benmerar et al., 2023; Alonso et al., 2023a; Harzenetter et al., 2023; Huang and Wang, 2023; Wang et al., 2023; Nagarajan et al., 2023), Docker (Pham et al., 2015; Naik, 2016a; Ouyang et al., 2022; Slominski et al., 2015; Pai and Kumar, 2021; Gao et al., 2017; Alonso et al., 2019; Barna et al., 2017; Gupta et al., 2021; Sheridan et al., 2017; Mittal and Risco-Martín, 2017; Al-shammari and Alwan, 2018; Carpio et al., 2020; Bolivar et al., 2018; Rodigari et al., 2021; Rahman and Lama, 2019; Mfula and Nurminen, 2018; Ranjbar et al., 2017; Serhiienko and Spillner, 2018; Quint and Kratzke, 2018), Docker Swarm (Pai and Kumar, 2021; Wong et al., 2019; Altran et al., 2022; Rodigari et al., 2021; Quint and Kratzke, 2018), Mesos (Cui et al., 2016; Wong et al., 2019; Rodigari et al., 2021; Quint and Kratzke, 2018), LXC (Pahl, 2015; Gao et al., 2017; Gupta et al., 2021), OverlayFS (Al-shammari and Alwan, 2018), runC (Bélair et al., 2019), Container Technology (Wang et al., 2020), Docklet (Cui et al., 2016)
Multi-cloud and Hybrid Cloud Platforms	Multi-cloud environment (Shi et al., 2020; Fadda et al., 2019; Di Modica et al., 2019; Tang, 2021), Cloudify (Zeck and Bouroudjian, 2017), CloudBroker Platform (Kiss, 2018), Cloudcheckr (Serhiienko and Spillner, 2018), MistIO (Serhiienko and Spillner, 2018), ManageIQ (Tang, 2021; Serhiienko and Spillner, 2018), CMP2 (Serhiienko and Spillner, 2018), AWS SDK (Graupner et al., 2015), CloudSME Simulation Platform (CSSP) (Kiss, 2018), Vagrant (Pham et al., 2015), AWS OpsWorks (Pham et al., 2015), Universal Compute Xchange (UCX) (Zeck and Bouroudjian, 2017), Liferay portlets (Kiss, 2018), Cloud Service Providers (CSPs) (Graupner et al., 2015), AWS Data Pipeline (Sitaram et al., 2018), GENI (Pandey et al., 2022), Google Cloud Platform (Pandey et al., 2022; Acquaviva et al., 2017; Tang, 2021; Quint and Kratzke, 2018), AWS EC2 (He et al., 2022; Buyya and Barreto, 2015; Pai and Kumar, 2021), Alibaba cloud (Zeck and Bouroudjian, 2017; Quint and Kratzke, 2018), Tencent (Zeck and Bouroudjian, 2017; Quint and Kratzke, 2018), OpenStack (Cui et al., 2016; Dreibholz et al., 2019), VMware vSphere (Mfula and Nurminen, 2018), Aliyun ECS (Hua et al., 2020), Baidu BCE (Hua et al., 2020), Tencent CW (Hua et al., 2020), JD VW (Hua et al., 2020), GoGrid (Buyya and Barreto, 2015)
Networking, Security, and Access Management	Open vSwitch (OvS) (Alaluna et al., 2017), Cloud networks (Ranjbar et al., 2017), Host Identity Protocol (HIP) (Ranjbar et al., 2017), Gateways (Harwalkar et al., 2019a), Pledge (Bélair et al., 2019), Seccomp-BPF (Bélair et al., 2019), Landlock LSM (Bélair et al., 2019), Linux Security Modules (LSM) (Bélair et al., 2019), MUSA Security Assurance Platform (Rios et al., 2017), Firewall Deployment (Sheridan et al., 2017), Access control policies (Gao et al., 2017), Attribute-Based Encryption and Signature (Shen et al., 2017), ABE/ABS cryptographic model (Shen et al., 2017)
Monitoring, Management, and Automation	Prometheus (Wong et al., 2019; Dogani et al., 2023; Wong et al., 2023; Marques et al., 2024; Orzechowski et al., 2023; Merlino et al., 2024; Ahmad and Aujla, 2023; Centofanti et al., 2024; Hossain et al., 2023; Liu and Qin, 2024; Tundo et al., 2024; Roy and Kalafatis, 2023), Grafana (Alonso et al., 2019; Hossain et al., 2023; Liu and Qin, 2024; Tundo et al., 2024; Roy and Kalafatis, 2023), Telegraf (Alonso et al., 2019), AWS CloudWatch (Alhamazani et al., 2014), cAdvisor (Wong et al., 2019), Ceilometer (del Castillo et al., 2013; Sitaram et al., 2018), Gnocchi (Sitaram et al., 2018), ELK stack (Mfula and Nurminen, 2018), Terraform (Alonso et al., 2019; Gupta et al., 2021; da Silva et al., 2023; Pandi et al., 2023; Harzenetter et al., 2023), CloudBees (Pham et al., 2015), Jenkins (Pandey et al., 2022; Solayman and Qasha, 2023), Puppet (Pham et al., 2015), Chef (Pham et al., 2015; Sheridan et al., 2017), GitHub (Sheridan et al., 2017), Shell scripts (Rios et al., 2017), Maven (Pham et al., 2015), NPM (Pham et al., 2015), Consul (Pham et al., 2015)
Development, APIs, and Cloud Storage	Public cloud storage (Harwalkar et al., 2019a), Private cloud storage (Harwalkar et al., 2019a), Hybrid cloud storage (Harwalkar et al., 2019a), Google Cloud Storage (Ramesh et al., 2023), Azure Blob Storage (Ramesh et al., 2023), REST API (Harwalkar et al., 2019a), YAML (Serhiienko and Spillner, 2018), OpenVAS Vulnerability Scanner (Sheridan et al., 2017), DAX format Tang (2021), Java (Sheridan et al., 2017; Rios et al., 2017; Carpio et al., 2020; Bucur et al., 2018; Quint and Kratzke, 2018; Challita et al., 2017), Go programming language (Al-shammari and Alwan, 2018; Lorenzo et al., 2019), Python (Carpio et al., 2020; Bucur et al., 2018; del Castillo et al., 2013; Sitaram et al., 2018; Serhiienko and Spillner, 2018), Ruby Bucur et al. (2018), PHP (Bucur et al., 2018), C# (Bucur et al., 2018), Cloud Modelling Framework (CLOUMDF) (Ferry et al., 2014), Cloud Modelling Language (CLOUDML) (Ferry et al., 2014; Challita et al., 2017), CAMEL modeling language (Casola et al., 2017; Quint and Kratzke, 2018; Kritikos et al., 2019), Eclipse IDE (Alonso et al., 2019), MCSLA editor (Alonso et al., 2019)

4. Discussion

The following discussion synthesizes key findings for each research question and their implications for research and practice. These implications also highlight open problems and unresolved challenges that require further research attention, particularly those emerging from the intersection of containerization and multi-cloud environments.

4.1. Containers in multi-cloud: Roles and strategies (RQ1)

Containers play a crucial role in the development and operation of modern software systems within multi-cloud environment. Understanding the various roles and implementation strategies of containers provides significant insights for both researchers and practitioners.

Multifaceted Role of Containers in Multi-Cloud Environment: In a multi-cloud environment, containers serve multiple purposes, such as managing resources, hosting services, and deploying applications. They simplify the deployment of apps across various platforms, including web applications and Internet of Things (IoT) applications. Additionally, containers enhance security in multi-cloud operations. These roles are shaped by the inherent heterogeneity of multi-cloud settings, where containers are used to ensure portability, manage provider-specific constraints, and support workload migration across isolated cloud infrastructures. **Implications**: These findings open up opportunities for researchers to explore additional areas, including security vulnerabilities, management challenges, and performance overhead. Likewise, practitioners can apply these insights to design more efficient and effective deployment strategies, ultimately contributing to stronger and more reliable multi-cloud implementations.

Container Technology Features and their Significance: Container technology offers several benefits due to its unique features, such as support for edge computing, compatibility with orchestration tools, and lightweight virtualization. In multi-cloud environments, these features enable containers to act as a unifying abstraction layer across diverse infrastructure stacks, allowing consistent deployment behavior despite differing cloud-specific configurations. **Implications**: Researchers can explore these features further, including how characteristics like lightweight virtualization influence container performance and effectiveness in multi-cloud operations. Practitioners can use this

understanding for better utilization of container technology and more efficient multi-cloud deployments.

Implementation Strategies and their Real-World Implications: The strategic implementation of containers significantly affects their functionality in multi-cloud environment. Strategies related to deployment, orchestration, security, performance optimization, cloud service management, and data storage can have significant implications. The reviewed strategies demonstrate how container management must adapt to cloud-specific APIs, variable service guarantees, and cross-cloud policy enforcement, revealing how the interplay between containerization and multi-cloud governance introduces distinctive design trade-offs. **Implications**: Researchers may explore these strategies further, including investigations into their real-world implications, benefits, and limitations. Practitioners can refine their understanding of these strategies to achieve more efficient implementations and operations.

Containerization's Impact on IoT and Application Development: The incorporation of containers into IoT and app development has transform these fields. Containers offer improved software unit encapsulation, portability, and efficiency, which are particularly important in the distributed and resource-constrained nature of IoT. In multi-cloud scenarios, this impact is increase by the need to coordinate deployments across heterogeneous edge-cloud setups and maintain interoperability between varied cloud-native services and orchestration frameworks. **Implications**: Researchers can further explore potential challenges in detail such as network complexity, security, and the small footprint requirement of IoT devices. For practitioners, understanding the impact of containerization can lead to the development of more robust, efficient, and secure IoT and app solutions.

4.2. Patterns and strategies (RQ2)

Containerized applications in multi-cloud environment requires an understanding of prevailing patterns and strategies. In this context, we discuss the key findings related to the identified patterns and strategies presented in Table 9.

Diverse Architectural Patterns and Communication Strategies: Our study found that Microservice Architecture and Service-Oriented Architecture (SOA) are the most common used architectural patterns for designing container-based applications in multi-cloud environment. These results align with the existing literature that emphasizes the use of MSA and SOA because of their support for modular development, scalability, and efficient resource use (Balalaie et al., 2016; Dragoni et al., 2017). The other patterns frequently reported are Service Mesh and Service Chaining, which are employed as communication and networking patterns, likely due to their ability to manage complex, distributed service-to-service communications in a scalable way (Zhamak, 2018). The frequent use of black-Green Deployment and Object Store Service deployment patterns has also been identified in this study, which may stem from their ability to minimize downtime and ensure high availability—essential factors in a cloud-based setting (Fowler, 2010). Notably, these architectural and communication patterns are adapted in the multi-cloud setting to support distributed orchestration, cross-cloud resilience, and integration across heterogeneous platforms—attributes that are less emphasized in single-cloud deployments. For instance, service mesh implementations in multi-clouds enable secure communication across isolated cloud boundaries, and decentralized orchestration patterns accommodate independently governed cloud nodes. **Implications**: Practitioners can enhance the design, scalability, and efficiency of their applications in a multi-cloud environment by understanding and applying these architectural and communication patterns. Researchers might explore how these patterns evolve with the emergence of new technologies and standards.

Importance of Container Management and Multi-Cloud Strategies: Our findings report the significance of robust container management and multi-cloud strategies. Techniques like the Linux Container (LXC) project, Docker Container Images, and Lightweight Virtualization are frequently used, emphasizing the industry trend towards containerization due to its benefits in efficiency, scalability, and platform independence (Vaughn, 2020). Furthermore, the popularity of multi-cloud strategies like Hybrid/Multi-cloud Approach and Multi-cloud Load Balancing confirms the growing industry move towards using multiple cloud providers to enhance redundancy, flexibility, and avoid vendor lock-in Gillam et al. (2013). In particular, the reviewed studies illustrate how container technologies are adapted to meet challenges specific to multi-cloud deployments, such as orchestrating containers across provider-specific APIs, aligning container runtime environments across clouds, and ensuring policy compliance in cross-provider setups. These represent key peculiarities introduced by multi-cloud environments into the containerization landscape. **Implications**: Practitioners should focus on gaining expertise in container technologies and multi-cloud strategies to take advantage of their numerous benefits. Meanwhile, researchers might examine the challenges and best practices associated with the deployment and management of container technologies across multiple cloud platforms.

Security, Resilience, and Migration Patterns and Strategies: Our study also identifies patterns and strategies related to security, resilience, and efficient migration strategies in a multi-cloud context. Strategies such as File Encryption, Attribute-Based Encryption, and Secure Data sharing are often employed to ensure data security, an area of growing concern with the increase in cloud-based applications (Svantesson and Clarke, 2010). Moreover, resilience strategies such as Fault-tolerance with Redundant Engines and RAFT Consensus Algorithm are essential to ensure system reliability in distributed, multi-cloud environment (Ongaro and Ousterhout, 2014). Migration between different cloud environments, facilitated by strategies like Service-oriented Migration and Migration to a Virtualized Container, is another vital aspect, given the growth in cloud services and the need for interoperability (Jamshidi et al., 2013). Migrating applications and data between cloud environments is an important consideration in a multi-cloud scenario (Jamshidi et al., 2013). Strategies such as image-based migration and migration to virtualized containers, like Docker, are commonly used (Vaughn, 2020). These techniques facilitate portability and provide consistent environments across different cloud platforms. Additionally, they have significant implications for scaling and managing applications across different cloud service providers. Unlike traditional container security or resilience strategies in single-cloud contexts, the reviewed studies show that in multi-cloud scenarios, these approaches must handle cross-cloud identity management, consistency in fault detection and recovery across isolated infrastructures, and security rule federation across providers. Similarly, migration patterns uniquely address compatibility mismatches and compliance constraints introduced by the heterogeneity of cloud APIs and services. **Implications**: These findings suggest that security, resilience, and efficient migration should be key areas of focus for practitioners operating in multi-cloud environment. Researchers may also look into novel techniques for enhancing cloud security, improving system resilience, and easing the process of cloud migration.

4.3. Quality attributes and tactics (RQ3)

The exploration and implementation of quality attributes and associated tactics are critical to the successful deployment of containerized applications in multi-cloud environment, shaping performance, security, and compatibility characteristics. In the following, we will discuss the key findings of our study, providing insights into the QAs and tactics specifically tailored for containerized applications operating within multi-cloud environment.

Performance Optimization in Multi-Cloud Environment: Our study identifies the role of machine learning, container technology, and location-aware service brokering in enhancing the performance and efficiency of containerized applications within multi-cloud environment. Hashem et al. previously suggested similar efficiency gains from

machine learning and container technology, validating the interpretive value of our results (Hashem et al., 2016). Specifically, the application of machine learning aligns with a broader trend towards intelligent cloud resource management, a domain that Buyya et al. identified as significant in cloud computing research (Buyya et al., 2020). Meanwhile, location-aware service brokering emerges as a novel and impactful strategy, stressing the role of geographical considerations in further optimizing containerized applications' performance. Importantly, the reviewed studies emphasize that performance tactics must accommodate multi-cloud-specific challenges such as distributed latency, resource fragmentation across providers, and dynamic SLA enforcement, which are less relevant in single-cloud settings. **Implications**: For researchers, the results of our study offer opportunities to explore the integration of AI techniques with container-based cloud architectures further, especially the application of location-aware service brokering in multi-cloud environment. For practitioners, the results indicate that a adaptable approach involving AI, container technology, and geographical considerations can lead to significant performance enhancements in multi-cloud environment.

Security Measures in Multi-Cloud Deployments: The results of our study indicates the multi-faceted approach to ensuring security in multi-cloud deployments. It highlights the combination of encryption, network security protocols, machine learning, and user-specified security measures, while also encouraging deployment across different cloud providers. This finding complements Singh et al.'s argument for a comprehensive approach to cloud security, and aligns with the recent concept of distributed cloud security (Singh et al., 2016). Our results also report the importance of user-specified security measures, which are in agreement with the user-centric security model proposed by Shin (2013). Notably, multi-cloud containerization introduces additional layers of complexity in managing security policies across heterogeneous platforms and enforcing access control in federated settings. Tactics such as decentralized IAM logging and policy-aware container placement directly respond to these unique multi-cloud constraints. **Implications**: From a research perspective, the interact between different security measures and their effect on overall system security in multi-cloud environment warrants further investigation. For practitioners, these results point out the importance of employing a comprehensive security approach in multi-cloud deployments, including encryption, network security, machine learning, and user-specific security measures.

Ensuring Compatibility Across Different Cloud Providers: The results of our study suggests strategies to ensure compatibility in multi-cloud environment through standardization of interfaces, componentization, and interoperability approaches. Emphasizing lightweight communication protocols and interoperability between clouds, the results related to compatibility mirrors Mell and Grance's principles of service-oriented architecture, reinforcing the need for effective communication across different cloud platforms (Mell et al., 2011). Petcu et al. have also previously stressed the importance of standardization and componentization for ensuring compatibility (Petcu et al., 2018). Our findings highlight that achieving compatibility in a containerized multi-cloud context often involves abstracting away provider-specific interfaces through standardized APIs and decoupling infrastructure logic via containerization. These are adaptations not typically required in homogeneous cloud environments. **Implications**: From a research standpoint, the dynamic between lightweight communication protocols, interface standardization, and overall system compatibility in multi-cloud deployments offers fertile ground for future work. Practitioners, meanwhile, can adopt these approaches to ensure seamless communication and interoperability across different cloud providers.

These findings not only confirm existing literature results but also offer valuable insights into the practical application of machine learning, container technology, and location-aware services in multi-cloud environment. Future research could further explore the intersection of these quality attributes to develop more robust, efficient, and secure multi-cloud applications by using containerize technologies.

4.4. Automation challenges and solution framework (RQ4–RQ5)

The Automation Challenge-Solution Framework for Containerized Applications in multi-cloud environment (see Table 12) suggests that multi-cloud management for containerized application is a complex task encompassing several dimensions. These include multi-cloud automation, deployment and scaling, resource management, data and application migration, testing and bench marking, standardization and interoperability, application and service management, and runtime and service discovery. This indicates that addressing one challenge may potentially impact the other areas, necessitating a holistic approach to tackle these issues. In the following we discuss the some of the key observations about this framework.

Multi-dimensional Challenges: The Automation Challenge-Solution Framework for Containerized Applications in multi-cloud environment suggests that multi-cloud management is a complex task encompassing several dimensions. These include multi-cloud automation, deployment and scaling, resource management, data and application migration, testing and bench marking, standardization and interoperability, application and service management, and runtime and service discovery. This indicates that addressing one challenge may potentially impact the other areas, necessitating a holistic approach to tackle these issues. What makes these challenges unique to containerized multi-cloud setups is the dual complexity arising from container portability and cloud heterogeneity. Containers, while offering mobility and reproducibility, must now adapt to diverse runtime configurations, networking policies, and resource scheduling strategies across cloud providers. The multi-dimensionality of the challenges implies the need for an integrated approach that considers the interdependencies among these dimensions. It emphasizes the need for solutions that cater to this complexity rather than address individual challenges in isolation. **Implications**: For researchers, these findings underscore the need to develop multi-faceted, holistic solutions that can address the various interconnected challenges in managing multi-cloud environment. For practitioners, this implies the importance of strategic planning and implementing solutions that address the multiple facets of multi-cloud management while considering their potential impact on one another.

Emergence of AI/ML Solutions: The framework highlights the increasing utilization of AI and ML in providing intelligent solutions to manage containerized applications in multi-cloud environment. This suggests the growing maturity and applicability of these technologies in complex scenarios, a significant step towards achieving efficient multi-cloud management. Previous studies (e.g., de Laat et al. (2020), Chen et al. (2021) and Gill et al. (2022)) have recognized the potential of AI and ML in resolving complex cloud management issues, which is in line with the findings from the framework. This further validates the significant role AI and ML have started to play in this domain. The reviewed studies specifically show how AI/ML techniques help in real-time placement of containers across heterogeneous clouds, optimizing latency and cost under dynamic workload and policy constraints—issues that are intensified by the distributed and federated nature of multi-cloud environments. The increasing adoption of AI/ML in multi-cloud environment signifies an essential evolution in the field. This stresses the need for further research and development in these technologies to exploit their full potential in resolving multi-cloud management complexities. **Implications**: The growing use of AI/ML solutions implies that researchers need to focus on improving these technologies and adapting them to specific multi-cloud management challenges. For practitioners, this points towards the increasing necessity to invest in AI/ML capabilities and integrate them into their multi-cloud management strategies.

Importance of Standardization and Interoperability: The framework underscores the crucial role of standardization and interoperability in multi-cloud environment, facilitating seamless integration and management across different cloud platforms. Recent literature validates the importance of standardization and interoperability in multi-cloud environment, emphasizing their role in enhancing the efficiency

of cloud services (Alonso et al., 2023b). This agrees with our results, further confirming the critical need for these factors in the domain. The need for standardization and interoperability reiterates the necessity for a unified framework or protocol across different cloud environments. In containerized deployments, interoperability challenges become more pronounced due to variations in networking stacks, security policies, and orchestration tools between providers. Standardization at the container runtime level (e.g., Docker images, Kubernetes manifests) and orchestration abstraction (e.g., TOSCA, Helm charts) plays a vital role in enabling automation workflows that span diverse cloud ecosystems. It suggests that attention to these aspects can lead to more seamless, efficient, and secure management of multi-cloud services. **Implications**: For researchers, the results stress the need to investigate methods to achieve better standardization and interoperability in multi-cloud environment. For practitioners, the findings suggest the need to adopt standards and focus on interoperability while selecting and implementing cloud services, ensuring more efficient management across different platforms.

4.5. Security challenges and solutions framework (RQ4 - RQ5)

The framework proposed in our study provides a comprehensive and systematic approach to addressing the complex security issues associated with containerized applications in multi-cloud environment. In the following, we discuss the key takeaway from security challenges and solution framework presented in Section 3.5 and Table 11.

Emphasize on Container-Specific Security Mechanisms: Security solutions specifically designed for containerized applications are critical to managing threats in multi-cloud environment. The selected studies suggest a variety of methods, including secure container orchestration, container isolation, and advanced deployment strategies that make use of deep reinforcement learning. This aligns with the existing literature (e.g., Zhong et al. (2022), Li et al. (2023) and Combe et al. (2016)) which highlights the unique security considerations brought forth by containerization. Containers, being lighter than traditional virtual machines and offering process-level isolation, have revolutionized the way applications are packaged and run. However, they also introduce new security challenges that need to be addressed by using container-specific security mechanisms. Containerization, while a powerful tool for application deployment, necessitates its own suite of security strategies. Implementing these will be key to leveraging the benefits of containerization in a secure manner. Implementing these will be key to using the benefits of containerization in a secure manner. In multi-cloud settings, these challenges are compounded by the need to secure container interactions across diverse platforms, enforce consistent policies under different providers, and protect workloads that dynamically move between clouds. **Implications**: For researchers, this highlights an area of potential study: the development of advanced container-specific security strategies, including those that use advanced techniques such as machine learning. For practitioners, this suggests the need to be well-versed in the specifics of container security. Leveraging features of containerization platforms, such as Docker's isolation features, or incorporating tools specifically designed for container security will be crucial.

Implementing Multi-Layered Security and Compliance Measures: Implementing multi-layered security measures that encompass data protection, access control, and communication security, along with compliance with legal and regulatory requirements, is of utmost importance. Solutions range from data encryption, role-based access control, secure container orchestration, adherence to regulations like GDPR (Regulation, 2016). This is consistent with current cybersecurity frameworks, such as the NIST cybersecurity framework (National Institute of Standards and Technology, 2018), which emphasize a multi-layered approach to security. Furthermore, several studies point to the criticality of data security, user access control, and secure communication in cloud environments (Austin, 2018; Singh et al., 2016). In a multi-cloud, containerized environment, it is not enough to secure data; access and communication channels must also be secured, and all actions must comply with legal and regulatory requirements. In multi-cloud containerized setups, multi-layered security must also accommodate interoperability between CSPs, dynamic access management across domains, and distributed compliance enforcement for container workflows. **Implications**: For practitioners is to design and implement a comprehensive security strategy that encompasses multiple layers of protection. Compliance with legal and industry standards should be an integral part of this strategy. For researchers, this underlines the need for studies that address the integration of various layers of security and the development of comprehensive security frameworks.

Ensuring Security through Standardization and Interoperability: Ensuring security through standardization and interoperability is highlighted as a critical step. Secure and standardized interfaces, such as OpenStack APIs, can be used to enhance security and performance across multiple cloud providers. This finding is aligned with existing literature, which emphasizes the role of standardization in achieving security and interoperability in multi-cloud environment (Buyya et al., 2010; Petcu, 2011). By implementing standardized interfaces and practices, organizations can ensure a level of interoperability and security across cloud environments. For containerized applications, this also means ensuring consistent security postures across orchestrators, managing identity federation between providers, and enabling trust in federated environments that dynamically provision containers at runtime. **Implications**: For practitioners, this means choosing platforms and tools that support standardized interfaces and practices, or working to implement those standards within their own environments. Researchers, on the other hand, could focus on the development of new standards or the improvement of existing ones to better address security challenges in multi-cloud, containerized environments.

4.6. Deployment challenges and solutions framework (RQ4 - RQ5)

Table 14 offers a comprehensive overview of challenges and their corresponding solutions encountered during the deployment of containerized applications in multi-cloud environment. In the following, we provide a discussion on key findings pertaining to the challenges and solutions related to the deployment of containerized applications in multi-cloud environment.

Standardization and Automation in Deployment: Standardization and automation stand out as foundational pillars for achieving streamlined and uniform deployments in multi-cloud landscapes. As Raj et al. (2018) highlighted, embracing automated deployment pipelines paired with standardized toolsets can significantly diminish complexity and mitigate human-induced errors within multi-cloud frameworks. The challenges, captured under *Performance Testing Deployment*, *Deployment Validation*, and *Cloud Management Platform Evaluation Deployment* in the table, elucidate the nuanced complexities associated with deployments spanning multiple cloud platforms. Solutions such as *Standardized Benchmarking and Automation*, *Testing Process Management System*, and *Standardized Output Formats and Evaluation Criteria* emphasize the imperative nature of adopting uniform procedures and harnessing automated solutions. Multi-cloud deployments, inherently intricate, benefit profoundly from standardization, bringing forth predictability and coherence, while automation paves the way for diminishing human-driven inconsistencies and bolstering deployment speeds. Multi-cloud deployments, inherently intricate, benefit profoundly from standardization, bringing forth predictability and coherence, while automation paves the way for diminishing human-driven inconsistencies and bolstering deployment speeds. In containerized environments, these challenges are amplified due to the need to consistently deploy lightweight, encapsulated workloads across heterogeneous infrastructure with varying APIs and orchestration logic. **Implications**: For

researchers, the quest for optimized frameworks championing standardization and automation within multi-cloud environment presents a promising research trajectory. For practitioners, adeptness with standardized methodologies and tools, combined with proficiency in automation implementation, becomes crucial in navigating the multi-cloud management maze efficiently

Security and Compliance in multi-cloud environment: Security remains paramount in cloud deployments. Fernandes et al. underscore that the labyrinth of diverse platforms and differing standards in multi-cloud configurations amplifies security complexities (Fernandes et al., 2014). Challenges articulated as *Security Deployment in Multi-container Environment* and *Malicious Service Management Deployment* in Table 13 accentuate the primacy of security considerations. Propounded solutions like *Cross-Container Isolation* and *Collusion-Resilient Trust Aggregation Technique* spotlight the industry's shift towards advanced and meticulous security protocols within multi-cloud paradigms. The ever-evolving landscape of multi-cloud deployments demands security solutions that are not only rigorous but also malleable and formidable. To attain comprehensive security across diverse platforms, strategies must encapsulate both depth of understanding and breadth of application. Containerization introduces new layers of isolation and deployment granularity, which—when combined with multi-cloud distribution—creates a unique surface of vulnerability that necessitates tailored controls at both the container and orchestration levels. **Implications**: For the academic community, pioneering avant-garde security solutions apt for multi-cloud environment offers a fertile domain of inquiry. Meanwhile, practitioners should enshrine principles of data integrity, trust assurance, and vulnerability mitigation at the heart of their multi-cloud deployment blackprints.

Integration, Monitoring, and Infrastructure Management: As highlighted by Ferrer et al. (2016) the ability to operate seamlessly across disparate cloud providers and maintain flexible infrastructures is paramount in harnessing the full potential of multi-cloud deployments. The presented challenges, including *Network Latency Deployment* and *Vendor Lock-in Prevention Deployment*, underscore the indispensability of agility and cross-platform compatibility. Proposed solutions such as *VM Selection for Composite Applications* and *Interoperability Layer Above Cloud Infrastructure* accentuate the importance of crafting adaptable infrastructure and formulating strategies to mitigate vendor entrenchment. Here, containerization plays a distinct role by enabling application components to be packaged once and executed across diverse cloud platforms, yet this same portability necessitates careful orchestration, consistent networking, and compatibility layers across varied cloud APIs and runtime environments. In the context of multi-cloud environment, flexibility and interoperability transcend mere advantageous attributes; they become pivotal determinants shaping adaptability, scalability, and the sustained success of deployment blackprints. **Implications**: For the academic community, delving into novel approaches that champion infrastructure pliability and seamless operation across diverse cloud platforms emerges as a prospective research avenue. For industry professionals, anchoring strategies around flexibility and interoperability not only optimizes current deployments but also fortifies them against future technological evolutions and shifting business dynamics.

4.7. Monitoring challenges and solutions (RQ4 - RQ5)

Table 14 presents a comprehensive view of the Monitoring Challenge-Solution Framework for Containerized Applications in multi-cloud environment. It covers a wide range of monitoring challenges within the intricate multi-cloud setting. In the following, we are providing discussion on the key findings.

Enhancing Performance Stability through Multi-Cloud Monitoring: In the context of multi-cloud environment, ensuring consistent and reliable performance monitoring for containerized applications can be challenging due to the dynamic nature of cloud resources. Existing studies (e.g., Zhong et al. (2022) and Mungoli (2023)) highlighted the impact of varying resource availability on the performance of containerized applications across multiple clouds. They emphasized the need for performance benchmarking to track fluctuations and maintain a consistent user experience. The solutions presented in our framework, such as benchmarking performance variation and utilizing Kubernetes for monitoring, align with existing studies suggestion for benchmarking. Unlike traditional single-cloud deployments, containerized applications spanning multiple clouds must be monitored for platform-specific latency, orchestration timing mismatches, and fragmented telemetry data pipelines. Our proposed framework expands on this by incorporating Kubernetes for improved multi-cloud performance monitoring. Kubernetes, while originally designed for single-cluster orchestration, plays a central role in aggregating monitoring data across containerized services deployed in heterogeneous clouds, helping unify performance visibility. Our proposed framework expands on this by incorporating Kubernetes for improved multi-cloud performance monitoring. Our framework also suggest reliable performance monitoring requires a combination of standardized benchmarking techniques and cloud-specific tools like Kubernetes. This approach can effectively address performance variations and ensure a consistent user experience across diverse cloud environments. **Implications**: For practitioners, adopt benchmarking practices to track performance fluctuations, and use Kubernetes for unified multi-cloud performance monitoring. For researchers, explore further refinements in benchmarking methodologies and investigate the impact of dynamic multi-cloud environment on performance consistency.

Streamlining Resource and Network Management across Multi-Clouds: Managing network communications, system images, and resource utilization in a multi-cloud setting poses challenges due to the diversity of cloud platforms. Sedghpour et al. discussed the complexities of managing network communications in multi-cloud environment. They emphasized the need for standardized approaches like service mesh to enhance network visibility and control (Sedghpour and Townend, 2022). Solutions in our proposed framework, such as using Docker Server for containerized services and Prometheus/Grafana for resource monitoring, align with Sedghpour and Townend's (2022) suggestion for standardized approaches. However, container-based deployments compound the difficulty by adding layers of ephemeral, distributed services that must be monitored across volatile container lifecycles. This increases the burden of synchronization and observability. Additionally, our framework extends these solutions to address broader resource management challenges. Containers, especially when orchestrated over multiple cloud platforms, generate high-churn environments that require lightweight, highly adaptive monitoring strategies to ensure timely insights and cost control. Additionally, our framework extends these solutions to address broader resource management challenges. Employing standardized tools like Docker Server, along with platforms like Prometheus and Grafana, can help streamline resource management and network communication monitoring across diverse cloud platforms. **Implications**: For practitioners, implement Docker Server and utilize platforms like Prometheus and Grafana for efficient resource monitoring and management across multi-cloud environment. For researchers, investigate further integration possibilities for standardized resource management tools and explore the impact of such tools on overall system performance and reliability.

Strengthening Security, Compliance, and Trust: Ensuring data privacy, continuous security control, and compliance in multi-cloud containerized applications is essential but challenging due to the distributed nature of cloud resources. Existing studies (e.g., Alonso et al. (2023b) and Raj et al. (2018)) discussed the challenges of ensuring security and compliance across multi-cloud environment. They highlighted the importance of continuous monitoring and enforcement mechanisms to address security and compliance gaps. Solutions presented in our proposed framework, such as using machine learning for performance supervision and employing security assurance platforms,

are aligned with existing studies that emphasis on continuous monitoring and enforcement for security and compliance. Containerized deployments introduce distinct security risks, e.g., image poisoning, privilege escalation between co-resident containers, and cross-platform trust boundary misconfigurations that demand container-aware monitoring solutions. Implementing advanced techniques like machine learning for monitoring and adopting security assurance platforms can significantly enhance security, compliance, and trust management in multi-cloud containerized applications. **Implications**: For practitioners, integrate machine learning-based monitoring and adopt security assurance platforms to ensure continuous security control and compliance across diverse cloud environments. For researchers, explore the scalability and effectiveness of machine learning-driven monitoring techniques and investigate novel methods for enforcing security and compliance in multi-cloud setups.

4.8. Tools and frameworks (RQ6)

This study report 160 distinct tools and frameworks for container based applications in multi-cloud environment, all of which are systematically classified into 6 categories, further delineated into 46 subcategories. In the following we briefly discuss some of the key findings about identified tools and frameworks.

Diverse Cloud Services by Leading Providers: Our study reveals a diverse array of cloud services, including IaaS, PaaS, and SaaS offerings from major providers like AWS, Azure, and Google Cloud, emphasizing the extensive options available to developers (see Table 15). This underscores the significant range of choices available to developers. Noteworthy cloud service providers highlighted in our study include Amazon EC2, Google Cloud App Engine, and Azure, further underscoring this finding. In multi-cloud environments, selecting services becomes more complex due to differing billing models, feature parity, and integration capabilities across providers—requiring tools that support cross-provider abstraction and orchestration. **Implications**: For practitioners, our outcomes underscore the critical necessity of meticulously evaluating and selecting the most fitting service model based on the unique demands of a given project. This strategic approach ensures the best alignment between technology and goals. Moreover, the implications for researchers are promising: an avenue emerges for in-depth exploration of trade-offs and the identification of optimal practices while navigating the intricate array of cloud service models. This exploration holds potential for enhancing both the efficiency and effectiveness of cloud solutions.

Docker and Kubernetes: Pioneers of Cross-Cloud Containerization: Our investigation spotlights the profound significance of Docker and Kubernetes in streamlining the containerization and orchestration of applications across a spectrum of cloud platforms. Docker adeptly encapsulates applications and their dependencies within self-contained containers, while Kubernetes takes center stage as the orchestration juggernaut, automating deployment, scaling, and management of these containers. These technologies receive direct validation through table references, serving as prime exemplars. However, our analysis also reveals that deploying Kubernetes in a multi-cloud environment demands adaptations, such as federation control planes, multi-zone DNS, and workload placement strategies tailored to cross-cloud latencies and SLAs. Our research yields a compelling revelation: Docker and Kubernetes hold pivotal roles in ensuring consistent application deployment and seamless scalability, effectively simplifying the complexities of multi-cloud environment. **Implications**: Practitioners stand to gain significantly by adopting Docker and Kubernetes, as this potent combination bolsters efficiency and cultivates heightened portability. Researchers can explore emerging gaps in areas such as secure cross-cloud scheduling, fault tolerance across zones, and policy-compliant orchestration.

The Rise of DevOps: Embracing Automation Tools: Our exploration spotlights the important role of DevOps practices and tools such as Puppet, Chef, and Terraform within contemporary software development landscapes. This emphasis revolves around the power of automation in facilitating streamlined deployment and management processes. These tools, by enabling consistent provisioning and deployment of infrastructure, effectively curtail errors and ensure the replicability of processes. The table substantiates this notion through references to Puppet, Chef, and Terraform. Within a multi-cloud context, these tools must be adapted to handle provider-specific APIs, authentication schemes, and compliance constraints, which introduces configuration drift and policy management complexity.. Implications: Practitioners are advised to seamlessly integrate DevOps principles into their workflows by harnessing the prowess of these tools. Researchers may investigate how DevOps pipelines can be extended to support containerized workloads that span multiple clouds, ensuring traceability, consistency, and auditability across heterogeneous execution environments.

5. Threats to validity

This systematic mapping study is susceptible to various threats that could influence its outcomes. To address these potential threats, we adhered to the established guidelines for conducting SMSs and SLRs as outlined in Keele et al. (2007) and Petersen et al. (2008). We further categorized and analyzed the validity threats to our study based on the four distinct types of validity threats mentioned in Zhou et al. (2016) and Wohlin et al. (2012). In the subsequent sections, we delve into the specific validity threats that pertain to the different phases of this SMS.

5.1. Internal validity

Internal validity refers to the factors that could affect the analysis of the data extracted from the selected studies. The threats to internal validity could happen in the following steps of this SMS:

- **Study Search**: The potential to overlook studies during the search process requires careful measures. To mitigate this risk, we used a combination of primary and snowballing search methodologies, as detailed in Section 2.3.1. To increase the collection of primary studies during the initial search phase, we also took additional measures to address search-related issues. Specifically, we improved our search strings through pilot searches before applying them to the databases. This process helped to create a more effective search strategy.
- **Study Selection**: We have outlined the study screening and selection process in Table 3. To ensure objectivity and eliminate personal bias in study selection, we implemented a two-phase approach: (i) initial study screening, and (ii) qualitative evaluation of the shortlisted studies. During this procedure, the lead two authors conducted the study screening based on the criteria explicitly detailed in Table 3. To assess the objectivity of the screening process and the level of agreement between the two authors, we applied Cohen's Kappa (Cohen, 1968). In cases where the lead authors did not agree, the second and third authors independently reviewed the disputed studies to reach a consensus. All the researchers involved in this study have extensive knowledge and research experience in containerization applications within a multi-cloud environment.
- **Data Extraction**: Bias from researchers during data extraction can pose a significant challenge in both SMSs and SLRs. To address this potential issue, we developed a standardized data extraction form (see Table 5) to ensure consistent data retrieval. Initially, the first and fourth authors extracted the data. If any uncertainties arose regarding the extracted data, comprehensive discussions were convened among all authors. Following the recommendations in Wohlin et al. (2012), a subset of the extracted data was cross-verified by the second and third authors.

- **Bias on Themes Classification**: Misclassification of data and primary studies may result in biases stemming from subjective interpretation. To minimize this risk, we adhered to the thematic analysis guidelines established by Braun and Clarke (2006), and implemented a six-step process for thematic classification, as detailed in Section 2.4.
- **Data Synthesis**: We employed both qualitative and quantitative approaches to assess the gathered data. Potential biases in data synthesis could influence the interpretation of our findings. To address this concern, we used open coding and constant comparison techniques from Grounded Theory (Glaser and Strauss, 1967) to analyze the qualitative data extracted from the selected studies.
- **Transparency of Quality Assessment Process:** While our replication package (Waseem et al., 2025) provides a detailed breakdown of quality assessment scores for each study and each criterion, it does not include per-study justifications or annotations explaining how individual scores were assigned. This may present a limitation in terms of external reproducibility, as other researchers may not fully reconstruct the decision-making rationale behind each score. Although we followed a consistent and internally validated scoring logic (e.g., peer-reviewed venues scored higher, empirical data presence was required for full marks), we recognize that the absence of a fully documented scoring rubric or narrative rationale could reduce transparency. However, to partially mitigate this, we have explicitly described our scoring criteria and interpretation rules in Section 2.3.3. This clarification improves traceability by explaining the rationale used across all assessed studies. The limitation remains inherent to balancing the breadth and depth of quality assessment in a large-scale review and is acknowledged as a tradeoff in terms of replication of this study.

5.2. External validity

Concerns about external validity relate to the applicability and generalizability of study findings. Our research provides a thorough examination of containerization in multi-cloud environment, with findings, analyses, and conclusions specifically tailored to this domain. To ensure robust external validity, we developed a study protocol that outlines our research methodology. The literature review spanned a decade, from January 2013 to March 2023, and included peer-reviewed articles from eight preeminent databases in the fields of software engineering and computer science, which are listed in Table 2. While the review is anchored in academic research, which may not encapsulate unpublished industry practices, the systematic approach and comprehensive timeframe of our analysis make the insights valuable for both academia and practitioners. To complement and enrich our findings, future work could include an industrial study (e.g., blogs, white papers) that would provide a broader view of the application of containerization in practice

5.3. Construct validity

Construct validity concerns to the accuracy of the operational measures used for data collection in a study. The primary constructs of this study revolve around two concepts: "containerization" and "multi-cloud environment". Utilizing imprecise or incomplete search terms, or employing unsuitable search strategies, can lead to potential pitfalls such as overlooking pertinent papers or including numerous irrelevant ones during the search phase, and omitting relevant papers during the selection phase. To counter these risks, we implemented the following measures: (i) we initiated a pilot search to verify the relevance and comprehensiveness of our search terms; and (ii) we searched paper on eight databases renowned for computer science and software engineering research. Additionally, we customized search string according to each database syntax.

5.4. Conclusion validity

Threats to conclusion validity relate to factors, such as data inaccuracies, that can impede drawing accurate conclusions. To mitigate these threats, we adhered to best practices, including the search protocol, pilot search, and pilot data selection, as recommended by Keele et al. (2007) and Petersen et al. (2008). Moreover, to further ensure the validity of our conclusions, the authors engaged in multiple brainstorming sessions to collaboratively interpret the results and finalize the conclusions.

6. Related work

This section reviews the most relevant existing research in terms of secondary studies such as literature surveys, state-of-the-art analysis, systematic reviews, and mapping studies that consolidate published literature on containerization in multi-cloud environment. The review focuses on (i) existing challenges, proposed solutions, and emerging trends detailed in Section 6.1 and (ii) container-based deployment of multi-cloud systems in Section 6.2.

6.1. Challenges, solutions, and performance of multi-cloud containers

In recent years, several literature surveys have been published to analyze state-of-the-art on the applications (Pahl et al., 2017), tools and technologies (Casalicchio and Iannucci, 2020), potential and limitations (Bentaleb et al., 2022), as well as emerging trends (Watada et al., 2019) of container-based solutions for cloud computing systems, discussed below.

6.1.1. Challenges, applications, tools, and emerging trends

Containers in multi-cloud environment provide a standardized and portable way to package, deploy, and run applications, ensuring seamless deployment and management across diverse cloud platforms (Alonso et al., 2023b; Saxena et al., 2021). One of the earliest SLR-based studies by Pahl et al. (2017) on containerized clouds reviewed a total of 46 studies to identify, taxonomically classify and systematically compare the existing research on containers and their application in the context of cloud-based systems. The SLR classified and compared the selected research studies using a conceptual framework to highlight existing trends and needs for future research. SLR results indicate container orchestration, microservice delivery, continuous development and deployment as emerging trends of research on containerized clouds. The SLR in Alonso et al. (2023b) reviewed a total of 88 studies (published from 2011–2021) on multi-cloud systems. Similar survey studies such as Saxena et al. (2021) concepts, challenges, requirements and future directions for multi-cloud environment are discussed. A survey of existing approaches and solutions provided by different multi-cloud architectures is entailed along with analysis of the pros and cons of different architectures while comparing the same (Imran et al., 2020).

6.1.2. Performance and resource utilization

In a study by Bentaleb et al. (2022), the authors follow the taxonomical classification from Pahl et al. (2017) to categorize container-based technologies for cloud systems. The study argues for the need of performance metrics to objectively define and evaluate quality attributes such as resource virtualization, service elasticity, orchestration, and multi-tenancy in containerized cloud systems. The study also highlights the needs for future research in terms of best practices, i.e., patterns and tactics (enabling reuse) and tools (supporting automation) for container-based cloud deployment. In the context of performance, the studies (Casalicchio and Iannucci, 2020; Watada et al., 2019) report literature review and experimental analysis of factors that influence performance of container-based cloud systems. Specifically, Casalicchio

and Iannucci (2020) reviewed a total of 97 research studies investigating performance evaluation and run-time adaptation of container-based solution. The study highlighted several unsolved challenges such as I/O throughput optimization, performance prediction, and multi-layer monitoring performance bottlenecks. Moreover, Watada et al. (2019) conducted an experimental study to compare the performance of VMs, containers and uni kernels in terms of and technological maturity using standard benchmarks and observed containers to optimize performance of container-based. The performance of containerized clouds is also determined by resource utilization, which includes but is not limited to CPU utilization, memory footprints, energy consumption, network bandwidth, and execution time.

Netaji and Bhole (2021) reviewed 64 research papers to gain insight into resource allocation, management, and scheduling. Furthermore, limitations of existing resource allocation algorithms are discussed, indicating the need to investigate algorithms or techniques of performance optimization of containers for cloud systems. The study by Maenhaut et al. (2020) provided an overview of the current state of the art regarding resource management within the broad sense of cloud computing, complementary to existing surveys in the literature. The study investigated how research is adapting to the recent evolution within cloud solutions, including container technologies.

6.2. Container-based cloud orchestration, deployment, and security

This section reviews the most relevant research on the container-based deployment with a focus on orchestration and security issues of cloud-based systems, detailed below.

6.2.1. Orchestration and deployment

Zhou et al. (2022) explored containerization in High Performance Computing (HPC) environments, contrasting it with cloud computing. The results of this study indicate that containers enhance application deployment efficiency, but face challenges in HPC due to high security levels and the need for extensive libraries and packages that affect cloud portability, often resulting in vendor lock-ins. The study also provides a survey and taxonomy on containerization and orchestration efforts in HPC, pointing out differences with cloud environments and identifying future research and engineering potentials. A systematic mapping study by Bachiega et al. (2018) explores virtualization in container-based cloud deployments. The mapping study, based on a comprehensive review of major databases, identifies a significant research gap in the performance evaluation of containers, underscoring the need for further investigation on cloud deployment. Awada (2018) present an optimization strategy for deploying microservices in multi-cloud environment focusing on minimizing cloud service costs, network latency, and startup time for new microservices. Their approach uses a Non-dominated Sorting Genetic Algorithm II (NSGA-II) compared to a Greedy First-Fit algorithm, demonstrating a 300% improvement with NSGA-II. This highlights its effectiveness in container and VM orchestration, offering a significant enhancement in deployment of multi-cloud solution in containers. Paladi et al. (2018) survey container orchestration, proposing a reference architecture for autonomic orchestrators, and identifying research challenges in the field. Their work emphasizes the importance of container technologies in cloud environments and the need for advanced orchestration solutions to manage complex multi-container applications effectively. Guerrero et al. (2018) reviews container orchestration tools and platforms, comparing the architectures, components, and capabilities of several Container Service Platforms (CSPs) and orchestration tools like Amazon ECS, Kubernetes, Docker Swarm, and Mesos. The study offers insights into the current state of container orchestration and suggests future research directions, serving as a guide for developers and organizations.

Dolui and Kiraly (2018) explore multi-container deployment on IoT gateways to meet the stringent latency requirements of advanced IoT applications. Through their study within the AGILE project, they highlight containerization's role in overcoming the diversity and resource constraints of IoT gateways, showcasing containerization's advantages for IoT gateway performance optimization. Their research underscores the potential of containerized environments in enhancing application compatibility, portability, and efficient deployment across diverse hardware architectures. Nardelli et al. (2018) introduce the Adaptive Container Deployment (ACD) model to optimize containerized application deployment in geo-distributed environments, focusing on IoT and fog computing resources. ACD, formulated as an Integer Linear Programming problem, aims to improve application performance by leveraging containers' horizontal and vertical elasticity. The study evaluates ACD's effectiveness against greedy heuristics, highlighting the need for advanced orchestration solutions to exploit emerging computing environments' characteristics efficiently.

6.2.2. Security of containerized clouds

Casalicchio (2019) analyze security challenges in cloud orchestration for multi-cloud deployments, proposing a security-enabled orchestration framework. Their research identifies potential attack scenarios and security enforcement mechanisms, aiming to enhance security guarantees for cloud operators in a multi-cloud setting. Bokhari et al. (2018) offer a detailed analysis of cloud computing, defining its essential characteristics, architecture, service models (SaaS, PaaS, IaaS), and deployment models. They discuss the security requirements for public and private clouds across different service models, aiming to provide researchers with a comprehensive understanding of cloud computing's potential and security challenges. Sultan et al. (2019) survey the landscape of container security, addressing challenges and solutions across four generalized use cases within the host-container threat landscape. Their analysis covers both software-based solutions utilizing Linux kernel features and hardware-based solutions for enhanced security. The study aims to clarify container security requirements and encourage further research in addressing potential vulnerabilities and attacks. Graupner et al. (2015) tackle privacy, security, and trust issues in cloud computing through a multi-cloud architecture perspective. They propose a novel technique for enhancing security and unifying access control mechanisms across cloud providers, addressing the challenges of inadequate cross-provider APIs and non-unified access control. Gupta et al. (2016) propose a security-enhancing methodology for cloud data storage using container clustering and Docker instances for on-demand encryption. This approach aims to secure data while optimizing resource usage and cost, highlighting the benefits of container technology in improving cloud computing security and efficiency.

6.3. Conclusive summary

Current survey-based studies and systematic reviews, discussed above (e.g., Casalicchio and Iannucci (2020) and Pahl et al. (2017)), aim to consolidate the latest findings in published research concerning container-based solutions for cloud computing systems. These studies investigated various aspects of multi-cloud, including exploring applications, tools, and technologies, assessing potential and limitations, and identifying emerging trends. Table 16 presents an overview of the differences between this study and existing secondary studies. The symbols used in the table are as follows: ✗ indicates "Yes", ✓ indicates "No", and △ indicates "Partially". This study distinguishes itself from prior secondary studies in multiple critical ways, thereby advancing the state of the art in the domain of containerization for multi-cloud environments. While existing surveys and mapping studies (e.g., Pahl et al. (2017), Casalicchio and Iannucci (2020) and Netaji and Bhole (2021)) provide valuable overviews of tools, technologies, performance, and resource management, they often focus on isolated aspects or lack integration across key quality and deployment dimensions. In contrast, our SMS presents a comprehensive, theme-based classification of 121 selected studies—making it the most extensive secondary analysis in this domain to date. First, our work systematically

Table 16
Comparison of this SLR results with existing secondary studies. MS indicates "Mix Study", LR indicates "Literature Review", ✓ indicates "Yes", ✗ indicates "No", and △ indicates "Partially".

Contributions	This study	Pahl et al. (2017)	Casalicchio and Iannucci (2020)	Netaji and Bhole (2021)	Bentaleb et al. (2022)	Watada et al. (2019)	Saxena et al. (2021)	Imran et al. (2020)	Casalicchio (2019)
Number of selected studies	121	46	97	64	✗	40	15	11	✗
Study types	SMS	SMS	LR	LR	LR	MS	LR	LR	MS
Container roles (Section 3.2.1)	✓	△	✗	△	△	△	✗	✗	△
Container implementation strategies (Section 3.2.2)	✓	✗	△	✗	△	△	✗	✗	✗
Pattern and strategies (Section 3.3)	✓	✗	✗	✗	✗	✗	✗	△	△
Quality attributes and tactics (Section 3.4)	✓	△	✗	✗	△	✗	✗	✗	✗
Security challenges and solution (Section 3.5)	✓	△	△	✗	△	✗	✗	✗	✗
Automation challenges and solution (Section 3.6)	✓	△	△	✗	△	✗	✗	✗	✗
Deployment challenges and solution (Section 3.7)	✓	△	✗	✗	✗	✗	✗	✗	✗
Monitoring challenges and solution (Section 3.8)	✓	✗	△	✗	✗	✗	✗	✗	✗
Tools and frameworks (Section 3.9)	✓	△	△	✗	△	✗	✗	✗	✗

categorizes container utilization in multi-cloud environment into coherent categories and themes. Some of the key themes are Scalability and High Availability, Performance and Optimization, Security and Privacy, and Multi-Cloud Container Monitoring and Adaptation. We identified and classified seventy-four patterns and strategies for containerization, which existing studies have not provided. This level of operational detail is absent in prior work. Furthermore, we explored the QAs and associated tactics for containerization in multi-cloud environment. Unlike existing reviews that superficially mention quality attributes, we provide a dedicated framework linking QAs with implementation tactics, offering practical insights for engineering design decisions—an aspect previously unexplored. Our SMS also identifies and formalizes four distinct challenge-solution frameworks in the areas of security, automation, deployment, and monitoring. These frameworks synthesize fragmented research findings into cohesive, actionable knowledge structures, not previously consolidated in the literature. Lastly, our comparative analysis (Table 16) demonstrates that none of the previous studies collectively cover the breadth and depth addressed by our SMS, particularly in integrating performance, security, orchestration, and QA strategies across the entire lifecycle of container-based multi-cloud systems. By addressing these multiple dimensions holistically, our study not only fills significant gaps in the literature but also provides a practical roadmap for researchers and practitioners aiming to optimize container deployments in complex, heterogeneous cloud environments.

7. Conclusions

This SMS presents the current state of research regarding containerization in multi-cloud environment, focusing on aims of selected studies, the roles of containers, containers implementation strategies, architectural patterns and strategies, quality attributes and corresponding tactics. Additionally, this SMS also explores deployment, monitoring, security challenges, accompanied by their respective solutions, tools, and frameworks used to implement containerized applications in multi-cloud environment. We investigated 121 relevant studies to answer the RQs in Table 1 with key findings of this SMS include:

- The findings of this SMS reveal various insights regarding SMS demographics, publishers, publication types, authors' affiliations, and research themes. The yearly distribution reached its peak in 2016, signifying a shifting level of interest over time. The field of publishing is largely dominated by IEEE (55.37%), with conferences emerging as the primary avenue for publication. Notably, academia (75.02%) stands out as the leading affiliation among authors. The prominent research themes encompass *Orchestration and Management*, *Scalability and High Availability*, *Performance and Optimization*, and *Security and Privacy*. These themes underscore crucial aspects such as container management, scalability, performance, and security within the context of multi-cloud environment.
- This SMS identifies and classifies a total of 98 patterns and strategies across 10 subcategories and 4 categories for container-based applications in multi-cloud environment. These patterns and strategies encompass a wide range of aspects such as cloud architecture models, communication and networking, deployment, service models, cloud management, container management, edge computing, IoT strategies, security, resilience, fault-tolerance, and cloud migration. These findings are significant for the application and advancement of containerization-based applications in multi-cloud environment, providing valuable insights into optimizing architecture, communication, management, security, and migration strategies.
- This SMS identifies and classifies QAs and tactics from selected studies regarding containerized applications in multi-cloud environment. A total of ten QAs and their related terms are identified. Along with these QAs, a total of 47 tactics are identified to enhance the QAs. Notable takeaways include the utilization of machine learning and location-aware service brokering to enhance the performance and efficiency of container-based applications in multi-cloud environment.
- This SMS presents a comprehensive security challenge-solution framework for containerized applications in multi-cloud environment. This framework offers valuable insights and actionable strategies for practitioners and researchers to enhance the security of containerized applications, address complex multi-cloud security concerns, and ensure compliance with various regulatory requirements, thereby enabling the development of robust and secure containerized applications for multi-clouds.
- This SMS presents a catalogs for the automation challenge-solution catalogs for containerized applications in multi-cloud environment. The automation challenge-solution catalog offers a comprehensive guide for practitioners and researchers to address the complex challenges associated with automating various aspects of deploying, managing, scaling, testing, migrating, and ensuring interoperability of containerized applications in multi-cloud environments.

- The third catalog presented in this SMS is the deployment challenge-solution catalog for containerized applications in multi-cloud environment. This catalog provides valuable insights and practical guidance to practitioners and researchers alike, aiding practitioners in navigating the complexities of multi-cloud deployment through specific solutions to challenges, while also serving as a comprehensive resource for researchers to delve into various aspects of containerized application deployment, orchestration, security, scalability, and more, thus contributing to the advancement of knowledge and innovation in the field of multi-cloud deployment.
- Fourth catalog presented in this SMS is monitoring challenge-solution catalog for containerized applications in multi-cloud environment. This catalog serves as a comprehensive guide for practitioners, offering specific solutions to challenges related to monitoring performance, resource management, security, optimization, system complexity, and multi-cloud coordination, facilitating their understanding of complex monitoring scenarios and providing actionable insights. Furthermore, researchers can benefit from this catalog by gaining a deeper insight into the nuances of monitoring containerized applications in multi-cloud environment, and it can serve as a foundation for further research and innovation in the field of multi-cloud monitoring.

The findings of this SMS will benefit researchers who are interested in understanding the state of research on containerized applications in multi-cloud environment and conducting further investigations to address the open research issues highlighted in Section 4. Additionally, the insights gained from this SMS will support knowledge transfer to practitioners by providing insights into the challenges, solutions, and methods for monitoring, securing, and optimizing containerized applications in multi-cloud environment. We emphasize the importance for practitioners to develop targeted solutions that effectively tackle monitoring, security, and performance degradation concerns within multi-cloud environment deployment. As a future endeavor, we intend to enhance our SMS by conducting industrial case studies with companies, thereby obtaining practical insights and perspectives from practitioners on the effectiveness and applicability of our proposed catalogs. This approach will allow us to bridge the gap between research and practical implementations concerning containerized applications in multi-cloud environment and contribute to the advancement of both academic and industry understanding in this domain.

CRediT authorship contribution statement

Muhammad Waseem: Writing – review & editing, Writing – original draft, Visualization, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Aakash Ahmad:** Writing – review & editing, Visualization, Methodology, Data curation. **Peng Liang:** Writing – review & editing, Writing – original draft, Supervision, Funding acquisition. **Muhammad Azeem Akbar:** Writing – review & editing, Visualization, Investigation. **Arif Ali Khan:** Writing – review & editing. **Iftikhar Ahmad:** Writing – review & editing. **Manu Setälä:** Writing – review & editing. **Tommi Mikkonen:** Writing – review & editing, Writing – original draft, Validation, Supervision, Investigation, Funding acquisition, Conceptualization.

Declaration of AI Assistance

During the preparation of this work, the author(s) used ChatGPT to refine grammar, improve sentence structure, and resolve formatting issues. After utilizing this tool, the author(s) thoroughly reviewed and edited the content as needed, taking full responsibility for the final publication.

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.

Acknowledgments

This research is funded by Business Finland through QLEAP (2022–24) project, the National Natural Science Foundation of China (NSFC) under Grant No. 62172311, and the Major Science and Technology Project of Hubei Province under Grant No. 2024BAA008.

Data availability

The dataset used in this study is openly available on Zenodo : https://zenodo.org/records/14692878.

References

Acquaviva, Luca, Bellavista, Paolo, Bosi, Filippo, Corradi, Antonio, Foschini, Luca, Monti, Stefano, Sabbioni, Andrea, 2017. NoMISHAP: A novel middleware support for high availability in multicloud paas. IEEE Cloud Comput. 4 (4), 60–72.

Ahmad, Haris, Aujla, Gagangeet Singh, 2023. GDPR compliance verification through a user-centric blockchain approach in multi-cloud environment. Comput. Electr. Eng. 109, 108747.

Ahmad, Iftikhar, Autto, Teemu, Das, Teerath, H"am"al"ainen, Joonas, Jalonen, Pasi, J"arvinen, Viljami, Kallio, Harri, Kankainen, Tomi, Kolehmainen, Taija, Kontio, Pertti, Kotilainen, Pyry, Kurittu, Matti, Mikkonen, Tommi, Mohanani, Rahul, M"akitalo, Niko, Partanen, Jari, Pajasmaa, Roope, Pellikka, Jarkko, Set"al"a, Manu, Siukonen, Jari, Sorvisto, Anssi, Sroor, Maha, Suominen, Teppo, Timonen, Salla, Waseem, Muhammad, Yevstihnyeyev, Yuriy, "Aberg, Verneri, Åstrand, Leif., 2025. Containers as the quantum leap in software development. https://arxiv.org/abs/2501.07204.

Al-shammari, Mohammad Matar, Alwan, Ali Amer, 2018. Disaster recovery and business continuity for database services in multi-cloud. In: Proceedings of the 1st International Conference on Computer Applications & Information Security. ICCAIS, IEEE, pp. 1–8.

Alaluna, Max, Vial, Eric, Neves, Nuno, Ramos, Fernando M.V., 2017. Secure and dependable multi-cloud network virtualization. In: Proceedings of the 1st International Workshop on Security and Dependability of Multi-Domain Infrastructures. pp. 1–6.

Alhamazani, Khalid, Ranjan, Rajiv, Mitra, Karan, Jayaraman, Prem Prakash, Huang, Zhiqiang, Wang, Lizhe, Rabhi, Fethi, 2014. Clams: Cross-layer multi-cloud application monitoring-as-a-service framework. In: Proceedings of the 11th International Conference on Services Computing. IEEE, pp. 283–290.

Almeida, André, Cavalcante, Everton, Batista, Thais, Cacho, Nelio, Lopes, Frederico, 2014. A component-based adaptation approach for multi-cloud applications. In: Proceedings of the 7th International Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE, pp. 49–54.

Alonso, Juncal, Orue-Echevarria, Leire, Casola, Valentina, Torre, Ana Isabel, Huarte, Maider, Osaba, Eneko, Lobo, Jesus L, 2023a. Understanding the challenges and novel architectural models of multi-cloud native applications–a systematic literature review. J. Cloud Comput. 12 (1), 6.

Alonso, Juncal, Orue-Echevarria, Leire, Casola, Valentina, Torre, Ana Isabel, Huarte, Maider, Osaba, Eneko, Lobo, Jesus L, 2023b. Understanding the challenges and novel architectural models of multi-cloud native applications–a systematic literature review. J. Cloud Comput. 12 (1), 1–34.

Alonso, Juncal, Stefanidis, Kyriakos, Orue-Echevarria, Leire, Blasi, Lorenzo, Walker, Michael, Escalante, Marisa, López, María José, Dutkowski, Simon, 2019. DECIDE: an extended devops framework for multi-cloud applications. In: Proceedings of the 3rd International Conference on Cloud and Big Data Computing. ICCBDC, pp. 43–48.

Altran, Luiz Fernando, Galante, Guilherme, Oyamada, Marcio Seiji, 2022. Label-affinity-scheduler: Considering business requirements in container scheduling for multi-cloud and multi-tenant environments. In: Proceedings of the 12th Brazilian Symposium on Computing Systems Engineering. SBESC, IEEE, pp. 1–8.

Aral, Atakan, Uriarte, Rafael Brundo, Simonet-Boulogne, Anthony, Brandic, Ivona, 2020. Reliability management for blockchain-based decentralized multi-cloud. In: Proceedings of the 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing. CCGRID, IEEE, pp. 21–30.

Austin, Greg, 2018. Cybersecurity in China: The Next Wave. Springer.

Awada, Uchechukwu, 2018. Application-container orchestration tools and platform-as-a-service clouds: A survey. Int. J. Adv. Comput. Sci. Appl..

Baby, Kiran, Vysala, Anupriya, 2015. Multicloud architecture for augmenting security in clouds. In: Proceedings of the 1st Global Conference on Communication Technologies. GCCT, IEEE, pp. 474–478.

Bachiega, Naylor G, Souza, Paulo SL, Bruschi, Sarita M, De Souza, Simone Do RS, 2018. Container-based performance evaluation: A survey and challenges. In: Proceedings of the 6th IEEE International Conference on Cloud Engineering (IC2E). IEEE, pp. 398–403.

Balalaie, Armin, Heydarnoori, Abbas, Jamshidi, Pooyan, 2016. Microservices architecture enables devops: Migration to a cloud-native architecture. IEEE Softw. 33 (3), 42–52.

Barati, Masoud, Adu-Duodu, Kwabena, Rana, Omer, Aujla, Gagangeet Singh, Ranjan, Rajiv, 2023. Compliance checking of cloud providers: design and implementation. Distrib. Ledger Technol.: Res. Pr. 2 (2), 1–20.

Baresi, Luciano, Guinea, Sam, Quattrocchi, Giovanni, Tamburri, Damian A, 2016. Microcloud: A container-based solution for efficient resource management in the cloud. In: Proceedings of the 1st International Conference on Smart Cloud (SmartCloud). IEEE, pp. 218–223.

Barletta, Marco, Cinque, Marcello, De Simone, Luigi, Corte, Raffaele Della, 2024. Criticality-aware monitoring and orchestration for containerized industry 4.0 environments. ACM Trans. Embed. Comput. Syst. 23 (1), 1–28.

Barna, Cornel, Khazaei, Hamzeh, Fokaefs, Marios, Litoiu, Marin, 2017. Delivering elastic containerized cloud applications to enable DevOps. In: Proceedings of the 12th IEEE/ACM International Symposium on Software Engineering for Adaptive and Self-Managing Systems. SEAMS, IEEE, pp. 65–75.

Bélair, Maxime, Laniepce, Sylvie, Menaud, Jean-Marc, 2019. Leveraging kernel security mechanisms to improve container security: a survey. In: Proceedings of the 14th International Conference on Availability, Reliability and Security. pp. 1–6.

Benmerar, Tarik Zakaria, Theodoropoulos, Theodoros, Fevereiro, Diogo, Rosa, Luis, Rodrigues, João, Taleb, Tarik, Barone, Paolo, Tserpes, Konstantinos, Cordeiro, Luis, 2023. Intelligent multi-domain edge orchestration for highly distributed immersive services: an immersive virtual touring use case. In: Proceedings of the IEEE International Conference on Edge Computing and Communications. EDGE, IEEE, pp. 381–392.

Bentaleb, Ouafa, Belloum, Adam SZ, Sebaa, Abderrazak, El-Maouhab, Aouaouche, 2022. Containerization technologies: Taxonomies, applications and challenges. J. Supercomput. 78 (1), 1144–1181.

Bhamare, Deval, Jain, Raj, Samaka, Mohammed, Vaszkun, Gabor, Erbad, Aiman, 2015. Multi-cloud distribution of virtual functions and dynamic service deployment: Open ADN perspective. In: Proceedings of the 3rd IEEE International Conference on Cloud Engineering. IEEE, pp. 299–304.

Bisht, Sankalp Singh, Kaur, Parmeet, 2022. An empirical investigation of a fault tolerant containerized application deployment. In: Proceedings of the 1st International Conference on Informatics. ICI, IEEE, pp. 171–175.

Bokhari, Mohammad Ubaidullah, Makki, Qahtan, Tamandani, Yahya Kord, 2018. A survey on cloud computing. In: Proceedings of CSI 2015 on Big Data Analytics. CSI, Springer, pp. 149–164.

Bolivar, Luis Tomas, Tselios, Christos, Area, Daniel Mellado, Tsolis, George, 2018. On the deployment of an open-source, 5G-aware evaluation testbed. In: Proceedings of the 6th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud). IEEE, pp. 51–58.

Bordeleau, Francis, Combemale, Benoit, Eramo, Romina, van den Brand, Mark, Wimmer, Manuel, 2020. Towards model-driven digital twin engineering: Current opportunities and future challenges. In: Proceedings of the 1st International Conference on Systems Modelling and Management. ICSMM, Vol. 1, Springer International Publishing, pp. 43–54.

Bou Ghantous, Georges, Gill, Asif Qumer, 2021. Evaluating the DevOps reference architecture for multi-cloud IoT-Applications. SN Comput. Sci. 2, 1–35.

Braun, Virginia, Clarke, Victoria, 2006. Using thematic analysis in psychology. Qual. Res. Psychol. 3 (2), 77–101.

Bucur, Vlad, Dehelean, Catalin, Miclea, Liviu, 2018. Object storage in the cloud and multi-cloud: State of the art and the research challenges. In: Proceedings of the 7th International Conference on Automation, Quality and Testing, Robotics. AQTR, IEEE, pp. 1–6.

Buyya, Rajkumar, Barreto, Diana, 2015. Multi-cloud resource provisioning with aneka: A unified and integrated utilisation of microsoft azure and amazon EC2 instances. In: Proceedings of the 1st International Conference on Computing and Network Communications (CoCoNet). IEEE, pp. 216–229.

Buyya, Rajkumar, Ranjan, Rajiv, Calheiros, Rodrigo N., 2010. Intercloud: Utility-oriented federation of cloud computing environments for scaling of application services. In: Processing of the 10th International Conference on Algorithms and Architectures for Parallel. Springer, pp. 13–31.

Buyya, Rajkumar, Srirama, Satish Narayana, Casale, Giuliano, Calheiros, Rodrigo, Simmhan, Yogesh, Varghese, Blesson, Gelenbe, Erol, Javadi, Bahman, Vaquero, Luis Miguel, Netto, Marco AS, et al., 2020. A manifesto for future generation cloud computing: Research directions for the next decade. ACM Comput. Surv. 51 (5), 1–38.

Caldiera, Victor R. Basili1 Gianluigi, Rombach, H. Dieter, 1994. The goal question metric approach. Encycl. Softw. Eng. 528–532.

Carpio, Francisco, Delgado, Marta, Jukan, Admela, 2020. Engineering and experimentally benchmarking a container-based edge computing system. In: Proceedings of the 33rd International Conference on Communications. ICC, IEEE, pp. 1–6.

Casalicchio, Emiliano, 2019. Container orchestration: A survey. Syst. Model.: Methodol. Tools 221–235.

Casalicchio, Emiliano, Iannucci, Stefano, 2020. The state-of-the-art in container technologies: Application, orchestration and security. Concurr. Comput.: Pr. Exp. 32 (17), e5668.

Casola, Valentina, De Benedictis, Alessandra, Rak, Massimiliano, Villano, Umberto, Rios, Erkuden, Rego, Angel, Capone, Giancarlo, 2017. MUSA deployer: Deployment of multi-cloud applications. In: Proceedings of the 26th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises. WETICE, IEEE, pp. 107–112.

del Castillo, Juan Angel Lorenzo, Mallichan, Kate, Al-Hazmi, Yahya, 2013. Openstack federation in experimentation multi-cloud testbeds. In: Proceedings of the 5th International Conference on Cloud Computing Technology and Science. Vol. 2, IEEE, pp. 51–56.

Centofanti, Carlo, Santos, José, Gudepu, Venkateswarlu, Kondepu, Koteswararao, 2024. Impact of power consumption in containerized clouds: A comprehensive analysis of open-source power measurement tools. Comput. Netw. 245, 110371.

Challita, Stéphanie, Paraiso, Fawaz, Merle, Philippe, 2017. Towards formal-based semantic interoperability in multi-clouds: the fclouds framework. In: Proceedings of the 10th International Conference on Cloud Computing. CLOUD, IEEE, pp. 710–713.

Chandler, Jacqueline, Cumpston, Miranda, Li, Tianjing, Page, Matthew J, Welch, VJHW, 2019. Cochrane Handbook for Systematic Reviews of Interventions. Wiley, Hoboken.

Chen, Jacky, Lim, Chee Peng, Tan, Kim Hua, Govindan, Kannan, Kumar, Ajay, 2021. Artificial intelligence-based human-centric decision support framework: an application to predictive maintenance in asset management under pandemic environments. Ann. Oper. Res. 1–24.

Chen, Yuheng, Shi, Tao, Ma, Hui, Chen, Gang, 2022. Automatically design heuristics for multi-objective location-aware service brokering in multi-cloud. In: Proceedings of the 19th International Conference on Services Computing. SCC, IEEE, pp. 206–214.

Cohen, Jacob, 1960. A coefficient of agreement for nominal scales. Educ. Psychol. Meas. 20 (1), 37–46.

Cohen, Jacob, 1968. Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. Psychol. Bull. 70 (4), 213.

Combe, T., Martin, A., Di Pietro, R., 2016. To docker or not to docker: A security perspective. IEEE Cloud Comput. 3 (5), 54–62.

Cui, Wei, Zhan, Hanglong, Li, Bao, Wang, Hu, Cao, Donggang, 2016. Cluster as a service: A container based cluster sharing approach with multi-user support. In: Proceedings of the 2016 Symposium on Service-Oriented System Engineering. SOSE, IEEE, pp. 111–118.

de Laat, Maarten, Joksimovic, Srecko, Ifenthaler, Dirk, 2020. Artificial intelligence, real-time feedback and workplace learning analytics to support in situ complex problem-solving: A commentary. Int. J. Inf. Learn. Technol. 37 (5), 267–277.

Debroy, Vidroha, Miller, Senecca, Brimble, Lance, 2018. Building lean continuous integration and delivery pipelines by applying devops principles: a case study at varidesk. In: Proceedings of the 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. pp. 851–856.

Deng, Qiqing, Tan, Xinrui, Yang, Jing, Zheng, Chao, Wang, Liming, Xu, Zhen, 2022. A secure container placement strategy using deep reinforcement learning in cloud. In: Proceedings of the 25th International Conference on Computer Supported Cooperative Work in Design. CSCWD, IEEE, pp. 1299–1304.

Di Modica, Giuseppe, Tomarchio, Orazio, Wei, Hao, Rodriguez, Joaquin Salvachua, et al., 2019. Policy-based deployment in a hybrid and multicloud environment. In: CLOSER. pp. 388–395.

Dimitrijevic, Zoran, Sahin, Cetin, Tinnefeld, Christian, Patvarczki, Jozsef, 2019. Importance of application-level resource management in multi-cloud deployments. In: Proceedings of the 7th International Conference on Cloud Engineering (IC2E). IEEE, pp. 139–144.

Docker, 2021. What is a container?. https://docs.docker.com/get-started/overview/#what-is-a-container.

Docker, 2023. Use containers to build, share, and run your applications. https://www.docker.com/resources/what-container/. (Accessed 20 August 2023).

Dogani, Javad, Namvar, Reza, Khunjush, Farshad, 2023. Auto-scaling techniques in container-based cloud and edge/fog computing: Taxonomy and survey. Comput. Commun. 209, 120–150.

Dolui, Koustabh, Kiraly, Csaba, 2018. Towards multi-container deployment on IoT gateways. In: Proceedings of the 34th Global Communications Conference (GlobeCom). IEEE, pp. 1–7.

Dragoni, Nicola, Giallorenzo, Saverio, Lluch-Lafuente, Alberto, Mazzara, Manuel, Montesi, Fabrizio, Mustafin, Ruslan, Safina, Larisa, 2017. Microservices: yesterday, today, and tomorrow. In: Present and Ulterior Software Engineering. Springer, pp. 195–216.

Dreibholz, Thomas, Mazumdar, Somnath, Zahid, Feroz, Taherkordi, Amir, Gran, Ernst Gunnar, 2019. Mobile edge as part of the multi-cloud ecosystem: a performance study. In: Proceedings of the 27th Euromicro International Conference on Parallel, Distributed and Network-Based Processing. PDP, IEEE, pp. 59–66.

Fadda, Edoardo, Plebani, Pierluigi, Vitali, Monica, 2019. Monitoring-aware optimal deployment for applications based on microservices. IEEE Trans. Serv. Comput. 14 (6), 1849–1863.

Fernandes, Diogo AB, Soares, Liliana FB, Gomes, João V, Freire, Mário M, Inácio, Pedro RM, 2014. Security issues in cloud environments: a survey. Int. J. Inf. Secur. 13, 113–170.

Ferrer, Ana Juan, Pérez, David García, González, Román Sosa, 2016. Multi-cloud platform-as-a-service model, functionalities and approaches. Procedia Comput. Sci. 97, 63–72.

Ferry, Nicolas, Rossini, Alessandro, Chauvel, Franck, Morin, Brice, Solberg, Arnor, 2013. Towards model-driven provisioning, deployment, monitoring, and adaptation of multi-cloud systems. In: Proceedings of the 6th International Conference on Cloud Computing. ICCC, IEEE, pp. 887–894.

Ferry, Nicolas, Song, Hui, Rossini, Alessandro, Chauvel, Franck, Solberg, Arnor, 2014. Cloudmf: applying MDE to tame the complexity of managing multi-cloud applications. In: Proceedings of the 7th International Conference on Utility and Cloud Computing. IEEE, pp. 269–277.

Fowler, Martin, 2010. Continuous delivery. https://martinfowler.com/bliki/BlueGreenDeployment.html.

Fugkeaw, Somchart, 2023. Achieving decentralized and dynamic sso-identity access management system for multi-application outsourced in cloud. IEEE Access 11, 25480–25491.

Gao, Xing, Gu, Zhongshu, Kayaalp, Mehmet, Pendarakis, Dimitrios, Wang, Haining, 2017. Containerleaks: Emerging security threats of information leakages in container clouds. In: Proceedings of the 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks. DSN, IEEE, pp. 237–248.

Gattobigio, Luca, Thielemans, Steffen, Benedetti, Priscilla, Reali, Gianluca, Braeken, An, Steenhaut, Kris, 2022. A multi-cloud service mesh approach applied to internet of things. In: Proceedings of the 48th Annual Conference of the IEEE Industrial Electronics Society. IECON, IEEE, pp. 1–6.

Gill, Sukhpal Singh, Xu, Minxian, Ottaviani, Carlo, Patros, Panos, Bahsoon, Rami, Shaghaghi, Arash, Golec, Muhammed, Stankovski, Vlado, Wu, Huaming, Abraham, Ajith, et al., 2022. AI for next generation computing: Emerging trends and future directions. Internet Things 19, 100514.

Gillam, Lee, Li, Bin, O'Loughlin, John, 2013. Benchmarking cloud performance for service level agreement parameters. Int. J. Cloud Comput. 2 (1), 3–23.

Glaser, Barney G., Strauss, Anselm L., 1967. The Discovery of Grounded Theory: Strategies for Qualitative Research. Aldine Publishing Company, Chicago.

Graupner, Hendrik, Torkura, Kennedy, Berger, Philipp, Meinel, Christoph, Schnjakin, Maxim, 2015. Secure access control for multi-cloud resources. In: Proceedings of the 40th Local Computer Networks Conference Workshops (LCN Workshops). IEEE, pp. 722–729.

Guechi, Farida Ali, Maamri, Ramdane, 2018. Secure and Parallel Expressive Search over Encrypted Data with Access Control in Multi-CloudIoT. In: Proceedings of the 3rd Cloudification of the Internet of Things (CIoT), pp. 1–8.

Guerrero, Carlos, Lera, Isaac, Juiz, Carlos, 2018. Resource optimization of container orchestration: a case study in multi-cloud microservices-based applications. J. Supercomput. 74 (7), 2956–2983.

Gundu, Srinivasa Rao, Panem, Charan Arur, Thimmapuram, Anuradha, 2020. Hybrid IT and multi cloud an emerging trend and improved performance in cloud computing. SN Comput. Sci. 1 (5), 256.

Gupta, Rohan Raj, Mishra, Gaurav, Katara, Subham, Agarwal, Arpit, Sarkar, Mrinal Kanti, Das, Rupayan, Kumar, Sanjay, 2016. Data storage security in cloud computing using container clustering. In: Proceedings of the 7th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference. UEMCON, IEEE, pp. 1–7.

Gupta, Manu, Sanjana, Konte, Akhilesh, Kontham, Chowdary, Mandepudi Nobel, 2021. Deployment of multi-tier application on cloud and continuous monitoring using kubernetes. In: Proceedings of the 5th International Conference on Electrical, Electronics, Communication, Computer Technologies and Optimization Techniques. ICEECCOT, IEEE, pp. 602–607.

Harwalkar, Sudheendra, Sitaram, Dinkar, Jadon, Shivangi, Kidiyoor, Dhanaraj V, D'souza, Ornella, 2019a. Private STaaS with OpenStack cinder volumes for hybrid/multi-cloud. In: Proceedings of the 4th International Conference on Advances in Computing and Communication Engineering. ICACCE, IEEE, pp. 1–6.

Harwalkar, Sudheendra, Sitaram, Dinkar, Kidiyoor, Dhanaraj V, Milan, ML, D'souza, Ornella, Agarwal, Radhika, Agarwal, Yamini, 2019b. Multicloud-auto scale with prediction and delta correction algorithm. In: Proceedings of the 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies. ICICICT, Vol. 1, IEEE, pp. 227–233.

Harzenetter, Lukas, Breitenbücher, Uwe, Binz, Tobias, Leymann, Frank, 2023. An integrated management system for composed applications deployed by different deployment automation technologies. SN Comput. Sci. 4 (4), 370.

Hashem, Ibrahim Abaker Targio, Yaqoob, Ibrar, Anuar, Nor Badrul, Mokhtar, Salimah, Gani, Abdullah, Khan, Samee Ullah, 2016. The rise of "big data" on cloud computing: Review and open research issues. Inf. Syst. 47, 98–115.

He, Heng, Liu, Jiaqi, Gu, Jinguang, Gao, Feng, 2022. An efficient multi-keyword search scheme over encrypted data in multi-cloud environment. In: 2022 IEEE 7th International Conference on Smart Cloud (SmartCloud). IEEE, pp. 59–67.

Helali, Leila, Omri, Mohamed Nazih, 2021. A survey of data center consolidation in cloud computing systems. Comput. Sci. Rev. 39, 100366.

Horchulhack, Pedro, Viegas, Eduardo K, Santin, Altair O, Ramos, Felipe V, Tedeschi, Pietro, 2024. Detection of quality of service degradation on multi-tenant containerized services. J. Netw. Comput. Appl. 224, 103839.

Hossain, Md Delowar, Sultana, Tangina, Akhter, Sharmen, Hossain, Md Imtiaz, Thu, Ngo Thien, Huynh, Luan NT, Lee, Ga-Won, Huh, Eui-Nam, 2023. The role of microservice approach in edge computing: Opportunities, challenges, and research directions. ICT Express.

Hua, Lei, Tang, Ting, Wu, Heng, Yuewen, WU, He, LIU, Yuanjia, XU, Zhang, Wenbo, 2020. A framework to support multi-cloud collaboration. In: Proceedigs of the 13th World Congress on Services. SERVICES, IEEE, pp. 110–116.

Huang, Zongwei, Wang, Qi, 2023. Industrial robot control system optimized by wireless resources and cloud resources based on cloud edge multi-cluster containers. Int. J. Syst. Assur. Eng. Manag. 14 (2), 538–547.

Imran, Hamza Ali, Latif, Usama, Ikram, Ataul Aziz, Ehsan, Maryam, Ikram, Ahmed Jamal, Khan, Waleed Ahmad, Wazir, Saad, 2020. Multi-cloud: a comprehensive review. In: Proceedings of the 23rd International Multitopic Conference. INMIC, IEEE, pp. 1–5.

Jamshidi, Pooyan, Ahmad, Aakash, Pahl, Claus, 2013. Cloud migration research: a systematic review. IEEE Trans. Cloud Comput. 1 (2), 142–157.

Janarthanan, Kugathasan, Peramune, PRLC, Ranaweera, AT, Krishnamohan, Theviyanthan, Rupasinghe, Lakmal, Sampath, Kalpa Kalhara, Liyanapathirana, Chethana, 2018. Policies based container migration using cross-cloud management platform. In: Proccedings of the 8th International Conference on Information and Automation for Sustainability (ICIAfS). IEEE, pp. 1–6.

Jha, Devki Nandan, Wen, Zhenyu, Li, Yinhao, Nee, Michael, Koutny, Maciej, Ranjan, Rajiv, 2019. A cost-efficient multi-cloud orchestrator for benchmarking containerized web-applications. In: Proceedings of the 20th International Conference on Web Information Systems Engineering. WISE, Springer, pp. 407–423.

Kataru, Shanmukh Sai, Gude, Rohith, Shaik, Shoaib, Kota, Lalithesh VNSS Sathvik, Srithar, S, Balajee, RM, 2023. Cost optimizing cloud based docker application deployment with cloudfront and global accelerator in AWS cloud. In: Proceedings of the International Conference on Sustainable Communication Networks and Application. ICSCNA, IEEE, pp. 200–208.

Keele, Staffs, et al., 2007. Guidelines for Performing Systematic Literature Reviews in Software Engineering. Technical Report, Citeseer.

Kiss, Tamas, 2018. Scalable multi-cloud platform to support industry and scientific applications. In: Proceedings of the 41st International Convention on Information and Communication Technology, Electronics and Microelectronics. MIPRO, IEEE, pp. 0150–0154.

Kritikos, Kyriakos, Zeginis, Chrysostomos, Iranzo, Joaquin, Gonzalez, Roman Sosa, Seybold, Daniel, Griesinger, Frank, Domaschka, Jörg, 2019. Multi-cloud provisioning of business processes. J. Cloud Comput. 8 (1), 1–29.

Lazuka, Malgorzata, Parnell, Thomas, Anghel, Andreea, Pozidis, Haralampos, 2022. Search-based methods for multi-cloud configuration. In: Proceedings of the 15th International Conference on Cloud Computing. CLOUD, IEEE, pp. 438–448.

Lee, Seungsoo, Nam, Jaehyun, 2023. Kunerva: Automated network policy discovery framework for containers. IEEE Access.

Li, Yuanbo, Hu, Hongchao, Liu, Wenyan, Yang, Xiaohan, 2023. An optimal active defensive security framework for the container-based cloud with deep reinforcement learning. Electronics 12 (7), 1598.

Libardi, Rafael Mira De Oliveira, Bedo, Marcos Vinicius Naves, Reiff-Marganiec, Stephan, Estrella, Julio Cezar, 2014. MSSF: A step towards user-friendly multi-cloud data dispersal. In: Proceedings of the 7th International Conference on Cloud Computing. IEEE, pp. 952–953.

Liu, Jiali, Qin, Yuqin, 2024. CSPUMS: Pioneering integrated monitoring in multi-service provider ecosystems. In: Proceedings of the 9th International Conference on Cloud Computing and Big Data Analytics. ICCCBDA, IEEE, pp. 211–215.

Lorenzo, C., Guillaume, P., Bellavista, P., 2019. FogDocker: Start container now fetch image later. In: Proceedings of the 12th IEEE/ACM International Conference on Utility and Cloud Computing. UCC.

Maenhaut, Pieter-Jan, Volckaert, Bruno, Ongenae, Veerle, De Turck, Filip, 2020. Resource management in a containerized cloud: Status and challenges. J. Netw. Syst. Manage. 28, 197–246.

Marques, Gonçalo, Senna, Carlos, Sargento, Susana, Carvalho, Luís, Pereira, Luís, Matos, Ricardo, 2024. Proactive resource management for cloud of services environments. Future Gener. Comput. Syst. 150, 90–102.

Mell, Peter, Grance, Tim, et al., 2011. The NIST definition of cloud computing.

Merkel, Dirk, et al., 2014. Docker: lightweight linux containers for consistent development and deployment. Linux J. 239 (2), 2.

Merlino, Giovanni, Tricomi, Giuseppe, D'agati, Luca, Benomar, Zakaria, Longo, Francesco, Puliafito, Antonio, 2024. Faas for IoT: Evolving serverless towards deviceless in I/Oclouds. Future Gener. Comput. Syst. 154, 189–205.

Mfula, Harrison, Nurminen, Jukka K., 2018. Self-healing cloud services in private multi-clouds. In: Proceedings of the 10th International Conference on High Performance Computing & Simulation. HPCS, IEEE, pp. 165–170.

Mihai, Stefan, Yaqoob, Mahnoor, Hung, Dang V., Davis, William, Towakel, Praveer, Raza, Mohsin, Karamanoglu, Mehmet, et al., 2022. Digital twins: A survey on enabling technologies, challenges, trends and future prospects. IEEE Commun. Surv. Tutor..

Mittal, Saurabh, Risco-Martín, José L., 2017. DEVSML 3.0 stack: rapid deployment of DEVS farm in distributed cloud environment using microservices and containers. In: Proceedings of the 2017 Symposium on Theory of Modeling & Simulation. pp. 1–12.

Mungoli, Neelesh, 2023. Scalable, distributed AI frameworks: Leveraging cloud computing for enhanced deep learning performance and efficiency. arXiv preprint arXiv:2304.13738.

Nagarajan, S, Rani, P Shobha, Vinmathi, MS, Subba Reddy, V, Saleth, Angel Latha Mary, Abdus Subhahan, D, 2023. Multi agent deep reinforcement learning for resource allocation in container-based clouds environments. Expert Syst..

Naik, Nitin, 2016a. Applying computational intelligence for enhancing the dependability of multi-cloud systems using docker swarm. In: Proceedings of the 2nd Symposium Series on Computational Intelligence. SSCI, IEEE, pp. 1–7.

Naik, Nitin, 2016b. Building a virtual system of systems using docker swarm in multiple clouds. In: Proceedings of the 2nd International Symposium on Systems Engineering. ISSE, IEEE, pp. 1–3.

Nardelli, Matteo, Cardellini, Valeria, Casalicchio, Emiliano, 2018. Multi-level elastic deployment of containerized applications in geo-distributed environments. In: Proceedings of the 6th IEEE International Conference on Future Internet of Things and Cloud (FiCloud). IEEE, pp. 1–8.

National Institute of Standards and Technology, 2018. NIST Cyber Security Framework Version 1.1. Technical Report.

Neeraj, NK, Nellikeri, Aditya, Varun, P, Reddy, Santosh, Shanbhag, Mangesh, Narayan, Dg, Husain, Altaf, 2023. Service level agreement violation detection in multi-cloud environment using ethereum blockchain. In: Proceedings of the International Conference on Networking and Communications. ICNWC, IEEE, pp. 1–7.

Netaji, Vhatkar Kapil, Bhole, Girish P., 2021. A comprehensive survey on container resource allocation approaches in cloud computing: State-of-the-art and research challenges. In: Web Intelligence. Vol. 19, IOS Press, pp. 295–316.

Nguyen, Hai Duc, Chien, Andrew A., 2023. Storm-RTS: Stream processing with stable performance for multi-cloud and cloud-edge. In: Proceedings of the 16th IEEE International Conference on Cloud Computing. CLOUD, IEEE, pp. 45–57.

Ongaro, Diego, Ousterhout, John, 2014. In search of an understandable consensus algorithm. In: Proceedings of the 2014 USENIX Annual Technical Conference (USENIX ATC). USENIX, pp. 305–319.

Orzechowski, Michał, Wrzeszcz, Michał, Kryza, Bartosz, Dutka, Łukasz, Słota, Renata G, Kitowski, Jacek, 2023. Indexing legacy data-sets for global access and processing in multi-cloud environments. Future Gener. Comput. Syst. 148, 150–159.

Ouyang, Mingxue, Xi, Jianqing, Bai, Weihua, Li, Keqin, 2022. Band-area resource management platform and accelerated particle swarm optimization algorithm for container deployment in internet-of-things cloud. IEEE Access 10, 86844–86863.

Pahl, Claus, 2015. Containerization and the paas cloud. IEEE Cloud Comput. 2 (3), 24–31.

Pahl, Claus, Brogi, Antonio, Soldani, Jacopo, Jamshidi, Pooyan, 2017. Cloud container technologies: a state-of-the-art review. IEEE Trans. Cloud Comput. 7 (3), 677–692.

Pai, Pradeep, Kumar, CRS, 2021. Building cloud native application—analysis for multi-component application deployment. In: Proceedings of the 10th International Conference on Computer Communication and Informatics. ICCCI, IEEE, pp. 1–6.

Paladi, Nicolae, Michalas, Antonis, Dang, Hai-Van, 2018. Towards secure cloud orchestration for multi-cloud deployments. In: Proceedings of the 5th Workshop on CrossCloud Infrastructures & Platforms (CrossCloud). ACM, pp. 1–6.

Pandey, Ashish, Calyam, Prasad, Lyu, Zhen, Wang, Songjie, Chemodanov, Dmitrii, Joshi, Trupti, 2022. Knowledge-engineered multi-cloud resource brokering for application workflow optimization. IEEE Trans. Netw. Serv. Manag..

Pandi, S. Senthil, Kumar, P., Suchindhar, R.M., 2023. Integrating jenkins for efficient deployment and orchestration across multi-cloud environments. In: Proceedings of the International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems. ICSES, IEEE, pp. 1–6.

Pellegrini, Roland, Rottmann, Patrick, Strieder, Georg, 2017. Preventing vendor lock-ins via an interoperable multi-cloud deployment approach. In: Proceedings of the 12th International Conference for Internet Technology and Secured Transactions. ICITST, IEEE, pp. 382–387.

Petcu, Dana, 2011. Portability and interoperability between clouds: challenges and case study. In: Proceedings of the 4th European Conference on Towards a Service-Based Internet (ServiceWave). Springer, pp. 62–74.

Petcu, Dana, 2013. Multi-cloud: expectations and current approaches. In: Proceedings of the International Workshop on Multi-Cloud Applications and Federated Clouds (MultiCloud). ACM, pp. 1–6.

Petcu, Dana, Martino, Beniamino Di, Venticinque, Salvatore, Rak, Massimiliano, Máhr, Tamás, Lopez, Gorka Esnal, Brito, Fabrice, Cossu, Roberto, Stopar, Miha, Šperka, Svatopluk, et al., 2018. Experiences in building a mOSAIC of clouds. J. Cloud Comput. 1 (1), 1–23.

Petersen, Kai, Feldt, Robert, Mujtaba, Shahid, Mattsson, Michael, 2008. Systematic mapping studies in software engineering. In: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering. EASE, ACM, pp. 1–10.

Petrovski, Toshe, Gusev, Marjan, 2024. Container vs function as a service: Impact on cloud deployment for real-world applications. In: Proceedings of the 47th MIPRO ICT and Electronics Convention. MIPRO, IEEE, pp. 869–874.

Pham, Linh Manh, Pham, Tuan-Minh, 2015. Autonomic fine-grained migration and replication of component-based applications across multi-clouds. In: Proceedings of the 2nd National Foundation for Science and Technology Development Conference on Information and Computer Science. NICS, IEEE, pp. 5–10.

Pham, Linh Manh, Tchana, Alain, Donsez, Didier, Zurczak, Vincent, Gibello, Pierre-Yves, De Palma, Noel, 2015. An adaptable framework to deploy complex applications onto multi-cloud platforms. In: Proceedings of the 8th International Conference on Computing & Communication Technologies-Research, Innovation, and Vision for Future. RIVF, IEEE, pp. 169–174.

Pop, Daniel, Iuhasz, Gabriel, Craciun, Ciprian, Panica, Silviu, 2016. Support services for applications execution in multi-clouds environments. In: Proceedings of the 7th IEEE International Conference on Autonomic Computing. ICAC, IEEE, pp. 343–348.

Quint, Peter-Christian, Kratzke, Nane, 2018. Towards a lightweight multi-cloud DSL for elastic and transferable cloud-native applications. arXiv preprint arXiv:1802.03562.

Rahman, Joy, Lama, Palden, 2019. Predicting the end-to-end tail latency of containerized microservices in the cloud. In: Proceedings of the 7th International Conference on Cloud Engineering (IC2E). IEEE, pp. 200–210.

Raj, Pethuru, Raman, Anupama, Raj, Pethuru, Raman, Anupama, 2018. Automated multi-cloud operations and container orchestration. Software- Defin. Cloud Cent.: Oper. Manag. Technol. Tools 185–218.

Ralph, Paul, Ali, Nauman bin, Baltes, Sebastian, Bianculli, Domenico, Diaz, Jessica, Dittrich, Yvonne, Ernst, Neil, Felderer, Michael, Feldt, Robert, Filieri, Antonio, et al., 2020. Empirical standards for software engineering research. arXiv preprint arXiv:2010.03525.

Ramesh, Manju, Chahal, Dheeraj, Singhal, Rekha, 2023. Multicloud deployment of ai workflows using faas and storage services. In: Proceedings of the 15th International Conference on COMmunication Systems & NETworkS. COMSNETS, IEEE, pp. 269–277.

Ramos, Felipe, Viegas, Eduardo, Santin, Altair, Horchulhack, Pedro, dos Santos, Roger R, Espindola, Allan, 2021. A machine learning model for detection of docker-based APP overbooking on kubernetes. In: Proceedings of the 34th International Conference on Communications. IEEE, pp. 1–6.

Ranjbar, Alireza, Komu, Miika, Salmela, Patrik, Aura, Tuomas, 2017. Synaptic: Secure and persistent connectivity for containers. In: Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. CCGRID, IEEE, pp. 262–267.

Regulation, General Data Protection, 2016. General data protection regulation (GDPR)–official legal text. Gen. Data Prot. Regul..

Rios, Erkuden, Iturbe, Eider, Mallouli, Wissam, Rak, Massimiliano, 2017. Dynamic security assurance in multi-cloud DevOps. In: Proceedings of the 5th IEEE Conference on Communications and Network Security. CNS, IEEE, pp. 467–475.

Rodigari, Simone, O'Shea, Donna, McCarthy, Pat, McCarry, Martin, McSweeney, Sean, 2021. Performance analysis of zero-trust multi-cloud. In: Proceedings of the 14th International Conference on Cloud Computing. CLOUD, IEEE, pp. 730–732.

Roy, Swarnabha, Kalafatis, Stavros, 2023. RoboCon: A modular robotic containerization, orchestration, and load balancing technique for mixed hierarchy task handling across computing platforms. In: Proceeings of the 5th International Conference on Control and Robotics. ICCR, IEEE, pp. 120–129.

Sabbioni, Andrea, Bujari, Armir, Foschini, Luca, Corradi, Antonio, 2020. An efficient and reliable multi-cloud provider monitoring solution. In: Proceedings of the 35th Global Communications Conference. GLOBECOM, IEEE, pp. 1–6.

Saif, Mufeed Ahmed Naji, Niranjan, SK, Murshed, Belal Abdullah Hezam, Al-ariki, Hasib Daowd Esmail, Abdulwahab, Hudhaifa Mohammed, 2022. Multi-agent qos-aware autonomic resource provisioning framework for elastic BPM in containerized multi-cloud environment. J. Ambient. Intell. Humaniz. Comput. 1–26.

Saif, Mufeed Ahmed Naji, Niranjan, SK, Murshed, Belal Abdullah Hezam, Al-ariki, Hasib Daowd Esmail, Abdulwahab, Hudhaifa Mohammed, 2023. Multi-agent qos-aware autonomic resource provisioning framework for elastic BPM in containerized multi-cloud environment. J. Ambient. Intell. Humaniz. Comput. 14 (9), 12895–12920.

Sami, Hani, Mourad, Azzam, 2020. Dynamic on-demand fog formation offering on-the-fly IoT service deployment. IEEE Trans. Netw. Serv. Manag. 17 (2), 1026–1039.

Santos, Álvaro, Correia, Noélia, Bernardino, Jorge, 2023. On the suitability of cloud models for MEC deployment purposes. In: Proceedings of the 6th Experiment@ International Conference. EXPAT, IEEE, pp. 255–260.

Sarker, Iqbal H., 2022. AI-based modeling: Techniques, applications and research issues towards automation, intelligent and smart systems. SN Comput. Sci. 3 (2), 158.

Saxena, Deepika, Gupta, Rishabh, Singh, Ashutosh Kumar, 2021. A survey and comparative study on multi-cloud architectures: emerging issues and challenges for cloud federation. arXiv preprint arXiv:2108.12831.

Schardt, Connie, Adams, Martha B, Owens, Thomas, Keitz, Sheri, Fontelo, Paul, 2007. Utilization of the PICO framework to improve searching PubMed for clinical questions. BMC Med. Inform. Decis. Mak. 7, 1–6.

Sedghpour, Mohammad Reza Saleh, Townend, Paul, 2022. Service mesh and ebpf-powered microservices: A survey and future directions. In: Proceedings of the 2022 International Conference on Service-Oriented System Engineering. SOSE, IEEE, pp. 176–184.

Serhiienko, Oleksii, Spillner, Josef, 2018. Systematic and recomputable comparison of multi-cloud management platforms. In: Proceedings of the 9th International Conference on Cloud Computing Technology and Science (CloudCom). IEEE, pp. 107–114.

Shen, Pengyan, Liu, Wan, Wu, Zheng, Xiao, Mingzhong, Xu, Quanqing, 2017. SpyStorage: A highly reliable multi-cloud storage with secure and anonymous data sharing. In: Proceedings of Th 11th International Conference on Networking, Architecture, and Storage. NAS, IEEE, pp. 1–6.

Sheridan, Craig, Massonet, Philippe, Phee, Andrew, 2017. Deployment-time multi-cloud application security. In: Proceedings of the 3rd International Conference on Smart Computing. SMARTCOMP, IEEE, pp. 1–5.

Shi, Tao, Ma, Hui, Chen, Gang, Hartmann, Sven, 2020. Location-aware and budget-constrained service deployment for composite applications in multi-cloud environment. IEEE Trans. Parallel Distrib. Syst. 31 (8), 1954–1969.

Shin, Dong-Hee, 2013. User centric cloud service model in public sectors: Policy implications of cloud services. Gov. Inf. Q. 30 (2), 194–203.

da Silva, Leandro Costa, De Medeiros, Robson, Rosa, Nelson, 2023. COSTA: A cost-driven solution for migrating applications in multi-cloud environments. In: Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing. pp. 57–63.

Sim, Julius, Wright, Chris C., 2005. The kappa statistic in reliability studies: use, interpretation, and sample size requirements. Phys. Ther. 85 (3), 257–268.

Singh, Saurabh, Jeong, Young-Sik, Park, Jong Hyuk, 2016. A survey on cloud computing security: Issues, threats, and solutions. J. Netw. Comput. Appl. 75, 200–222.

Sitaram, Dinkar, Harwalkar, Sudheendra, Sureka, Chetna, Garg, Harsh, Dinesh, Manusarvathra, Kejriwal, Mayank, Gupta, Shikhar, Kapoor, Vivek, 2018. Orchestration based hybrid or multi clouds and interoperability standardization. In: Proceedings of the 7th International Conference on Cloud Computing in Emerging Markets. CCEM, IEEE, pp. 67–71.

Slominski, Aleksander, Muthusamy, Vinod, Khalaf, Rania, 2015. Building a multi-tenant cloud service from legacy code with docker containers. In: Proceedings of the 3rd International Conference on Cloud Engineering. ICCE, IEEE, pp. 394–396.

Sofia, Rute C, Dykeman, Doug, Urbanetz, Peter, Galal, Akram, Dave, Dushyant Anirudhdhabhai, 2023. Dynamic, context-aware cross-layer orchestration of containerized applications. IEEE Access 11, 93129–93150.

Sofia, Rute C, Salomon, Josh, Ferlin-Reiter, Simone, Garcés-Erice, Luis, Urbanetz, Peter, Mueller, Harald, Touma, Rizkallah, Espinosa, Alejandro, Contreras, Luis M, Theodorou, Vasileios, et al., 2024. A framework for cognitive, decentralized container orchestration. IEEE Access.

Solayman, Haleema Essa, Qasha, Rawaa Putros, 2023. Seamless integration of DevOps tools for provisioning automation of the IoT application on multi-infrastructures. In: Proceedings of the 3rd International Conference on Intelligent Communication and Computational Techniques. ICCT, IEEE, pp. 1–7.

Struhár, Václav, Craciunas, Silviu S, Ashjaei, Mohammad, Behnam, Moris, Papadopoulos, Alessandro V, 2024. Hierarchical resource orchestration framework for real-time containers. ACM Trans. Embed. Comput. Syst. 23 (1), 1–24.

Sultan, Sari, Ahmad, Imtiaz, Dimitriou, Tassos, 2019. Container security: Issues, challenges, and the road ahead. IEEE Access 7, 52976–52996.

Svantesson, Dan Jerker B., Clarke, Roger, 2010. Privacy and consumer risks in cloud computing. Comput. Law Secur. Rev. 26 (4), 391–397.

Tang, Xiaoyong, 2021. Reliability-aware cost-efficient scientific workflows scheduling strategy on multi-cloud systems. IEEE Trans. Cloud Comput. 10 (4), 2909–2919.

Tiwari, Pankaj, Sharma, Sangeeta, 2023. Automation of faas serverless frameworks openfaas and openwhisk in private cloud. In: Proceedings of the World Conference on Communication & Computing. WCONF, IEEE, pp. 1–11.

Tricomi, Giuseppe, Panarello, Alfonso, Merlino, Giovanni, Longo, Francesco, Bruneo, Dario, Puliafito, Antonio, 2017. Orchestrated multi-cloud application deployment in OpenStack with TOSCA. In: Proceedings of the 3rd International Conference on Smart Computing. SMARTCOMP, IEEE, pp. 1–6.

Trihinas, Demetris, Pallis, George, Dikaiakos, Marios D., 2015. Monitoring elastically adaptive multi-cloud services. IEEE Trans. Cloud Comput. 6 (3), 800–814.

Tundo, Alessandro, Mobilio, Marco, Riganelli, Oliviero, Mariani, Leonardo, 2024. Monitoring probe deployment patterns for cloud-native applications: Definition and empirical assessment. IEEE Trans. Serv. Comput..

Vaughn, Nigel, 2020. Docker: Up & running: Shipping reliable containers in production. O'Reilly Media.

Verdugo, Pedro, Salvachiua, Joaquín, Huecas, Gabriel, 2017. An agile container-based approach to Taas. In: Proceedings of the 56th FITCE Congress. IEEE, pp. 10–15.

Wahab, Omar Abdel, Bentahar, Jamal, Otrok, Hadi, Mourad, Azzam, 2016. Towards trustworthy multi-cloud services communities: A trust-based hedonic coalitional game. IEEE Trans. Serv. Comput. 11 (1), 184–201.

Wang, Long, Ramasamy, Harigovind, Salapura, Valentina, Arnold, Robin, Wang, Xu, Bakthavachalam, Senthil, Coulthard, Phil, Suprenant, Lee, Timm, John, Ricard, Denis, et al., 2019. System restore in a multi-cloud data pipeline platform. In: Proceedings of the 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks. ICDSN, IEEE, pp. 21–24.

Wang, Yu, Sun, Yi, Lin, Zhaowen, Min, Jiangsong, 2020. Container-based performance isolation for multi-tenant saas applications in micro-service architecture. In: Journal of Physics: Conference Series. Vol. 1486, 052032, (5).

Wang, Yulong, Wang, Qixu, Qin, Xue, Chen, Xingshu, Xin, Bangzhou, Yang, Run, 2023. DockerWatch: a two-phase hybrid detection of malware using various static features in container cloud. Soft Comput. 27 (2), 1015–1031.

Waseem, Muhammad, Ahmad, Aakash, Liang, Peng, Akbar, Muhammad Azeem, Khan, Arif Ali, Ahmad, Iftikhar, Setälä, Manu, Mikkonen, Tommi, 2025. Dataset for the paper: Containerization in multi-cloud environment: Roles, strategies, challenges, and solutions for effective implementation. https://zenodo.org/record/10732611.

Watada, Junzo, Roy, Arunava, Kadikar, Ruturaj, Pham, Hoang, Xu, Bing, 2019. Emerging trends, techniques and open issues of containerization: A review. IEEE Access 7, 152443–152472.

Watanabe, Hiroki, Yasumori, Ryo, Kondo, Takao, Kumakura, Ken, Maesako, Keisuke, Zhang, Liang, Inagaki, Yusuke, Teraoka, Fumio, 2022. Contmec: An architecture of multi-access edge computing for offloading container-based mobile applications. In: Proceedings of the 35th International Conference on Communications ICC). IEEE, pp. 3647–3653.

Wohlin, Claes, 2014. 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, pp. 1–10.

Wohlin, Claes, Runeson, Per, Höst, Martin, Ohlsson, Magnus C, Regnell, Björn, Wesslén, Anders, 2012. Experimentation in Software Engineering. Springer Science & Business Media.

Wong, Ann Yi, Chekole, Eyasu Getahun, Ochoa, Martín, Zhou, Jianying, 2023. On the security of containers: Threat modeling, attack analysis, and mitigation strategies. Comput. Secur. 128, 103140.

Wong, Walter, Zavodovski, Aleksandr, Zhou, Pengyuan, Kangasharju, Jussi, 2019. Container deployment strategy for edge networking. In: Proceedings of the 4th Workshop on Middleware for Edge Clouds & Cloudlets. MECC, ACM, pp. 1–6.

Wurster, Michael, Breitenbücher, Uwe, Képes, Kálmán, Leymann, Frank, Yussupov, Vladimir, 2018. Modeling and automated deployment of serverless applications using TOSCA. In: Proceedings of the 11th IEEE Conference on Service-Oriented Computing and Applications. SOCA, IEEE, pp. 73–80.

Zeck, Adam, Bouroudjian, Jack, 2017. Real-world experience with a multicloud exchange. IEEE Cloud Comput. 4 (4), 6–11.

Zhamak, Dehghani, 2018. Data mesh: Towards data-driven architecture. https://martinfowler.com/articles/data-mesh-principles.html.

Zhang, Shuai, Ni, Lin, Han, Kun, 2021. A service driver based application execution and development method in multi-cloud context. In: Proceedings of the 1st International Conference on Data Science and Computer Application. ICDSCA, IEEE, pp. 33–37.

Zhong, Zhiheng, Xu, Minxian, Rodriguez, Maria Alejandra, Xu, Chengzhong, Buyya, Rajkumar, 2022. Machine learning-based orchestration of containers: A taxonomy and future directions. ACM Comput. Surv. 54 (10s), 1–35.

Zhou, Xin, Jin, Yuqin, Zhang, He, Li, Shanshan, Huang, Xin, 2016. A map of threats to validity of systematic literature reviews in software engineering. In: Proceedings of the 23rd Asia-Pacific Software Engineering Conference. APSEC, IEEE, pp. 153–160.

Zhou, Naweiluo, Zhou, Huan, Hoppe, Dennis, 2022. Containerization for high performance computing systems: Survey and prospects. IEEE Trans. Softw. Eng. 49 (4), 2722–2740.