Foundations and Trends® in Networking

Resilience in Edge Computing: Challenges and Concepts

Suggested Citation: Doğanalp Ergenç, Agon Memedi, Mathias Fischer and Falko Dressler (2025), "Resilience in Edge Computing: Challenges and Concepts", Foundations and Trends® in Networking: Vol. 14, No. 4, pp 254–340. DOI: 10.1561/1300000074.

Doğanalp Ergenç
TU Berlin
doganalp.ergenc@tu-berlin.de

Agon Memedi
TU Berlin
agon.memedi@tu-berlin.de

Mathias Fischer
University of Hamburg
mathias.fischer@uni-hamburg.de

Falko Dressler
TU Berlin
falko.dressler@tu-berlin.de

now
the essence of knowledge
Boston — Delft

Contents

Resilience in Edge Computing: Challenges and Concepts

Doğanalp Ergenç[1], Agon Memedi[1], Mathias Fischer[2] and Falko Dressler[1]

[1] *Technical University of Berlin, Germany; doganalp.ergenc@tu-berlin.de, agon.memedi@tu-berlin.de, falko.dressler@tu-berlin.de*
[2] *University of Hamburg, Germany; mathias.fischer@uni-hamburg.de*

ABSTRACT

Edge computing has evolved significantly from early research ideas to modern 5G mobile and multi-access edge computing (MEC). In many 6G-related projects, we see a clear trend toward virtualizing computing resources at the edge. Motivated by the cloud-edge-continuum that is the basis for next-generation metaverse applications, and the need for low-latency solutions, distributed computing is now receiving even more attention. A final hurdle for the wide use of (virtualized) edge computing for mission-critical applications is resilience. In this context, resilience is the ability of modern communication and computation systems to deal with unknown and unforeseen events, both from internal and external sources. Thus, making MEC resilient to outages (e.g., system failures or energy outages due to natural disasters), security incidents (e.g., the use of intelligent jamming or malicious users), and overall challenging conditions (e.g., high mobility or impaired connectivity) is of the highest importance. In this monograph, we review the current

Doğanalp Ergenç, Agon Memedi, Mathias Fischer and Falko Dressler (2025), "Resilience in Edge Computing: Challenges and Concepts", Foundations and Trends® in Networking: Vol. 14, No. 4, pp 254–340. DOI: 10.1561/1300000074.

state-of-the-art of resilience in mobile edge computing. We explore MEC-specific challenges and resilience objectives, and discuss selected resilience measures. We trust that this monograph will be an invaluable resource for beginners and experts in the field as a compound resource on resilience in MEC.

1

Introduction

Computing paradigms have continuously evolved to address the growing demands of modern applications. Initially, cloud computing emerged as a transformative approach, enabling the offloading of computationally intensive tasks to centralized and virtualized infrastructures. These infrastructures provide resource scalability and flexibility, and thus can adapt to the diverse applications with varying requirements. However, the centralized nature of cloud computing resources induce additional communication delay and cause network overhead. This is especially problematic for time-sensitive and mission-critical applications. Eventually, edge computing is introduced as a complementary paradigm, bringing computational resources closer to end-users and devices [42]. This proximity benefits time-critical applications by reducing delays and improving responsiveness. It also offers better privacy by processing sensitive data locally on devices or nearby servers, reducing the risk of exposure during transmission to centralized cloud systems.

As a natural progression, multi-access edge computing (MEC) has emerged to address the unique requirements of highly dynamic environments. Unlike traditional edge computing, MEC is tailored to the mobility and connectivity constraints of devices that cannot rely on

consistent connection with centralized entities [76]. For instance, modern connected vehicles must process vast amounts of sensory data for tasks like obstacle detection, navigation, and collision avoidance. MEC can offload these computationally intensive tasks to nearby edge servers, reducing the processing burden on individual vehicles and enabling faster response times [21], [54]. Moreover, internet of things (IoT) applications in smart cities, healthcare, and industrial automation, all orchestrated with mobile sensors, require real-time monitoring and analysis, and thus necessitate MEC solutions close to the data sources [43], [47], [58], [84]. Immersive applications like the Metaverse, using virtual and extended reality (VR/XR) technologies, take place in large-scale events, training simulations, digital twins, and collaborative mobile environments [48], [100], [102]. These applications generate massive amounts of data, requiring artificial intelligence (AI)-empowered processing at the edge due to the limited computational capacity of user devices. These diverse application requirements make it challenging to provision computational resources dynamically, and they remain a persistent hurdle for MEC systems.

Apart from the variety of MEC applications, the heterogeneity of MEC resources is also rapidly increasing, adding further complexity to MEC ecosystems. Initially, these resources were dedicated virtualized servers managed centrally by edge or cloud controllers. However, advancements such as 5G-enabled computation at the network edge have pushed these resources closer to applications, making them an integral part of the communication infrastructure, such as 5G base stations [98]. The evolution has also introduced mobile entities, like connected vehicles, which can act as both consumers and providers of computational resources. This dual role introduces additional layers of complexity, as computing resources are now not only heterogeneous but also mobile. Emerging concepts like virtual edge computing (V-Edge) are further decentralizing computation by enabling ad-hoc resource aggregation from diverse nodes like modern cars with advanced computational capabilities [20]. Ensuring interoperability across this heterogeneous landscape is an ongoing challenge.

In the light of this complexity, resilience has become a critical concern for MEC systems [7], [83]. It can be defined as *"the ability*

(of the network) to provide and maintain an acceptable level of service in the face of various faults and challenges to normal operation" [85]. As an example, for critical applications, computational results must be reliable and accurate despite potential failures in virtualized edge resources, e.g., as a result of system overloads, operational errors, or connectivity loss. Since the MEC ecosystem becomes more diverse, a multitude of security threats can be found at different MEC actors and components, such as (malicious) end hosts, (compromised) virtualized environments, and communication infrastructure and protocols. Privacy is another concern, particularly for offloaded tasks that involve sensitive user or application data [15], [88]. As a result, *resilience objectives* such as availability, reliability, security, and privacy must be carefully considered in the design of MEC systems.

Given the inherent complexity of MEC systems and their diverse resilience objectives, it is crucial to develop effective resilience measures. These must address potential faults and attack vectors while also overcoming the unique challenges to which MEC systems are exposed. Accordingly, this monograph provides a systematic analysis of potential resilience measures designed to fulfill the selected resilience objectives. Our contributions can be summarized as follows:

- We first present an overview of a heterogeneous MEC system model, analyzing key characteristics of different components within the MEC ecosystem. We also associate this model with an existing reference MEC architecture to align our analysis with the literature and standardization efforts.

- Second, we identify the primary challenges that limit and also necessitate the development of resilience measures. We also present the main resilience objectives (dependability and trustworthiness) and techniques (proactive and reactive) that are aimed at and employed in common by several resilience measures.

- We introduce advanced resilience concepts, linking them to MEC components and addressing identified challenges. This analysis is based on a literature review, each article providing essential building blocks for implementing comprehensive resilience measures.

Following these contributions, the methodological overview of our analysis is also shown in Figure 1.1. The rest of the work is organized as follows: Section 2 reviews related work that addresses resilience in the context of MEC. Section 3 introduces a system model for a comprehensive MEC ecosystem, encompassing various types of resources, users, and interfaces. Section 4 discusses the challenges specific to MEC that influence the design of resilient systems and highlights key resilience objectives considered in our analysis. Section 5 categorizes our resilience concepts by analyzing several selected studies from the literature and underlining their relevance with the presented resilience challenges and objectives. Section 6 outlines potential future directions, and Section 7 concludes the monograph.

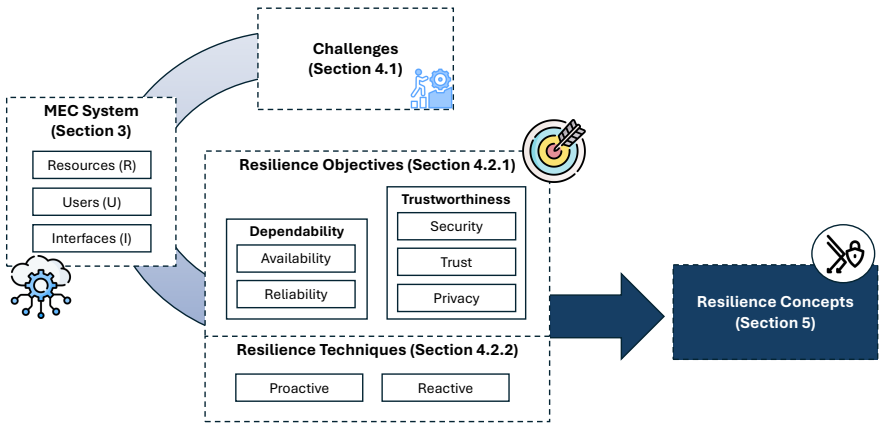


Figure 1.1: Overview of our analysis of challenges, objectives, and concepts of resilience in MEC.

2

Related Work

In recent years, mobile edge computing (MEC), and distributed computing paradigms more generally, have attracted significant attention from the research community due to their potential to meet the low-latency, scalability, and computational demands of emerging applications with stringent resource requirements. Researchers have extensively explored critical aspects of MEC, such as efficient resource allocation and utilization [115], energy efficiency [38], and task scheduling [58]. Among these, resilience has emerged as an essential topic, addressing the need for robust operation under dynamic environments, hardware and software failures, and security threats. Multiple surveys and review articles have investigated resilience alongside other topics, each employing varying scopes, methodologies, and focuses.

In the literature, resilience and security in edge computing are often discussed in surveys with a broader and holistic framework on distributed computing, treating edge computing as a specific case of distributed computing paradigms [61], [73], [83]. In other cases, resilience and security are briefly addressed in surveys that discuss edge computing in general, including its architecture, resource and energy efficiency, latency optimization, etc., but not specifically focusing on resilience and security

[44], [63], [101]. Additionally, there are works that focus exclusively on edge computing environments, addressing threats, challenges, and mechanisms unique to the edge context [4], [23], [76], [93], [107], [110].

In the following, we provide a summary of related work on resilience in edge computing – focusing on selected survey papers. We divide our related work study in two categories: First, presenting relevant surveys on resilience and security in cloud-based distributed computing, and secondly, presenting surveys with a stronger or resilience in the specific context of edge computing. Although this overview is not exhaustive, it represents a selected subset of studies that complement our work while also highlighting their respective unique contributions.

2.1 Resilience in Cloud-based Distributed Computing

Works that adopt a holistic perspective on resilience across different distributed computing paradigms treat edge computing as an extension of the cloud architecture, often together with fog computing. These surveys explore the shared resilience issues inherent in distributed systems while also addressing the unique challenges each paradigm presents. By examining these aspects across a range of paradigms, they provide a broader understanding of the threats and mechanisms needed to ensure the resilience and security of distributed infrastructures.

Shirazi *et al.* [83] provide a comprehensive survey that focuses on resilience in edge and fog computing, as two paradigms that extend traditional cloud computing. Using the ETSI MEC reference architecture (see Section 3.2) as a basis for edge computing, they identify distinct security and resilience requirements for both edge and fog computing. Their work emphasizes the need for tailored security mechanisms to ensure resilience in these decentralized environments. The paper also introduces resilience strategies such as the D2R2+DR framework (Defend, Detect, Remediate, Recover, Diagnose, and Refine), which is mapped to the ETSI MEC architecture to improve reliability.

While the primary focus of [83] is on resilience, Roman *et al.* [73] put a strong emphasis on security, still following a holistic model of different computing paradigms that bring cloud-like capabilities to the network edge (fog computing, mobile edge computing (MEC), and mo-

bile cloud computing). This survey offers a comprehensive analysis of security threats, challenges, and corresponding security mechanisms across various edge computing paradigms. It also examines the differences in security requirements for different edge paradigms, taking into account their unique characteristics, and identifies shared vulnerabilities that could be addressed through cross-paradigm solutions. Roman *et al.* [73] also discuss a range of security mechanisms applicable across edge paradigms, as well as the integration of solutions from other fields like cloud computing and grid computing.

Maciel *et al.* [61] integrate multiple paradigms, analyzing edge, fog, and cloud computing. Unlike broader surveys that focus more generally on security and resilience, this work specializes in reliability and availability metrics, with a particular focus on addressing delay-sensitive and context-aware challenges in IoT environments. It emphasizes the need for hierarchical edge–fog–cloud architecture to improve system performance and dependability for delay-sensitive tasks.

The articles discussed so far offer a general overview of resilience and security in cloud-based distributed computing, often examining these aspects in relation to architectural distinctions between cloud, fog, and edge computing. However, their broader scope limits their ability to provide a detailed overview of resilience concepts and measures specifically tailored to MEC.

2.2 Security in Edge Computing

Compared to surveys that address holistic security architectures spanning the different cloud computing layers, several studies specifically target the unique challenges and requirements of edge computing environments. However, despite its critical importance, resilience as a concept within the MEC context is often overlooked. Instead, many surveys focus on areas such as energy efficiency [38], resource optimization through task offloading [58], [115], latency minimization [40], [46], and security and privacy, often narrowing their scope to specific use cases. For example, some studies investigate security and privacy mechanisms tailored to edge-assisted IoT systems [4], [23], while others emphasize the role of AI in mitigating security threats in MEC environments [93].

Xiao *et al.* [107] present a survey on security threats and corresponding defense mechanisms in edge computing, focusing on four practically relevant attack types: distributed denial-of-service (DDoS), side-channel, malware injection, and authentication and authorization attacks. By comparing edge computing security challenges with those in cloud computing, the survey emphasizes how decentralization, limited resources, and multi-actor environments worsen security in edge systems. The authors identify root causes such as design flaws, misconfigurations, and inadequate access control, while proposing detection- and prevention-based defense mechanisms.

Zeyu *et al.* [110] present a detailed review of edge computing security, focusing on research efforts in five key domains: access control, key management, privacy protection, attack mitigation, and anomaly detection. The survey identifies critical security challenges that arise from the decentralized and resource-constrained nature of edge computing, the integration of emerging technologies, and the increasing demands for privacy. Each research area is thoroughly reviewed, highlighting current advancements and limitations in addressing these challenges. The paper emphasizes the complexities of managing decentralized access, ensuring robust privacy mechanisms, mitigating sophisticated cyber-attacks, and developing effective anomaly detection techniques. The survey suggests future research should focus on scalable and adaptive security solutions, interdisciplinary approaches, and holistic frameworks to enhance edge computing resilience and privacy protection.

Alwarafy *et al.* [4] conduct a survey on security and privacy in edge computing-assisted IoT systems (EC-IoT), emphasizing the unique vulnerabilities and threats in these environments. The paper provides a detailed classification of attacks based on type, security objectives, and network layers, offering an in-depth exploration of specific threats and countermeasures for EC-IoT integration. Compared to broader surveys on edge computing, this work specializes in IoT contexts, highlighting the interplay between IoT-specific challenges and EC capabilities. The survey also identifies open research questions and future directions to enhance security and privacy of EC-IoT systems.

Wang *et al.* [93] present a comprehensive survey on security and privacy in multi-access edge computing, focusing on using AI to ad-

dress complex and evolving threats. Using the ETSI MEC reference architecture as a framework, and extending it with software-defined networking (SDN) and network functions virtualization (NFV), the survey examines AI-driven solutions for specific security challenges. It highlights the trade-offs between real-time communication demands and robust security in time-sensitive MEC scenarios and outlines future research opportunities emphasizing the potential of AI in securing MEC environments. Although not a survey, the white paper by Sabella *et al.* [75] provide a detailed overview of security challenges in MEC, focusing exclusively on the ETSI MEC architecture emphasizing standardization, regulation, and collaboration between stakeholders, which are less prominent in other surveys that focus on technical attacks or specific architectures.

The narrow focus of the aforementioned surveys on security is helpful for understanding the unique security challenges, including attacks and potential measures against them in the context of edge computing. However, this omits a broader discussion on other resilience objectives, such as dependability and trustworthiness.

2.3 Resilience in Edge Computing: The Need for a Holistic View

In the current body of literature, we have identified a noticeable gap between surveys adopting a broad and holistic perspective on resilience across various distributed computing paradigms and those focusing specifically on security within edge computing environments. Therefore, in this monograph, we address this gap in the literature by using the concept of resilience to systematically describe open requirements, existing solutions, and unresolved research challenges in resilient edge computing.

Unlike existing surveys and reviews that categorize approaches based on specific resilience objectives, such as reliability or security, our primary aim is to derive overarching resilience concepts and measures. These include the general trends observed in the literature toward designing robust and resilient MEC systems. By adopting this approach, we aim to provide a more comprehensive view of the challenges and solutions in the field, offering a clearer understanding of the broader research landscape.

Additionally, instead of comparing the state-of-the-art computing paradigms – namely cloud, fog, and edge computing – an approach often taken in other surveys, we focus explicitly on the unique characteristics of MEC. These characteristics necessitate specialized resilience measures or pose additional challenges in their development. This targeted focus allows us to highlight these distinctive challenges and illustrate how the proposed resilience strategies address them effectively.

Finally, rather than simplifying the system model to a standalone MEC server interacting with remote virtualized servers and mobile users, we conceptualize a more comprehensive MEC ecosystem. This ecosystem accounts for the diverse resources, user types, and their interactions. By doing so, we associate the proposed resilience measures with specific MEC actors and components, thereby emphasizing their relevance to distinct aspects of edge computing. This perspective enables a more detailed analysis of resilience requirements and the practical applicability of proposed solutions.

3

System Model and Reference Architecture

In this section, we present a comprehensive MEC system model that emphasizes the heterogeneity of MEC resources, application and user types, and the interfaces connecting MEC systems and users. The diverse characteristics of these actors and components impose different challenges for the design of MEC systems and also necessitate the development of suitable resilience measures. While similar models have been introduced and discussed under various paradigms such as cloudlets and fog computing, we specifically frame these elements within the context of the MEC ecosystem, focusing on key aspects that are directly relevant to our resilience discussions in the following sections.

Additionally, we review the European Telecommunications Standards Institute (ETSI) reference MEC architecture,[1] which is widely regarded within the MEC community as a foundational framework outlining the primary design components. Although it does not fully capture the architectural complexity of a heterogeneous MEC ecosystem, it provides a high-level abstraction of essential components and interfaces. To align with existing efforts that adopt this reference model

[1] ETSI MEC Framework, https://www.etsi.org/deliver/etsi_gs/mec/001_099/003/02.01.01_60/gs_mec003v020101p.pdf.

as the basis for MEC design and analysis, we map the respective parts of our system model to corresponding elements of the ETSI reference architecture. This alignment ensures consistency and enhances the relevance of our discussions in the context of established frameworks.

3.1 System Model

A comprehensive system model is depicted in Figure 3.1, highlighting several key actors and components in the MEC ecosystem that are most relevant to the resilience discussions in the following sections.

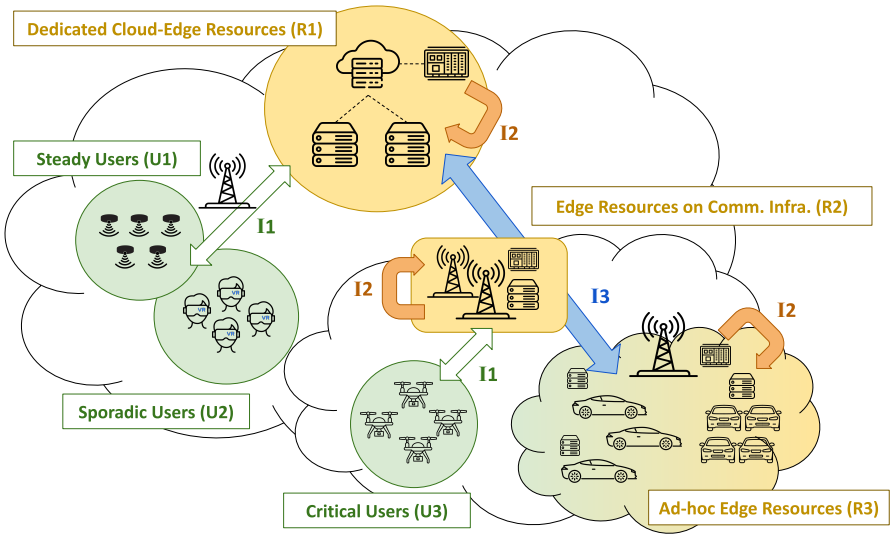


Figure 3.1: System model of an heterogeneous MEC environment.

We categorize these actors and components into three classes: MEC resources (R), users (U), and interfaces (I). MEC resources refer to the core MEC systems with virtualized infrastructure that hosts virtual machines (VMs) and containers and enables the execution of various tasks and applications. The resources also include MEC controllers and managers responsible for orchestrating and monitoring the systems within and across different MEC providers. MEC users represent a diverse set of applications with different resource and latency require-

ments, and criticality. Finally, MEC interfaces encompass intra- and inter-component connections that facilitate communication and interoperability between the components in the ecosystem. Table 3.1 provides a summary of the types of MEC resources, users, and interfaces presented.

Table 3.1: Description of the main types of actors and components in an MEC environment.

MEC Components		Description
Resources	R1	Dedicated and stationary MEC systems at established computation and data servers.
	R2	Stationary MEC systems integrated to the communication infrastructure.
	R3	Ad-hoc and mobile MEC systems with opportunistic computational resources.
Users	U1	Steady users with consistent and predictable requirements.
	U2	Temporary and mobile users with changing application demands.
	U3	Highly mobile users with time-critical applications.
Interfaces	I1	Interfaces between MEC users and resources.
	I2	Internal management interfaces between and within MEC systems and controllers.
	I3	Inter-host and inter-controller interfaces between different MEC resources.

3.1.1 Resources

In our system model, we group different MEC resources in the following hierarchy: dedicated cloud-edge resources (R1), edge resources integrated to the communication infrastructure (R2), and ad-hoc or virtual edge resources (R3).

Dedicated cloud-edge resources (R1) represent the cloud-edge continuum, consisting of *dedicated MEC systems* [9], [24]. These resources, often deployed at centralized or edge-specific facilities in the form of server farms or computing clusters, provide the backbone for many mobile edge applications. R1 resources ensure high availability and scalability, making them suitable for large-scale applications. Their centralized nature eases configuration and maintenance. However, their physical distance from users may result in higher latency compared to other resource types.

Edge resources on communication infrastructure (R2) bring MEC resources closer to the users by *integrating them into the communication infrastructure*, such as cellular base stations or small cell towers [68]. As such, this is a logical extension of the cloud-edge continuum. The proximity reduces latency and data forwarding overhead. However, the physical accessibility of R2 resources introduces additional vulnerabilities, as these infrastructure points are more exposed to physical attacks or tampering compared to the isolated servers of R1. Besides, typically, a large number of these resources are distributed in wide areas, which makes their management and orchestration more challenging. Despite being less powerful than R1, R2 resources play a critical role in delivering fast responses for MEC applications, especially in urban or densely populated areas.

Ad-hoc (or opportunistic) edge resources (R3) represent *a dynamic and mobile resource pool*, in which mobile servers and users can be utilized as MEC systems on-demand. One example is autonomous vehicles (e.g., vehicular edge hosts) that can share their excess computational power with other users in the MEC ecosystem and take part as temporary MEC systems [20]. Here, they can often function as both provider and consumers of edge resources, adding further complexity to resource orchestration. Another example is UAV-based MEC systems that leverage the mobility of aerial vehicles to quickly deploy and transfer edge resources when needed [8]. Despite their flexibility, the mobility and variability of R3 resources pose challenges for resource management, requiring robust mechanisms to ensure service continuity and fairness in such fluctuating resource environments. In most cases, R3 resources will be inherently heterogeneous, differing in computational capabilities and roles.

For all MEC systems, controllers (whether centralized or decentralized) are critical for resource coordination. Furthermore, collaboration among different MEC systems expands the resource pool, necessitating efficient inter-controller communication. In ad-hoc environments like R3, MEC systems may also serve users directly in a peer-to-peer manner.

3.1.2 Users

Among the various types of MEC users, based on their persistence, mobility, and application characteristics, we identify three primary categories: steady users (U1), sporadic users (U2), and critical users (U3):

Steady users (U1) have *consistent and predictable requirements*, such as connected IoT devices deployed on public transportation systems or smart city sensors and actors, which continuously offload data for analysis to R1 resources. These systems typically operate with predefined QoS requirements and benefit from the reliability and scalability of dedicated MEC resources.

Sporadic users (U2) represent applications that require MEC resources *temporarily* and involve *dynamic mobility*. Examples include augmented reality (AR) applications used at public events such as games or festivals, where participants receive real-time directions, updates, or event overlays on their devices. Since the number of users can fluctuate dynamically, these applications require efficient registration and resource allocation processes through MEC controllers and across different MEC systems.

Critical users (U3) encompass users running *highly mobile, latency-sensitive, and safety-critical* applications. Examples include autonomous vehicle coordination during high-speed road travel or UAV swarm control and coordination. These applications have strict deadlines for computational tasks, such as obstacle detection or route planning, and demand reliable, low-latency connections to MEC systems. Failures in communication or computation can lead to severe consequences, such as accidents or mission failure.

Each MEC user type faces distinct challenges and vulnerabilities. For instance, while U1 users benefit from pre-established authentication and authorization mechanisms, the dynamic nature of U2 users increases the risk of unauthorized or malicious actors infiltrating the system. For U3, even minor disruptions of the communication with MEC systems can lead to intolerable delays, making resilience a critical factor for this category.

3.1.3 Interfaces

We represent the functional connections and communication in the MEC ecosystem through user interfaces (I1), internal MEC interfaces (I2), and inter-MEC interfaces (I3) as shown in Figure 3.1.

User interfaces (I1) represent the interaction between MEC users and systems, encompassing both control data (e.g., service requests, QoS configurations) and application data (e.g., sensory input, task results). These interactions are typically facilitated through wireless communication technologies such as 5G/6G and Wi-Fi. As the communication infrastructure is often managed independently of MEC providers, any impairments in wireless links or disconnections cannot be directly mitigated by MEC providers but still have a significant impact on the I1 interface. Consequently, the allocation of MEC resources to users must carefully account for the reliability of the I1 interface, ensuring that the connectivity provided is sufficient to meet the service requirements of the respective users. For MEC resources that combine computation and communication infrastructure, i.e., R2, the communication links can also be optimally configured. We will discuss this in more detail in Section 5.3.

Internal MEC interfaces (I2) enable the management and orchestration processes between MEC controllers and systems through the internal management modules of MEC systems. These interfaces oversee fundamental operations such as the installation and migration of VMs and containers. To ensure operational integrity, monitoring and anomaly detection mechanisms are typically employed to continuously observe the I2 interface for potential issues or irregularities.

Inter-MEC interfaces (I3) extend I2 to enable collaboration between multiple MEC systems. This interface supports resource sharing across different MEC systems and domains, such as coordinating workloads between R1 resources at a data center and R2 nodes in a metropolitan area. Alternatively, it can manage hierarchical aggregation of diverse resource types, such as integrating R3 resources from a fleet of vehicles with existing R2 infrastructure for large-scale disaster response coordination. This is also relevant to handle handovers between MEC systems, which may require inter-MEC synchronization.

3.2 ETSI Reference Model

The ETSI reference model provides an abstract architecture outlining the fundamental modules and components within MEC systems and controllers. This can be considered as a *zoom-in* view of the resources introduced in Section 3.1.1, comprising the internals of MEC systems. Figure 3.2 illustrates an adaptation of this model, highlighting *only* the relevant modules for our subsequent discussions.

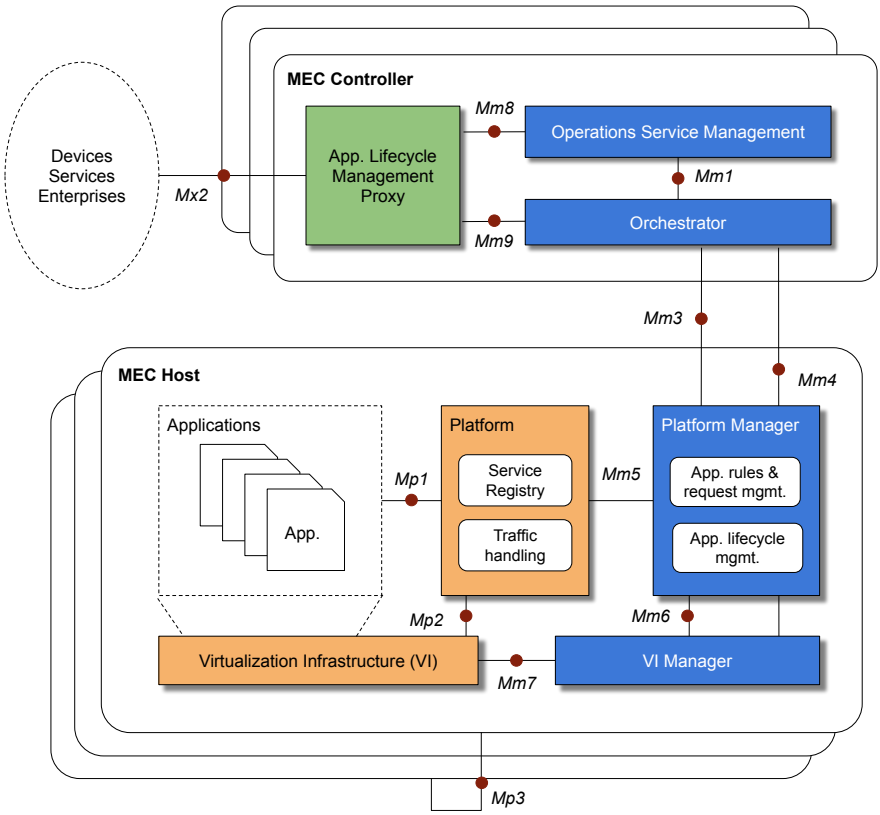


Figure 3.2: MEC reference architecture adapted from the ETSI framework.

3.2.1 Components

The ETSI architecture contains two main components: MEC system and MEC controller,[2] which together correspond to a MEC resource in Section 3.1. The **MEC system** consists primarily of application modules (orange): the *platform* and the *virtualization infrastructure (VI)*, which supply compute, storage, and network resources to MEC users. The platform provides essential functionalities such as service registry, traffic steering, and policy enforcement for running user applications on an MEC system. It also enables communication with other MEC systems via control plane interfaces, forming a communication grid that corresponds to the I3 interface described in Section 3.1.3. The *virtualization infrastructure* delivers virtualization and containerization technologies and handles data traffic routing between applications, services, and networks according to the platform's forwarding rules.

On top of the VI, *applications* operate on VMs or containers within the MEC system. They interact with the platform to consume or provide services. The platform ensures that applications' resource and performance requirements are met and supports application relocation during events like user handovers.

Within the MEC system, there are also management modules (blue), the *platform manager (PM)*, and the *VI manager (VIM)*. The PM oversees MEC system management and orchestration. It handles application life-cycle operations, such as instantiation and termination, and enforces traffic rules through continuous communication, configuration, and monitoring of the platform. The VIM manages the MEC system's virtualized compute, storage, and network resources, functioning as a hypervisor. It allocates virtual instances, monitors performance and faults, stores application images for rapid deployment, and maintains resource availability information.

As the central entity, the **MEC controller** manages resources and applications across the entire edge network, potentially encompassing multiple MEC systems. It has visibility over all MEC systems, ser-

[2] Our naming convention is slightly different compated to the original proposal. We simplified some names and terms for a better alignment with the rest of our discussions.

vices, and resources; and acts as an *orchestrator*, ensuring application requirements are met. The controller oversees application deployment, integrity checks, relocation, and maintains a catalog of available applications. It collaborates with the PM and VIM on each host to allocate resources and monitor their availability. The MEC controller includes two additional modules: *the application life-cycle management proxy* and *the operations service management (OSM)*. Together, they handle and authorize user requests for application onboarding, instantiation, termination, integrity checks, and relocation between MEC systems. The proxy is depicted in a distinct color (green) in Figure 3.2, as it is the only module offering direct user access, and thus act as a front-end.

3.2.2 Reference Points

The ETSI reference architecture specifies several reference points to represent interconnections between different modules. Within the MEC system, there exist reference points like *Mp1, Mp2, Mm5, Mm6,* and *Mm7* which define interfaces between the platform, VI, and their managers. These typically function in a closed-loop system, such as an operating system (OS) with a hypervisor, and align with the I2 interface described in Section 3.1.3. Reference points *Mm3* and *Mm4* link controllers to hosts. In dedicated MEC systems (e.g., R1 in Section 3.1.1), where MEC controllers and hosts are co-located, these interfaces can achieve high reliability and are considered part of I2. In distributed and heterogeneous MEC systems (e.g., R2 and R3 in Section 3.1.1), multiple controllers and hosts must coordinate remotely, introducing diverse interoperability requirements. Their effectiveness depends on the underlying networking technologies. Reference point *Mp3* connects different MEC systems and can extend to remote hosts. Consequently, it is exposed to threats and challenges similar to those that affect the I3 interface. Although not part of the reference architecture, multiple controllers may require a dedicated interface for collaborative resource management in some scenarios.

The reference architecture does not extensively cover interfaces between users and the MEC system. Reference point *Mx2* represents the control interface through which users submit application requests

to the MEC controller. Afterward, users typically communicate directly with MEC systems, often over wireless networking technologies. These interactions correspond to the I1 interface described in Section 3.1.3.

4

Resilience Challenges and Objectives

MEC environments are vulnerable to a wide range of operational errors, communication bottlenecks, and other challenges arising from their mobile, heterogeneous, and distributed nature with a multitude of actors and components. These challenges are often interdependent, making it difficult to isolate their impacts or rely on a single solution to ensure resilience. For instance, to cope with application failures, traditional primary/backup (PB) redundancy techniques have been effective in cloud environments. However, their utilization in MEC context is not straightforward [66], [97].

One key challenge is the resource limitations of MEC systems compared to cloud servers in data centers. This constraint necessitates installing application instances across geographically distributed MEC systems. Additionally, dynamic and unpredictable networking conditions can hinder mobile users from accessing such distributed PB instances seamlessly. This connectivity variability adds another layer of complexity of ensuring reliable service continuity in MEC environments. Consequently, developing effective resilience measures requires addressing these unique challenges and inherent complexity of MEC systems.

This complexity also complicates the analysis of MEC ecosystem in terms of traditional resilience objectives [70], [85]. For instance, to ensure resource availability becomes insufficient unless paired with a reliable computation and communication performance with minimal errors or interruptions – especially for time-critical applications.

Consequently, in Section 4.1, we first present an analysis of the unique challenges in MEC that require the development of resilience measures. Then, we introduce the resilience objectives and techniques that serve as the foundation for analyzing resilience measures in Section 5.

4.1 Challenges

MEC faces a distinct set of challenges mainly stemming from its distributed and heterogeneous architecture, resource-constrained systems, and dynamic operational conditions. Unlike traditional cloud systems, MEC must support a wide range of applications with diverse criticality and performance requirements, while operating under unpredictable demand patterns and high mobility of both users and hosts. Fluctuating connectivity, due to heterogeneous and unstable access technologies, further complicates maintaining seamless service delivery. Resource contention is a common issue due to the limited capacity of MEC systems, making it difficult to accommodate over-provisioning for redundancy.

Additionally, the distributed nature of MEC introduces complexities in management, monitoring, load balancing, and fault tolerance across geographically dispersed nodes. MEC environments are also prone to operational errors, hardware failures, and environmental vulnerabilities, as many edge resources lack the physical protections of centralized cloud data centers. Security and privacy concerns are exacerbated by the presence of malicious users, unreliable hosts, and the absence of robust trust mechanisms.

Accordingly, in the following, we present several challenges that highlight the need for resilience measures tailored to the unique characteristics of MEC systems. We also address them in our analysis and discussions of the potential resilience measures for MEC in the next section.

We focus on the following challenges:

(**C1**) Unpredictable workloads and diverse user requirements

(**C2**) High mobility and churn rates

(**C3**) Unstable connectivity over heterogeneous technologies

(**C4**) Resource constraints, heterogeneity, and contention

(**C5**) Distributed resources and architecture

(**C6**) Operational risks and environmental challenges

(**C7**) Malicious users and untrustworthy MEC systems

(C1) Unpredictable workloads with diverse user requirements

MEC comes with a variety of applications, ranging from latency-sensitive real-time systems (U3) to less critical, computation-intensive tasks (U1, U2). This diversity creates significant challenges in resource allocation, especially when critical applications such as autonomous driving or drone swarms compete with other workloads for MEC resources. These critical applications require uninterrupted service with stringent latency requirements, meaning even brief disruptions can lead to severe consequences, such as traffic accidents or entire system outages. Additionally, demand surges are difficult to predict, especially in scenarios like public events or emergencies, where the number of users and applications can rapidly escalate.

(C2) High mobility and churn rates

MEC environments can experience high churn rates of users and servers due to mobility. On the one hand, MEC users, such as vehicles, drones, or other mobile devices (U3), frequently join and leave the network, requiring constant reallocation of resources and re-establishment of communication links. These users often operate in challenging wireless environments, where connectivity is intermittent and performance fluctuates. On the other hand, mobile MEC systems, like vehicles or

drones acting as resource providers, introduce additional complexity. The transient nature of these servers leads to unpredictable resource availability and requires sophisticated management strategies to ensure continuous service delivery. High churn rates also hinder robust authentication mechanisms, as frequently authenticating users and devices can impose excessive overhead and delay.

(C3) Unstable connectivity over heterogeneous technologies

MEC systems depend on stable and efficient communication between users, hosts, and controllers. However, the reliance on heterogeneous and wireless technologies, such as 5G/6G and Wi-Fi, makes connectivity inherently unreliable (I1) [78]. Factors like signal interference, physical obstructions, and network congestion can cause frequent disruptions, leading to packet loss, latency spikes, or even complete disconnections. These issues are particularly problematic for latency-sensitive applications (U3), where consistent communication is vital. Seamless interoperability between different access technologies is necessary, but difficult to achieve due to different standards and configurations. Without robust solutions for mitigating these connectivity issues, MEC remains vulnerable to performance degradation and service disruptions.

(C4) Resource constraints, heterogeneity, and contention

Compared to cloud environments, MEC systems operate with significantly more constrained resources, making over-provisioning costly and impractical (especially for R2 and R3). These limitations result in resource contention among competing applications, where CPU, memory, or bandwidth are insufficient to meet the demand. The challenge is exacerbated by the diverse nature of MEC systems, which range from powerful servers to lightweight edge devices with limited capacity. This diversity creates disparities in resource availability across the MEC environments, complicating resource orchestration. The unpredictability of demand surges further amplifies the risk of resource contention, requiring MEC systems to implement adaptive and efficient resource management mechanisms.

(C5) Distributed resources and architecture

Unlike centralized cloud data centers, MEC systems are spread across diverse locations, each with varying access latencies. As a result, migrating services between MEC systems for handling failures or load imbalances becomes more complex and time-consuming (I3). Additionally, traditional load-balancing techniques face inefficiencies in a decentralized setting, leading to bottlenecks and under-utilized resources. From a management perspective, coordination of these resources holistically is also challenging due to multi-stakeholder nature. Proxy-based query and orchestration systems, often used to centralize workload information, can become single points of failure, undermining the inherent advantages of the distributed architecture.

(C6) Operational risks and environmental challenges

MEC systems are susceptible to a variety of operational errors and failures stemming from their hardware, software, and deployment environments. For instance, power outages, hardware malfunctions, or environmental factors like extreme weather and other natural disasters can disrupt the functionality of edge nodes. Hosts deployed in unprotected or remote locations are particularly vulnerable to physical damage or tampering. Software-related issues, such as application bugs, misconfigurations, or conflicts, can also lead to system crashes or degraded performance. Furthermore, MEC environments often lack the robust virtualization infrastructure seen in cloud platforms, making them more prone to hypervisor faults and other virtualization failures.

(C7) Malicious users and untrustworthy hosts

The decentralized and heterogeneous nature of MEC environments makes them an attractive target for malicious actors. Multi-tenancy on MEC nodes allows users to share resources, but malicious users can exploit this setup to launch attacks, such as overloading resources, eavesdropping on other tenants, or injecting malicious code. The transient nature of mobile and temporary users makes robust authentication and monitoring challenging. Furthermore, MEC systems, especially those

that are mobile or opportunistically used (R3), may be unreliable or compromised, posing risks to data integrity and system trust.

Privacy concerns also arise from the distributed architecture of MEC, where data is processed closer to users, often bypassing centralized security policies. While this proximity improves latency and performance, it also means sensitive data is handled on nodes with varying levels of trust and security. For example, personal data from IoT devices, such as health trackers or smart home systems, may be processed on a nearby MEC system with insufficient encryption or monitoring. This makes the system vulnerable to unauthorized access, data theft, or even large-scale privacy breaches if a MEC system is compromised.

4.2 Objectives and Techniques

Considering resilience as a system property, MEC systems aim at different resilience objectives depending on their characteristics and operational environments. These include minimizing failure time, ensuring service continuity with graceful degradation, prevention of resource overload, etc., to guarantee resilience against the aforementioned challenges. Achieving these objectives further necessitate embracing different resilience techniques. For instance, proactive techniques help MEC systems to tolerate failures and disruptions by design, and reactive ones improve their adaptability to counteract when any incident happens. Those techniques induce certain trade-offs in terms of cost, efficiency, and latency, and thus should be used complementarily. In this sense, while the resilience objectives specify what we want to achieve when facing the challenges, the resilience techniques are embraced to reach these objectives, considering their trade-offs.

Accordingly, in this section, we present the resilience objectives and techniques relevant to our analysis. We selected these objectives based on our observations from the literature and further categorize them under two main resilience goals – dependability and trustworthiness – to ease our analysis in Section 5. As mentioned, the techniques are also grouped as proactive and reactive ones, highlighting challenge tolerance and adaptability aspects, respectively.

4.2.1 Objectives

We group the MEC resilience objectives under two overarching resilience goals: dependability and trustworthiness. These goals serve as umbrella terms, encapsulating the interdependencies between objectives such as availability, reliability, security, and privacy. Of course, efficiency metrics such as energy consumption, data rate, and latency remain important. We integrate these in the scope of dependability and trustworthiness assuming a certain degree of graceful degradation.

Dependability

In the MEC context, dependability refers to the system's ability to deliver its intended services consistently and correctly, even under challenging conditions [85]. It encompasses two key objectives: availability and reliability. *Availability* in MEC ensures that computational resources, applications, and services remain accessible to users whenever needed, despite dynamic factors such as mobility (C2), fluctuating network conditions (C3), or variations in demand (C1) [58]. Unlike traditional cloud systems, where resources are centralized and relatively static, MEC environments are distributed (C5) and operate close to mobile users, making the assurance of availability more complex. In contrast, *reliability* focuses on ensuring that MEC services perform as expected, with minimal disruptions or errors, even in case of failures or environmental challenges [83]. Reliability is particularly important for time-sensitive and safety-critical applications, such as collaborative driving [59] or drone orchestration [60]. Disruptive events such as MEC system failures (C6) or abrupt connectivity loss must not compromise the continuity of these applications. Instead, MEC systems should employ mechanisms like seamless failover, adaptive resource allocation, and proactive fault recovery to sustain reliable service delivery, under significant resource limitations and high demands (C4). Together, availability and reliability form the core of dependability in MEC, ensuring that resources are not only present but also capable of maintaining the expected level of service under diverse conditions.

Trustworthiness

In MEC systems, trustworthiness encompasses the ability to ensure security, trust, and privacy throughout the system's operations. This is especially critical in environments characterized by distributed architectures (C5), multi-tenancy, and coexistence of trusted and untrusted entities (C7). *Security* in MEC involves protecting the system from malicious activities, such as unauthorized access, data breaches, and DoS attacks [73]. MEC systems face unique security challenges due to their proximity to end users, the physical exposure of hosts, and diverse connectivity interfaces. For instance, edge servers deployed on communication infrastructure are more vulnerable to physical tampering or cyber attacks than well-guarded cloud data centers. *Trust* in MEC refers to the confidence that stakeholders, such as users, providers, and application developers, place in the system to function as expected without malicious intent [99]. Establishing trust is complex in MEC's multi-stakeholder environment (C5), where resources may be shared among entities with varying levels of trustworthiness. For example, ad hoc MEC systems (R3), such as a vehicle acting as a computational provider, must be validated to ensure they are neither compromised nor providing faulty computations. Lastly, *privacy* focuses on protecting sensitive user data processed within MEC environments [93]. MEC systems frequently handle personal and real-time data, such as location information or video feeds, necessitating privacy-preserving mechanisms. Ensuring secure data processing while minimizing exposure to unauthorized parties is particularly challenging, especially when data must be offloaded to other MEC systems or cloud resources for further computation. Collectively, security, trust, and privacy define the trustworthiness of MEC systems, ensuring that users and providers can rely on the system without fearing malicious exploitation, misuse, or the loss of sensitive information.

In Section 5, we analyze various resilience measures through the lenses of dependability and trustworthiness. It is worth noting that, although security is one of the key resilience objectives, it is not the primary focus of this work, as it has been extensively studied in the literature compared to trust and privacy aspects [75], [107], [110].

4.2.2 Techniques

Proactive and reactive resilience techniques play distinct yet complementary roles in achieving resilience goals in MEC. Highlighting these techniques separately is important because they address different MEC challenges, pertain to different MEC components, and introduce specific trade-offs, that are explained in the following sections. Both proactive and reactive resilience techniques can be designed to handle graceful degradation of service quality. In terms of compute capabilities, this is often handled inherently as other MEC systems will be chosen if insufficient capacity remains at currently used ones. Similarly, communication paths will be selected according to minimum needs of applications in order to fulfill the overall application latency requirements.

Proactive

Proactive techniques aim to prevent failures, data breaches, and malicious attempts before they occur by taking preparatory actions. For dependability, this includes pre-allocating and scheduling backup resources, as well as replicating applications, particularly for critical or latency-sensitive applications (U3). Therefore, they encompass fault and disruption tolerance approaches that ensure the availability of MEC systems when facing challenges. Other proactive mechanisms include enforcing secure access control policies on MEC systems, deploying continuous authentication mechanisms (especially for the I1 interface), and verifying the integrity of MEC applications through attestation methods to ensure trustworthiness. Similarly, minimizing data exposure by applying encryption and anonymization techniques, particularly in MEC environments involving multiple stakeholders, serves as an effective privacy-preserving concept.

Reactive

Reactive techniques address incidents after they occur by relying on real-time monitoring (mostly within I2 interface), anomaly detection, and rapid recovery actions, such as restarting applications, migrating tasks, or reallocating resources to maintain availability. Therefore, these

techniques improve the adaptability of MEC systems, when proactive measures are absent or insufficient in case of unexpected or large-scale failures. For trustworthiness, reactive measures may involve runtime anomaly detection mechanisms to identify untrusted or compromised hosts (especially for ad-hoc R3 resources) and isolate them to prevent further harm. For instance, if a MEC system exhibits abnormal behavior, an attestation process can be triggered to verify its integrity and isolate the host to contain the threat. Similarly, anomaly detection algorithms can identify unusual data access patterns, such as an application attempting to access user data without proper permissions, and respond in real time by blocking the process or revoking its privileges.

The trade-off between proactive and reactive methods lies in their resource consumption, response time, and operational complexity. Proactive methods often consume more resources due to pre-allocation and ongoing computations, such as maintaining redundant application instances or continuously encrypting data. In contrast, reactive methods are activated only when anomalies or breaches occur, making them more resource-efficient. However, their response time may be slower, which can adversely affect time-sensitive applications.

5

Resilience Concepts and Measures

In this section, we present eight main resilience measures that address the challenges outlined in Section 4.1 to achieve dependability and trustworthiness in MEC environments. We discuss these measures as overarching resilience concepts that are identified as a result of the review and categorization of various works in the literature. We also briefly analyze the most representative works for each resilience concept regarding the primary resilience goals – dependability or trustworthiness – and the resilience techniques they employ, whether proactive or reactive. Our analysis is summarized in Table 5.1.

Note that we do not provide extensive details on each paper; rather, we highlight their relevant parts and design artifacts to demonstrate their correspondence to the respective resilience concepts. We associate them with specific types of MEC resources, users, and interfaces (Section 3) as well as the relevant challenges they address (Section 4.1) only when such associations are clearly emphasized in the studies. In the following discussion, these associations are often indicated symbolically. For example, if a resilience concept is particularly effective in protecting a specific type of MEC resource, say R3, we mark it as *(R3)* to highlight its relevance after a short description. A brief overview of these concepts is provided as follows:

Table 5.1: Resilience measures and their relevance to MEC components, challenges, and resilience goals and techniques.

Resilience Measure	Related Work	Addressed Challenges							Relevant Components									Resilience Aspects			
									Resources			Users			Interfaces			Goals		Techniques	
		C1	C2	C3	C4	C5	C6	C7	R1	R2	R3	U1	U2	U3	I1	I2	I3	Depend.	Trust.	Proactive	Reactive
Adaptive redundancy and fault-tolerance	Peng *et al.* [69]	✓		✓	✓		✓					✓	✓		✓			✓		✓	
	Wang *et al.* [97]				✓						✓			✓				✓		✓	✓
	Long *et al.* [57]				✓						✓							✓			✓
	Dong *et al.* [19]		✓				✓				✓			✓				✓		✓	✓
	Ghanavati *et al.* [25]	✓					✓			✓	✓							✓		✓	✓
	Cheng *et al.* [11]	✓		✓	✓		✓		✓	✓								✓		✓	
Real-time monitoring and anomaly detection	Park *et al.* [67]						✓		✓							✓		✓			✓
	Wang *et al.* [96]						✓			✓	✓			✓				✓		✓	
	Tuli *et al.* [90]	✓	✓				✓									✓		✓		✓	
	Tuli *et al.* [91]	✓	✓		✓								✓	✓		✓		✓		✓	✓
Joint computation and communication optimization	Dong *et al.* [18]			✓	✓					✓				✓	✓			✓		✓	
	Yang *et al.* [108]			✓	✓					✓				✓	✓			✓		✓	
	Satria *et al.* [80]	✓					✓			✓					✓		✓	✓			✓
	Ahmad *et al.* [1]	✓	✓								✓				✓			✓		✓	✓
Multi-level resource allocation and coordination	Liu *et al.* [55]	✓				✓			✓	✓	✓							✓		✓	
	Sun *et al.* [86]	✓				✓			✓	✓	✓				✓		✓	✓		✓	
	Wang *et al.* [95]		✓			✓				✓	✓			✓			✓	✓		✓	
	Guo *et al.* [27]		✓			✓				✓	✓	✓	✓		✓			✓		✓	
	Tang *et al.* [87]		✓			✓	✓		✓	✓	✓			✓			✓	✓		✓	✓
Middlewares for computation and communication	Wang *et al.* [92]	✓				✓	✓								✓		✓	✓		✓	
	Javed *et al.* [35]			✓		✓	✓							✓	✓			✓		✓	✓
	Harchol *et al.* [29]		✓			✓	✓										✓	✓			✓
	Samanta *et al.* [77]	✓		✓		✓	✓											✓		✓	
Bilateral reputation assessment	Monir *et al.* [64]							✓		✓	✓								✓	✓	
	Heydari *et al.* [31]	✓						✓											✓	✓	
	Wang *et al.* [94]				✓			✓			✓								✓	✓	
	Alioua *et al.* [3]			✓				✓										✓	✓	✓	
	Zhang *et al.* [112]	✓	✓					✓			✓			✓				✓	✓	✓	✓
Cross-domain federation and access control	Jia *et al.* [37]		✓					✓				✓	✓	✓	✓				✓	✓	
	He *et al.* [30]		✓					✓					✓	✓		✓	✓		✓	✓	
	Liu *et al.* [56]					✓		✓	✓	✓							✓		✓	✓	
	Liu *et al.* [53]		✓					✓		✓	✓			✓					✓	✓	✓
Privacy-preserving task allocation and management	Wu *et al.* [105]				✓			✓			✓								✓	✓	
	Wang *et al.* [103]		✓					✓		✓									✓	✓	
	Cui *et al.* [13]		✓					✓		✓									✓	✓	
	Zhang *et al.* [113]		✓	✓				✓					✓		✓				✓	✓	✓

1. *Adaptive redundancy and fault tolerance:* These measures efficiently leverage virtualization and containerization technologies to ensure maximum uptime in case of failures, addressing dynamic user demands and resource constraints by selectively deploying and migrating application replicas.

2. *Real-time monitoring and anomaly detection:* These mechanisms are crucial for reactive approaches, as they continuously track specific system variables to identify unexpected changes in MEC environments and trigger measures, posing unique engineering challenges for seamless integration.

3. *Joint computation and connectivity optimization:* Performance of MEC systems depends on reliable user connectivity and adequate computational resources, requiring joint optimization of communication links and MEC resources, especially in dynamic scenarios with mobile users and MEC systems.

4. *Multi-level resource allocation and coordination:* Utilizing diverse resource types (R1, R2, and R3) across multiple levels enhance reliability and efficiency but introduce complexity in task allocation and scheduling, necessitating intelligent coordination to balance latency and dependability.

5. *Middlewares for computation and communication:* Middlewares provide an abstraction layer that simplifies the deployment of resilience mechanisms across diverse MEC resources. This enables interoperability and dynamic adaptation to failures and uncertainties while ensuring high availability and reliability.

6. *Bilateral reputation assessment:* Reputation models in MEC environments ensure mutual trust by evaluating the trustworthiness of both users and MEC systems, enabling secure collaboration and dependable service delivery.

7. *Cross-domain federation and access control:* These mechanisms are critical for enabling secure and efficient task handovers across

geographically distributed MEC systems, ensuring mutual authentication between mobile users and MEC resources to maintain trustworthy service delivery during mobility.

8. *Privacy-preserving task offloading and management:* These approaches are necessary to safeguard users' mobility and task offloading patterns that can be exploited by untrustworthy MEC hosts to infer sensitive information.

5.1 Adaptive Redundancy and Fault Tolerance

State-of-the-art virtualization and containerization technologies enable edge computing to implement flexible redundancy and fault tolerance techniques. Similar to the traditional PB approach, replicas of virtual applications can be initiated to serve as hot or cold backups and migrated across MEC systems in the event of operational errors or failures. However, given the diverse and dynamic characteristics of user demands, as well as the high competition for limited resources, it is not always feasible to generously deploy redundant instances on MEC systems for every application. This requires adaptive redundancy and fault tolerance techniques that efficiently utilize available resources under dynamic conditions, ensuring maximum operational uptime in the face of failures. Some of the most prominent approaches are summarized in Figure 5.1. Here, the simplest PB model (top-left in Figure 5.1) ensures seamless redundancy, which is particularly crucial for critical applications, albeit at the cost of doubling resource consumption. Alternatively, redundancy can be managed reactively (top-right) by migrating and reexecuting tasks after a failure, though this approach introduces additional latency. For stateful tasks, it is necessary to track execution progress—such as through checkpointing (bottom-left)—to enable resumption on a different edge resource if the primary host fails. The details of such approaches are further described below.

One of the most important challenges is deciding which applications require higher reliability than others, so redundancy and fault-tolerance strategies can be adapted accordingly. This is relevant in two key aspects. First, some applications are safety-critical by nature and may explicitly

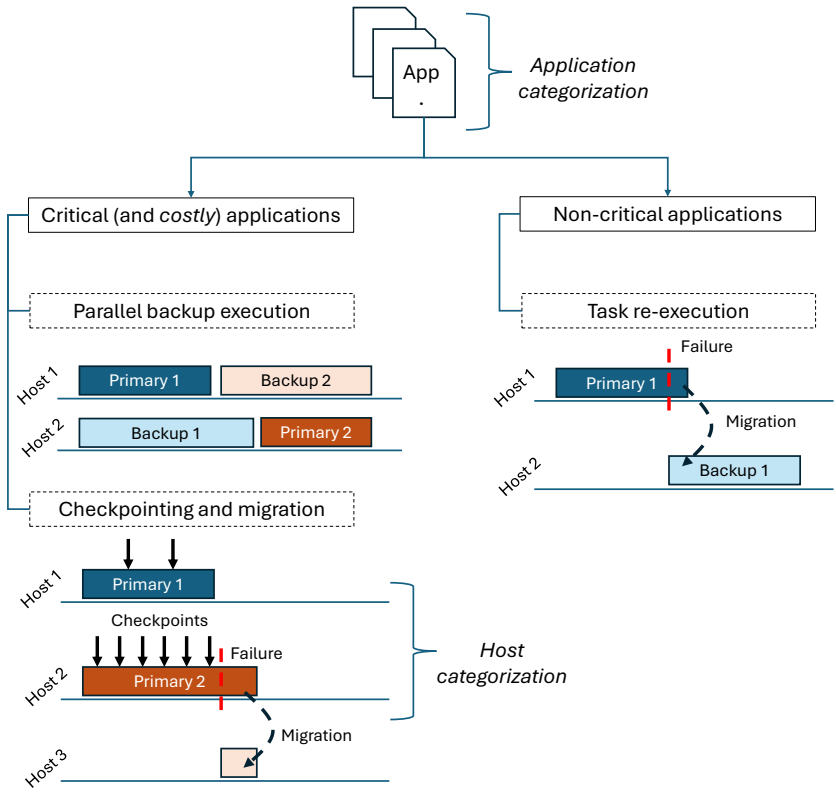


Figure 5.1: Illustration of different adaptive redundancy techniques, depending on diverse application and host characteristics.

request dependability guarantees, e.g., regarding mean failure time or predictable timing behavior. Second, other applications, while not highly critical, could be too costly for re-execution after a failure, potentially leading to additional operational errors and resource inefficiencies. Peng *et al.* [69] address the second challenge by *dynamically categorizing diverse MEC applications* according to the probability of experiencing an anomaly (C1). For instance, applications with a large number of predecessor or successor tasks (U1 and U2) are recognized as critical since their recomputation after a failure (C6). Similarly, applications requiring large amounts of data transmission in mobile environments

(U2) can benefit from redundant instances distributed across multiple MEC systems. This distribution helps to avoid additional latency due to retransmissions in case of impaired link quality (C3) between the MEC user and the host running the primary application instance (I1). The authors also propose that primary and backup instances for these selected applications perform computations in parallel, allowing the results to (i) be merged if one of the instances fails or (ii) take the earliest result otherwise. While this approach enhances MEC dependability proactively, i.e., with always-active backup instances, it also increases resource efficiency (C4) by adapting redundant resources to the characteristics of the applications.

The previous approach helps filter non-critical yet important applications for a resource-efficient PB scheme. In other scenarios, numerous latency-sensitive applications with explicit reliability requirements (U3) must be allocated and scheduled. This requires further adaptations to ensure the availability of primary and backup resources (C4). Wang *et al.* [97] propose a dynamic QoS-aware task scheduling mechanism that adapts the QoS levels of primary and backup instances dynamically during scheduling. The primary instances are scheduled to complete as early as possible within their time constraints, while the QoS levels of the backup instances are gradually relaxed to ensure the schedulability of the primary instances. Additionally, the resources allocated to the backups are released as soon as the primary instances are completed, improving resource efficiency. Such an approach combines both proactive and reactive redundancy measures. That is, the backup instance is activated by default if primary and backup instances are scheduled in parallel, providing proactive fault tolerance in case of a failure. Otherwise, the backup is computed later only if the primary instance fails, e.g., by migrating the remaining part of the primary instance at the scheduled time of the backup.

Such reactive fault tolerance approaches, where a failing application is executed partially after a migration, require *effective checkpointing techniques*. The MEC system must decide how often to record the operational state of an active application instance, i.e., defining checkpoints, to track remaining computations. This not only improves resource utilization but also ensures consistency for stateful tasks. Long *et al.* [57],

for example, adapt checkpoint intervals based on the failure probability of MEC systems with heterogeneous characteristics (C4). For example, less reliable systems, such as mobile ones (R3), should have shorter checkpoint intervals to enable precise migration of remaining parts of lengthy and data-intensive applications, thereby avoiding recomputations. This is illustrated in Figure 5.1, in which an unreliable host (host 2) has more frequent checkpoints than a reliable host (host 1), and only the remaining portion of the failing task migrated to another host (host 3). Another approach, described by Dong *et al.* [19], involves mobile MEC systems (e.g., vehicular hosts corresponding to R3) associating their availability with the duration they remain in a particular service area (C2). Each (main) host pairs with a shadow follower that executes the same applications as the main host at a reduced rate. The main host notifies the shadow follower in advance when it is about to leave the area, enabling faster computation of the backup tasks. By combining proactive (lazy computation) and reactive (early notification) methods, this approach eliminates the need for checkpointing.

The *characterization of MEC systems* is also important for assessing their run-time reliability, not only due to potential failures but also their non-deterministic behavior under high loads (C6). For instance, Ghanavati *et al.* [25] consider the uncertainty and dynamic nature of task execution runtimes (especially related to R2 and R3) to adapt their redundancy technique. Their stochastic model predicts task execution times for applications with diverse requirements (C1) and associate them with the likelihood of failure. Based on this model, they propose a hybrid proactive and reactive fault tolerance approach. Applications running on relatively unreliable hosts are protected proactively using traditional PB approaches, while those on more reliable hosts are re-executed only if the main instance fails, which is less likely. Similarly, Cheng *et al.* [11] formulate the uncertainties and diversity in application demands (C1) and the reliability of MEC systems (C6) and their connectivity (C3) to determine the best candidate hosts with different characteristics (e.g., more powerful R1 hosts and closer R2 hosts) for the given applications. They distinguish between the installation and activation of application instances. Specifically, application containers are kept ready on all candidate hosts but are dynamically activated only if the respective

hosts are operating reliably at a given time. This approach improves both resource efficiency (C4) and reliability.

5.2 Real-time Monitoring and Anomaly Detection

Every reactive approach described in the previous section relies on an effective monitoring and anomaly detection mechanism to identify unexpected changes and trigger the necessary actions. Unlike statistical models, which use predefined failure probabilities as abstract values [11], [25], these mechanisms continuously monitor specific system variables and make real-time decisions.

As illustrated in Figure 3.2, the MEC system consists of several modules responsible for task execution, orchestration, and management. Accordingly, an effective monitoring and anomaly detection module must be *capable of overseeing these modules and their complex interdependencies*. For instance, Park *et al.* [67] propose FATRIOT, a smart network interface card (NIC) for MEC systems, to deploy comprehensive fault and error detection features (C6), beginning from the data plane (e.g., as part of the VI module in Figure 3.2). Upon detecting packet processing failures, FATRIOT activates a fail-safe mode that seamlessly redirects affected traffic to a backup host as a reactive countermeasure. Moreover, it continuously heartbeats internal MEC system modules, services, and application instances (closely related to the I2 interface in Figure 3.1) to detect host and service unavailability as well as service-specific processing delays. However, this approach requires specific FATRIOT hardware, which may be feasible for R1 resources but less suitable for R2 and R3 resources due to their tight coupling with other systems, such as communication infrastructure and vehicles, respectively.

While FATRIOT and similar solutions such as [45] offer practical and powerful tools for monitoring, developing advanced anomaly detection models in complex MEC environments remains a significant challenge. Here, *AI/ML techniques are particularly well-suited to capture long-term behavioral patterns* of MEC systems and identify potential disparities. For instance, Wang *et al.* [96] propose a deep learning-based run-time anomaly detection method that observes the historical performance of

MEC resources based on service completion time. They associate the dependability of an MEC system with its ability to deliver results on time, which may degrade over time due to faults or volatile environmental conditions (C6). This is especially relevant to MEC resources in non-isolated environments (e.g., R2, R3). Such an approach not only aids in detecting faulty MEC systems, but also avoids assigning critical applications to hosts with occasional performance degradations – an important consideration for critical and time-sensitive MEC applications (U3). This is also related to the concept of *trust* in MEC environments, which will be discussed in Section 5.6.

Since MEC environments are highly dynamic, it is challenging to *distinguish between the root causes of performance volatility*, potentially caused by (i) anticipated challenges stemming from the nature of the MEC environment (C1, C2), and (ii) faulty or malicious scenarios (C6, C7). This necessitates more comprehensive models that consider multiple variables, unlike the previous approach in FATRIOT. A promising solution is presented by Tuli *et al.* [90]. The authors employ a generative adversarial network (GAN) leveraging several system metrics such as CPU over-utilization, abnormal disk utilization, memory leaks, and abnormal memory allocation. These metrics serve as stronger indicators for predicting potential faults in VMs and containers. This approach is also valuable for identifying checkpoints for virtual instances in real-time (see Section 5.1), thereby avoiding unnecessary task migrations that could strain the overall MEC system.

Despite their effectiveness, AI/ML models can suffer from high computational overhead when making real-time decisions. Moreover, out-of-the-box models may fail to *achieve optimal accuracy* under non-stationary user applications (U2, U3) and diverse host characteristics (C4) for anomaly detection. One potential solution is to design evolving models that continuously learn. For instance, Tuli *et al.* [91] combine backpropagation-based online learning with digital twins. On the one hand, the former constitutes an evolving model addressing dynamic MEC conditions (C1, C2). On the other hand, the digital twin improves application scheduling and resource allocation decisions of the AI model, by emulating them in advance and adapting the model parameters according to their optimality.

In addition to monitoring and anomaly detection mechanisms, various studies propose intrusion detection systems (IDS) tailored to the distributed nature of MEC resources (C5). For example, Li *et al.* [49] introduce an attack linkage mechanism that aggregates alerts and logs from multiple MEC systems to identify complex, distributed, and collaborative attacks. Similarly, Sharma *et al.* [82] develop hybrid and collaborative IDS solutions, combining rule-based, signature-based, and ML-based anomaly detection methods across distributed MEC servers to detect diverse attacks with minimal latency.

5.3 Joint Computation and Connectivity Optimization

The performance of a MEC system is often constrained by the reliability of user connectivity. This issue becomes even more complex when MEC resources themselves are mobile, as the quality of wireless links fluctuates rapidly based on the mobility patterns of both resources and users. Furthermore, even when users establish stable and reliable links with specific MEC systems, there is no guarantee that these systems will have sufficient computational resources available. As a result, ensuring service dependability must account for the provision of reliable communication links and the availability of adequate computational resources jointly.

Optimizing communication links between MEC systems and users is particularly critical for time-sensitive and critical applications (U3). Dong *et al.* [18], for instance, address ultra-reliability and low-latency communication (URLLC) requirements in 5G-based MEC environments (R2) to facilitate the offloading of user applications to available MEC systems. Specifically, they aim to minimize energy consumption and transmission errors within a non-orthogonal multiple access (NOMA)[1] scheme, while meeting the latency requirements of MEC users (focusing on their direct connection to MEC systems over the I1 interface). Under this scheme, their approach involves optimizing power allocation in NOMA to ensure tasks are offloaded to MEC systems inducing minimal latency. Similarly, Yang *et al.* [108] aim to minimize the error

[1]NOMA is a wireless communication technique that allows multiple users to share the same time, frequency, or code resources simultaneously by leveraging differences in their power levels or channel conditions.

probability for accessing and offloading tasks to integrated MEC systems (R2), assuming the use of finite blocklength (FBL) codes[2] for URLLC communication. They propose a reinforcement learning model to jointly optimize transmission time allocation and MEC computational resource allocation. It is worth noting that many previously discussed works emphasize the intricacies of MEC resource allocation while often considering communication link conditions in an abstract manner. In contrast, these two studies highlight varying aspects of networking technologies (C3) while employing relatively simpler resource allocation models. However, comprehensively modeling both MEC resource allocation and communication link optimization can significantly increase model complexity.

Initial task offloading to the optimal MEC systems with the best connectivity is a proactive approach to ensure reliable communication for MEC applications. Additionally, any task relocation equally requires *establishing stable links to ensure seamless migrations between distributed MEC systems*. Satria *et al.* [80] propose a reactive scheme to relocate applications from failed (C6) or overloaded MEC systems (C1) considering their connectivity with other hosts. Their approach assumes that MEC systems are integrated with the cellular communication infrastructure (R2). In this scheme, an overloaded MEC system offloads applications to neighboring MEC systems within its communication range, provided that certain link quality conditions are met. If there is no such neighbor, the overloaded system employs users as ad-hoc relay nodes to forward some applications to another suitable system via multi-hop connections. This scheme is also illustrated in Figure 5.2, in which the applications on System 1 is migrated to System 2 over a MEC user. Here, a logical inter-MEC connection (I3) is established over multiple host-to-user connections (I1); thus, requires a combined implementation of control and user interfaces.

The static deployment of MEC systems (e.g., R1 and R2) restricts resource allocation decisions to the given MEC topology and often faces connectivity challenges due to user mobility (C2). Mobile MEC systems

[2]FBL codes refer to coding schemes designed for communication scenarios with short codeword lengths, suitable for achieving low-latency communication under limited resources.

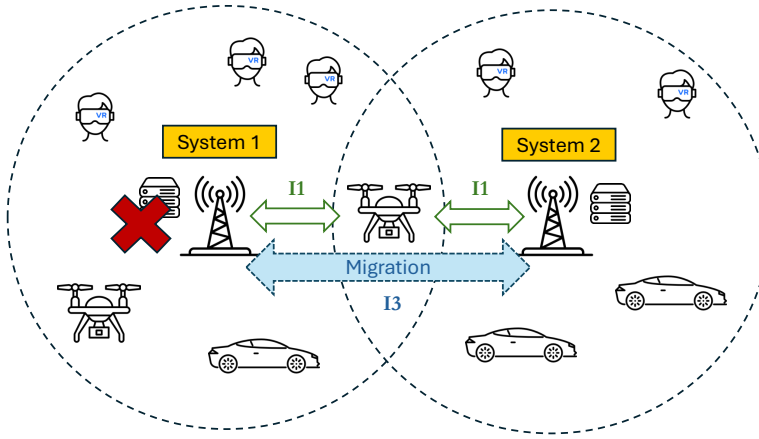


Figure 5.2: Multi-hop service migration after system 1 experiences an operational error. As system 2 is not its communication range, system 1 employs a user as a communication proxy. The selection of this node depends on connectivity to the MEC systems (I1) to form the most reliable multi-hop control channel for migration (I3).

can mitigate these issues by *relocating dynamically based on changing user demands (C1) and mobility patterns*. Ahmad *et al.* [1] develop a placement and mobility model for UAV-based MEC systems (R3), taking into account factors such as density, acceleration, trajectories, and speed to optimize mobile user connectivity in terms of transmission delays under bandwidth constraints (I1). Their model employs UAV-specific parameters in a federated learning-based collaborative resource allocation scheme, enabling real-time UAV deployment decisions. Complementarily, Falcão *et al.* [22] incorporate additional virtualization-related overhead (I2), VM setup times, energy consumption, and failure rates, for optimizing resource allocation on UAV-based MEC nodes. This approach provides a more fine-grained estimation of latency for URLLC applications (U3).

5.4 Multi-level Resource Allocation and Coordination

Different types of resources (R1, R2, and R3) impose trade-offs between latency and dependability, particularly in terms of their susceptibility

to failures and sensitivity to changing environmental conditions, as presented in Section 3. For instance, R1 resources represent well-established cloud and edge resources with greater processing power, whereas R3 resources are more dynamic (thus not as stable) but offer lower latency. Although different service providers often manage distinct resource types, leveraging all these resources jointly at different levels (e.g., based on proximity to users) can enhance resource efficiency, improve availability, and increase overall reliability. While utilizing a larger and more distributed pool of diverse resources offers greater flexibility, it also expands the solution space for task (re)allocation and scheduling problems, necessitating intelligent multi-layer resource coordination approaches.

Liu *et al.* [55] model the diverse characteristics of distributed resources (C5) at different levels (e.g., R1 and R2) based on their failure tendencies, expressed as failure rates. By considering these rates and the required availability levels demanded by MEC users (C1), they determine the minimum number of redundant application instances needed across different resource layers. In contrast, Sun *et al.* [86] adopt a more explicit distinction between resource levels. They deploy primary application instances on edge resources closer to users (e.g., R2 and R3) while performing backup computations on remote resources (i.e., stationary R1). Both approaches highlight *a critical trade-off in performance: leveraging remote, resourceful MEC systems introduces additional delays due to inter-MEC coordination (I3) and increased distance from users (I1)* as applications are transferred between hosts.

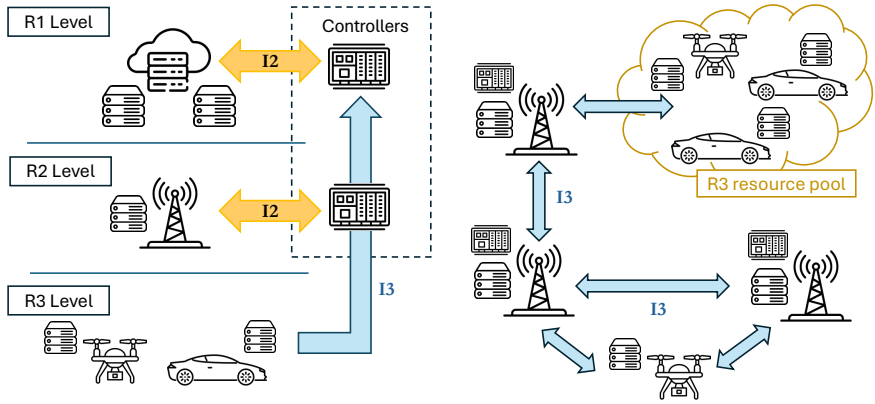
Multi-level resource allocation is particularly beneficial in high-mobility scenarios (C2). For example, in the Internet of Vehicles (IoV) environment, vehicles typically offload their computational tasks to roadside units (RSUs) or nearby vehicular MEC systems, depending on their connectivity. In worst-case scenarios, such as high-speed travel requiring multiple handovers between MEC systems, performing local computation directly on the MEC user can be a more reliable option, avoiding networking overheads and ensuring dependable task execution. For such scenarios, Wang *et al.* [95] propose a task offloading strategy that determines whether to rely on local computation, offload to dedicated R2 resources, or utilize temporary R3 resources, based on

vehicle mobility patterns. Their strategy leverages a software-defined networking (SDN) controller to coordinate multi-level MEC resources (I3). Guo *et al.* [27] adopt a similar approach in industrial systems, where local computation instead of MEC offloading can rapidly deplete the limited energy resources of users, such as mobile robots. They redefine the dependability objective in terms of the residual energy on an UE after completing a computation task, ensuring user availability. Their task offloading strategy balances computational energy costs (e.g., CPU cycles required to complete a task) with communication-related energy costs (e.g., transmission power) (I1) to maximize user operational time and, consequently, availability.

Despite their benefits, utilizing multi-level MEC resources introduces challenges related to their management and accessibility. As illustrated in the reference architecture in Figure 3.2, users and MEC systems must interact with a centralized proxy or orchestrator to register, offload, and maintain MEC applications. Although this approach is relatively straightforward within a single MEC system managed by one service provider, a multi-level and distributed MEC architecture (C5) that aggregates diverse resources may face scalability issues due to user query overload. To address these challenges, Tang *et al.* [87] propose a fully peer-to-peer (P2P) MEC architecture, where MEC systems communicate directly with one another to reduce configuration overhead (I3) and autonomously back up computational results on neighboring hosts, provided that sufficient resources are available within the P2P network. This decentralized approach, also shown in Figure 5.3, minimizes reliance on a central orchestrator, enabling faster responses to dynamic conditions, particularly in mobile scenarios (C2). However, a P2P MEC architecture also introduces additional challenges, such as host discovery and routing mechanisms, which must be addressed efficiently.

5.5 Middlewares for Computation and Communication

Distributed, heterogeneous, and multi-stakeholder nature of MEC resources (C5) make deployment of unified and collaborative resilience mechanisms significantly challenging. Middlewares serve as an abstrac-



(a) Multi-level coordination **(b)** P2P coordination

Figure 5.3: Multi-level coordination in MEC environments can be achieved either (a) through a logically centralized control plane, where multiple controllers collaborate to manage task offloading and system operations, or (b) through P2P coordination, enabling MEC resources to directly communicate and autonomously respond more quickly to dynamic conditions.

tion layer to decouple resilience mechanisms from the underlying infrastructure. Acting as a unifying framework, middlewares enable the implementation of redundancy, retransmission, and migration techniques in a way that is agnostic to the specific hardware, software, or management policies of the MEC resources. They provide general-purpose software modules that can be seamlessly deployed across diverse MEC systems, controllers, and user devices, facilitating interoperability and coordination.

Several works that we reviewed so far adopt service migration approaches to tolerate failures at MEC systems. This usually requires additional and time-consuming *mechanisms to maintain message backups and perform system rollbacks*. To address this, Wang *et al.* [92] propose a fault-tolerant real-time messaging middleware based on publish/subscribe model. They categorize different types of data traffic for MEC applications (C1) in terms of their loss-tolerance level (acceptable number of consecutive message losses), and end-to-end latency deadline. Their middleware handle scheduling of messages with data traffic (I1), forwarding them to the migrated service instances in case

of host failures (I3), and compensate for message losses. Similarly, Javed *et al.* [35] developed an fault-tolerant framework for vehicular edge systems to cope with hardware failures (C6) and connectivity disruptions (C3). They define an intermediate communication layer as a publish/subscribe data communication pipeline based on Open Messaging Interface (O-MI) and Open Data Format (O-DF) standards[3] to decouple their messaging framework from the underlying networking technologies and protocols. This pipeline helps distributing the results of MEC applications to the multiple users reliably (I1). Besides, they implement a Kubernetes-based[4] management layer to orchestrate the modules of this framework so that if the user temporarily fails, their message proxy can be reinitiated in a stateful and reactive manner.

Maintaining the state information of MEC applications is crucial not only for recovery after failures but also for *enabling seamless handovers in high-mobility scenarios (C2)*. Harchol *et al.* [29] introduce a messaging middleware designed to track the messaging order and computational state of data-intensive video analytics applications in MEC environments. This middleware deploys its modules on MEC systems, facilitating the migration of application instances across successive hosts by leveraging user movement projections over inter-MEC control channels (I3). By doing so, the system ensures uninterrupted computations in mobile scenarios and enables the rapid migration of stateful services reactively in case of failures (C6).

Samanta *et al.* [77] take an alternative approach for task offloading by *avoiding the assumption of a centralized orchestrator* making optimal decisions and involve all MEC actors in the offloading process through a middleware. They define an MEC proxy as a middleware to implement an auction-based task offlading strategy. In their framework, MEC users submit their bids to the middleware proxy to declare the amount they can pay for the computation of their tasks. They consider potential failure conditions in their bidding: more risky task offloads, e.g., due to unstable connectivity (C3), have higher price. Similarly, MEC systems submit their bids to execute the demanded applications. Their bids are

[3]The Open Group, Open Messaging Interface Technical Standard, https://www2.opengroup.org/ogsys/catalog/C14B.

[4]Kubernetes: Production-grade container orchestration, https://kubernetes.io/.

also proportional to their availability and reliability, e.g., if they are overloaded (C1) or faulty (C6). In this way, they are incentivized to perform dependable computations. In the end, the proxy assigns tasks according to these bids and executes payments from users to hosts only for successful services.

5.6 Bilateral Reputation Assessment

Reputation-based trust models in MEC environments address the critical challenge of ensuring mutual trust between users and MEC systems by evaluating the reliability and behavior of both parties. On the one hand, these models assist users in identifying trustworthy MEC systems by aggregating reputation scores based on past performance, thereby ensuring reliable task execution and resource management. For example, users can prioritize MEC systems with high reputation scores for consistent service quality and minimal failure rates (C6). On the other hand, MEC systems leverage reputation scores to evaluate users, identifying those with a history of legitimate and efficient task requests (C7). This bilateral reputation mechanism protects MEC systems from malicious or resource-draining user activities, such as excessive task requests or tampering attempts. By fostering trust in both directions, reputation-based models enable secure collaboration, secure resource allocation, and dependable service delivery, creating a resilient MEC ecosystem.

One approach to assess the reputation of MEC systems is by *evaluating their compliance with service level agreements (SLA)*. Monir *et al.* [64] propose measuring user dissatisfaction rates based on four SLA elements: cost, maintenance, storage capacity, and execution time. Each MEC resource is categorized according to its commitment to these elements. For instance, while an expensive service provider might result in user dissatisfaction, its execution time and maintenance commitments could still be satisfactory. This feedback mechanism discourages MEC providers from overloading their resources and risking service quality, which is particularly relevant for R2 and R3 resources, given their significant resource constraints. However, relying solely on user feedback introduces vulnerabilities, as it can be exploited by bad-mouthing attacks [72].

Assessing the reputation and trustworthiness of MEC users requires different considerations compared to MEC systems. Heydari *et al.* [31] formulate this in terms of a *risk factor* for users, based on practical aspects such as their service request time and location, and authentication success. In this simple reputation scheme, user requests are classified as risky if they are made (i) at unusual times of the day or (ii) from atypical locations. Additionally, unsuccessful authorization attempts with incorrect credentials are penalized more severely, marking the respective users as untrustworthy. While this approach adds a layer of trust management, it risks a high false-positive rate, denying services to legitimate users merely due to changing or atypical service patterns (C1). Such methods based on direct trust assessments fall into *direct trust* evaluation approaches illustrated in Figure 5.4.

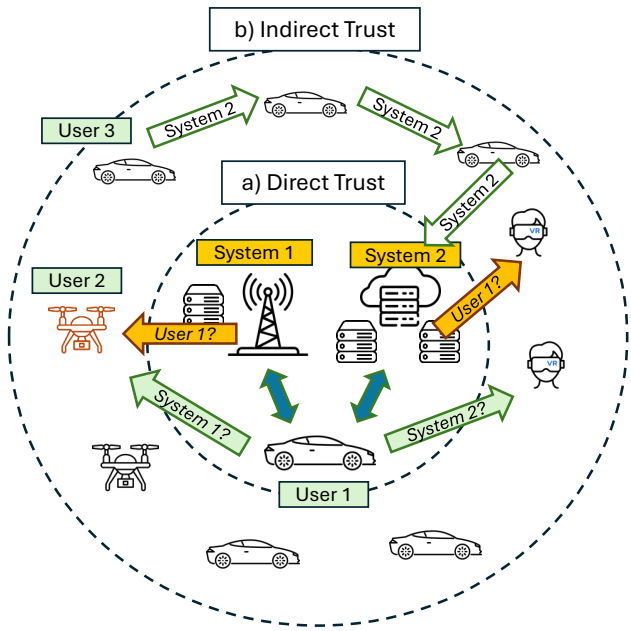


Figure 5.4: Direct and indirect trust zones. Direct trust is established based on historical interactions between two MEC nodes, while indirect trust is derived from feedback provided by other nodes. In the figure, MEC systems (systems 1 and 2) and users (e.g., user 1) request reputation scores from neighboring nodes to inform task offloading decisions. Meanwhile, user 3 leverages an end-to-end path of trust to establish a connection with system 2.

To address these limitations, bilateral reputation and trust management mechanism is more effective, as it *evaluates both MEC systems and users symmetrically* based on similar metrics. This is particularly important in scenarios where users can also act as service providers (R3). Wang *et al.* [94] propose a unified reputation scheme with metrics such as feedback satisfaction, service willingness (associated with their varying capacity and capabilities (C4)), and community relations to assess the trustworthiness of all MEC nodes based on their successful interactions. For instance, in Figure 5.4, while User 1 asks other users for the reputation of Systems 1 and 2, the systems also investigate the reputation score of User 1. While these metrics have slightly different implications for MEC systems and users, they help evaluate the overall *credibility* of each node. Notably, the community relations metric incorporates the role and popularity of a node within a social network, inspired by human behavior that tends to trust individuals within the same community.

Building on a similar community-based approach, Alioua *et al.* [3] introduce a reputation scheme where MEC systems calculate a trust score for MEC users based on feedback from nearby users when receiving an application request. This score, derived from subjective logic [41], considers the user's previous successful interactions and communication quality (C3); thus, also an indicator for its dependability. In a subsequent step, MEC systems also request for an indirect reputation score of the respective user from a set of other hosts and users that participate in a blockchain-based trust management mechanism. Specifically, auditors and verifiers in the blockchain environment lead to calculate a secondary reputation score of the respective node in consensus. The whole process is recorded in a shared blockchain, and thus integrity of the score is ensured. While this approach allows for broader participation of the selected (or trusted) nodes in reputation assessments, it also introduces additional roles and components, such as auditors and a trust authority, making the trust scheme more complex.

Establishing trust through direct and indirect reputation assessments is typically feasible when a MEC node (resource or user) can interact directly with the respective node or its immediate community. However, in highly mobile and dynamic environments (C2), such as vehicular

MEC resources and users (R3, U3), this approach is often impractical. This is because nearby MEC nodes frequently change and connections are disrupted, yet end-to-end trustworthy connections still need to be established. Zhang *et al.* [112] introduce the concept of a *path of trust*, enabling a MEC user to maintain a connection with a MEC resource even as they move further apart. The approach relies on a multi-hop path (between user 3 and system 2 in Figure 5.4) composed of trustworthy intermediate nodes, where the end-to-end trust score is calculated by multiplying the reputation scores of all nodes along the path. The trust score is based on previous interactions and current connection quality between nodes, which can also be associated with link reliability. The multiplicative calculation has the reasoning that, longer paths are more prone to disruption (C3) and are more likely to include a malicious node, leading to lower overall trust scores. To mitigate these risks, users can select paths with the highest (accumulated) trust scores to maintain their connections with specific MEC systems. Alternatively, they may dynamically choose to offload tasks to a new host with a better path of trust, if available, ensuring reliable and secure service delivery.

5.7 Cross-domain Federation and Access Control

As discussed in the previous sections, highly mobile MEC users (U3) can easily lose connection with MEC systems or experience untolerable delays due to increasing distances. This requires handing over their tasks across multiple geographically adjacent MEC systems (e.g., RSUs or base station-integrated hosts) according to their moving trajectory. These hosts can be managed by different service providers, or distributed units of the same provider that are control semi-autonomously within different MEC domains, i.e., geographical regions or control areas. Accordingly, handovers necessitate proper mutual authentication and access control mechanisms to ensure that services are provided only to (benign) authorized users, and the users needs to authenticate MEC resources to make sure that they are real and not compromised [106].

While existing authentication and access control approaches could be embraced in MEC environments, *they should still be adapted to the distinct user characteristics*. For instance, an authentication scheme

must be sufficiently lightweight for deployment resource-constrained devices (U1, U2). Besides, it should be performed quickly to minimize time and messaging overhead (I1) for mobile users (U3). To address those aspects, Jia *et al.* [37] propose modifying anonymous authenticated key agreement (AAKA) protocol, which prevents the disclosure of user private information while ensuring the authenticity of their identities, as well as producing a common session key to facilitate subsequent interactions. It has only one round message exchange overhead, thus suitable for high mobility MEC users, as long as they are registered with a trusted registration center before. Although such a trusted entity is sufficient for access control within a standalone MEC system (I2), a similar scheme across multiple MEC domains necessitates additional components to evaluate the legitimacy, e.g., reputation, trust, and access rights, of mobile users and perform handovers to the respective MEC systems in different domains. This is especially important when these domains have different access requirements and security levels.

Therefore, He *et al.* [30] introduce a cross-domain access control protocol, including additional components such as the reputation management server (RPM), the cross-domain request server (CRQ), and gateway nodes for each MEC domain. The RPM manages the reputation of users in a particular domain, i.e., a regional reputation, and also calculates a more extensive cross-domain reputation based on previous cross-domain access attempts of the respective user. The CRQ handles user requests for cross-domain access, e.g., for handovers or simply fetching information from a different MEC system. Additionally, a centralized cross-domain relay server coordinates these requests between different domains over the gateway nodes in each domain (I3). Here, a user with sufficient reputation or access rights for a domain can be rejected by another domain. That is, while such a comprehensive scheme helps build a large-scale access control scheme, it still preserves domain-specific access control policies that are enforced by different MEC systems.

Introducing additional cross-domain orchestration components into MEC systems increases the complexity of an already highly heterogeneous and distributed MEC environment (C5). Moreover, components such as centralized registration centers and RPMs, as proposed in the aforementioned works, introduce additional trust anchors, which require

continuous verification and protection to ensure their legitimacy (C7). To address the reliance on such entities, Liu *et al.* [56] propose a domain-committee architecture. Here, a domain does not necessarily represent an independent MEC system but rather a collection of MEC servers that are, for example, geographically proximate. In their proposal, a committee of MEC servers within each domain run a Byzantine fault tolerance protocol[5] to make collective decisions for cross-domain access and data sharing (I3). They also maintain their intra-domain blockchain instance to process data sharing requests. This approach is further enhanced with the integration of a public blockchain between domains to guarantee secure cross-domain operations. The overall architecture is illustrated in Figure 5.5. In this setup, a mobile user initially authenticated in domain 1 can push its data to domain 2 using the proposed inter-domain data-sharing mechanism. Likewise, the user can offload tasks to domain 2 without needing to re-authenticate.

An efficient access control and authentication scheme should also address *scalability issues*, which are particularly relevant to vehicular edge computing scenarios (R3). That is, numerous moving vehicles can strain a single authentication server, i.e., deployed on an RSU or an external remote entity as proposed in the aforementioned studies. Besides, the proposed handover techniques can still be latent due to far distances between distributed MEC systems (C5). To tackle these challenges, Liu *et al.* [53] propose a cooperative and decentralized low-latency authentication scheme, tailored for vehicular users (U3). In this scheme, multiple delegated proxy vehicles cooperate to authenticate vehicular users on the road. Different sets of proxy vehicles constitute authentication groups via secret sharing, and each group is associated with a distinct MEC system, e.g., RSUs or base stations (R2). Their trustworthiness is also ensured by a tamper-proof blockchain scheme, which manages the trust values of the proxy vehicles based on their previous authentication records. Moreover, these vehicles can be re-delegated reactively depending on their availability and changing network conditions; thus, ensures adaptability of this scheme.

[5] Byzantine fault tolerance indicates that a distributed system can reach consensus and operate correctly even if some nodes fail or act maliciously by relying on majority agreement.

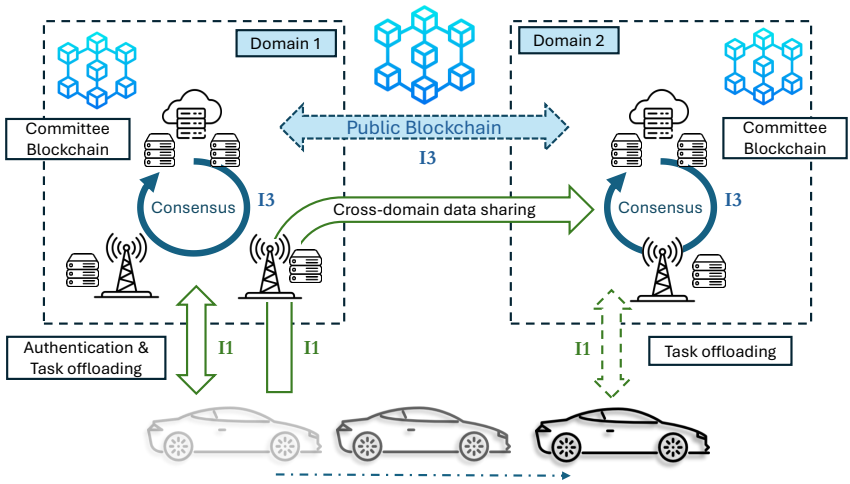


Figure 5.5: A distributed cross-domain access control and authentication mechanism proposed in [56]. In this approach, multiple MEC systems within domains 1 and 2 collectively reach a consensus to grant access permissions to users. The integrity and security of this process are ensured through their distributed committee blockchain. To facilitate cross-domain authentication and data sharing, interconnected domains utilize a public blockchain to record and track their decisions.

5.8 Privacy-preserving Task Offloading and Management

Untrustworthy MEC hosts can exploit the mobile nature of MEC users to breach privacy in several ways. For instance, location of mobile users can be inferred by analyzing the size and frequency of offloaded tasks, as closer proximity results in larger data transfers with lower latency. This leakage also enables malicious entities to derive more sensitive information, such as movement patterns potentially leading to extract personal and social habits [111]. Furthermore, providing high-quality services, e.g., for video streaming and gaming (U2), in MEC often requires analyzing user histories for accurate QoS predictions to provide the desired service quality seamlessly. While this enhances user experience, it can also expose users to profiling and surveillance. Addressing these challenges requires privacy-preserving task allocation and management mechanisms that can protect user data from untrusted entities, without compromising their QoS experience in MEC environments.

The first challenge in privacy-preserving task offloading is selecting a trustworthy MEC host that minimizes data leakage risks (C7). However, the high contention for limited resources (C4) in MEC environments may *prevent all users from utilizing a single MEC host with the best reputation, as this could overload that particular system.* To address this issue, Wu *et al.* [105] associate the privacy requirements of different user tasks with the reputation of MEC hosts, thereby developing a privacy-aware offloading strategy. They introduce *task sensitivity* to quantify the threat to user privacy if sensitive task-related information is leaked. Furthermore, they model an MEC social network to map trust relationships between MEC users and hosts based on identity and behavioral trust evaluations. A task is offloaded to a host when its sensitivity level (categorized into three levels) aligns with the host's associated trust value. To achieve optimal matching, the authors consider various offloading modes, including local computations (see Section 5.4) and P2P offloading (R3).

The integration of task sensitivity and trust relationships facilitates offloading critical tasks to reputable MEC hosts. However, from a privacy perspective, exposing users' private information to MEC systems, even those with high reputations, is still a concern. For instance, due to the mobile nature of MEC users (C2), their changing locations must be known to ensure service availability but can also be exploited to infer unintended user information [111]. To address this, Wang *et al.* [103] propose a location perturbation approach, where users perturb their location information based on their privacy requirements while remaining within the service coverage of MEC hosts (e.g., R2 resources integrated into base stations). The primary challenge is determining the perturbation region adaptively, such as the range within which a user perturbs their location, so tasks can still be offloaded to MEC hosts with the highest utility while minimizing information leakage. The authors employ a differential privacy[6] approach to measure privacy loss and evaluate trade-offs between this loss and utility gains. To address a similar issue, Cui *et al.* [13] introduce the concept of a *privacy area.*

[6]Differential privacy is a mathematical framework and guarantees that an individual's data cannot be inferred from a data collection. Thus, it ensures the privacy of individuals, while still allowing analysis of the overall data.

As illustrated in Figure 5.6, User 1 can connect to MEC systems 1, 2, and 3 (denoted by their coverage areas) to access their services and thus have a large *utility area* that equals to the total service coverage of all MEC systems. However, this could allow them to infer that the user is located within a small privacy area (area 1, blue dashes). In contrast, user 2 can only connect to system 3 and could be located anywhere in its service range, i.e., a large privacy area (area 2, purple) but a small utility area limited with the range of system 2. Consequently, the user's goal is to select the *optimal* set of MEC hosts that maximize the retrievable information or available services while minimizing the risk of localization. The authors then propose an optimization problem that (geometrically) balances the size of privacy area and utility area, ensuring the user can still use all necessary services provided by the minimum number of MEC hosts.

Apart from the general location-privacy challenges addressed in the aforementioned studies, other MEC applications may require sharing specific metrics and data, which can also lead to privacy concerns. For instance, in [113], MEC users require QoS-related metrics to forecast the service quality of different MEC hosts for a video streaming application (U2), enabling them to connect to the best resources within a specific region. To achieve this, they request the historical QoS values of similar users who were served by MEC hosts in the same region. The assumption is that users in similar regions experience comparable latency and video quality, as these metrics mainly depend on the local communication infrastructure. However, users are often reluctant to share such information, as it could expose their application usage patterns and preferences. To address this issue, the authors propose a differential privacy method, where users disguise their QoS metrics by adding Laplace noise[7] while still allowing accurate forecasting for other users. This noise is dynamically adapted (i.e., reactive) to account for the changing conditions resulting from user mobility (C2) and dynamic connectivity issues (C3), ensuring that the disguised value remains representative of the respective MEC region. This approach

[7]Laplace noise is a form of random noise derived from the Laplace distribution. It is added to data in the context of differential privacy to ensure privacy preservation while still allowing for meaningful analysis.

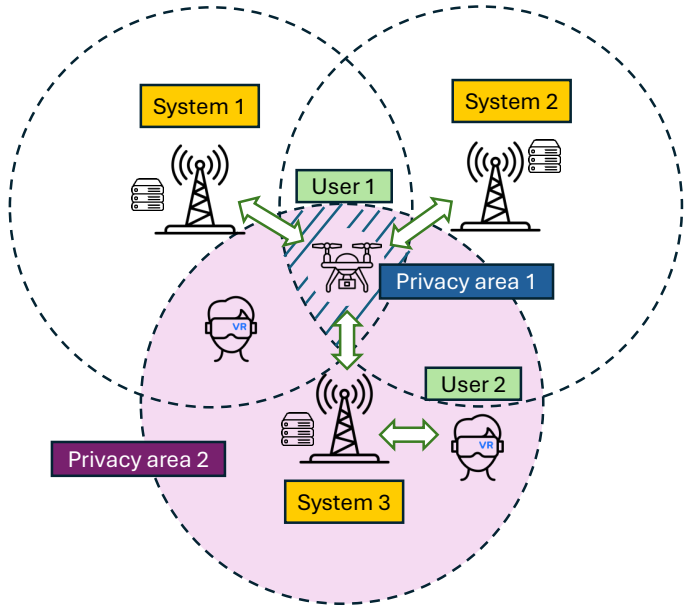


Figure 5.6: The concept of privacy area. When user 1 interacts with the system 1, 2 and 3, inferring its location is easy since it should be somewhere within the intersection of service coverage areas of these systems. Therefore, it has a small privacy area (blue dashed). In contrast, user 2 is connected only to system 3 and thus could be located anywhere in a larger area (pink). The concept is overall useful to register a user to the minimum number of MEC systems to maximize its privacy area, ensuring service quality and availability.

balances privacy preservation with the need for reliable service quality predictions in dynamic MEC environments.

6

Discussion and Future Directions

As a result of analyzing MEC challenges, reviewing several related works, and deriving overarching resilience measures and concepts, we have identified several key insights and future directions to enhance the resilience of MEC systems. This section highlights these takeaways and potential areas for future research.

Joint vertical and horizontal MEC orchestration

In Figure 3.1, we illustrate a highly heterogeneous and distributed MEC ecosystem, identifying these as two critical challenges in Section 4.1. These challenges are also reflected in the resilience concepts we discussed within the context of multi-level and cross-domain measures, particularly in Sections 5.4 and 5.7. The former addresses resource allocation, connectivity, and resilience issues *vertically* across different types of MEC resources (R1, R2, and R3), while the latter focuses on similar issues *horizontally* across diverse MEC systems (e.g., those belonging to either R1, R2, or R3). In essence, multi-level coordination emphasizes resource characteristics such as capacity, proximity, and dependability, whereas cross-domain problems stem from the multi-stakeholder and distributed nature of various MEC systems. A truly holistic MEC per-

spective requires integrating these vertical and horizontal optimization approaches to tackle distinct challenges related to C4 and C5. This holistic view is also illustrated in Figure 6.1.

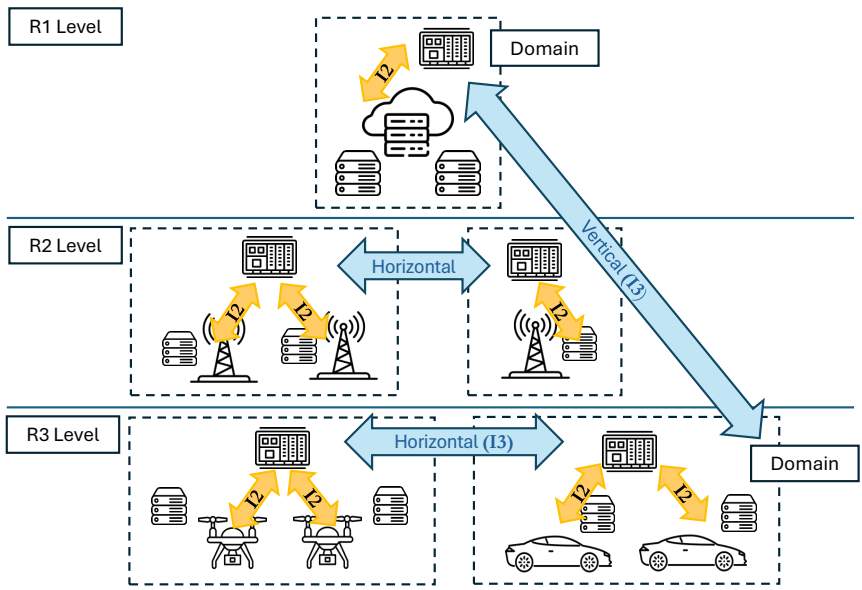


Figure 6.1: Vertical and horizontal inter-MEC optimization requires interaction between several distributed MEC systems and domains with different characteristics. Here, a domain can include multiple MEC systems, orchestrated by a single controller.

Such holistic approaches have several implications. For instance, the cross-domain access control mechanisms discussed in Section 5.7 involve handovers between geographically adjacent MEC systems. In these schemes, assuming a proxy MEC entity (or controller) to manage interactions with subsequent MEC systems is relatively straightforward. However, extending this to multi-level coordination between remote and heterogeneous MEC resources demands a more comprehensive controller architecture capable of accessing all MEC resources within acceptable latency constraints. Here, blockchain-based approaches (as also underlined in Section 5) can help evolve rather centralized management and orchestration functions of MEC environments to distributed and collaborative decision mechanisms with tamper-proof data structures

and processes. Another issue is that multi-level coordination models, as outlined in Section 5.4, often simplify the task by selecting among R1, R2, or R3 resources for offloading. This abstraction reduces distributed and cross-domain MEC systems of the same resource type to a single MEC entity, failing to capture the complexity of task offloading across distributed systems and missing the potential benefits of (horizontal) resource scalability. Joint vertical and horizontal coordination for large-scale MEC systems may also require topological optimizations to strategically place MEC resources, ensuring efficient and seamless operation even in the event of failures or attacks.

The final issue pertains to the standardization of cooperative MEC systems. The leading standardization effort, ETSI (see Section 3.2), does not address interactions between MEC systems owned by different service providers. It remains unclear how their orchestrators should cooperate to holistically utilize diverse MEC systems. This necessitates defining inter-MEC coordination standards and requirements, such as configuration models, data types, security constraints, and privacy policies. Another important challenge is harmonizing and adapting the different, potentially competing resilience priorities of distinct MEC providers.

Artificial intelligence and federated learning

MEC is regarded as the *front layer* for processing big data using algorithms based on AI and ML in various technological domains, including video content analysis, autonomous vehicles, and large-scale IoT systems [34]. From a resilience perspective, these algorithms are particularly valuable for capturing behavioral patterns in complex and dynamic MEC environments, detecting anomalies, and developing adaptive resilience strategies [84]. In Section 5, we have already presented some application scenarios of AI/ML, and here we briefly discuss their benefits and related challenges.

First of all, in distributed MEC environments (e.g., those employing multi-level and cross-domain orchestration), centralized AI models can correlate logs and telemetry data across multiple MEC systems to identify faults and attacks in large-scale interconnected incidents. AI-

based anomaly detection models extend beyond traditional solutions, addressing dynamic, complex, and interconnected failure and attacker models that are often overlooked in existing studies. Moreover, they can be leveraged for forecasting anomalies as part of predictive analytics methods.

From a model development perspective, federated learning (FL) is a promising approach, which aligns well with the distributed and interdependent nature of MEC systems. It enables the development of local and global models for anomaly detection, user behavior analysis, and task management [51]. For example, a standalone MEC system can allocate resources and schedule task handovers to neighboring MEC systems by predicting user mobility patterns using a global FL model collaboratively trained by multiple MEC systems serving those users. This approach ensures seamless availability and high reliability for critical tasks. FL also inherently preserves the privacy of individual MEC systems, reducing concerns about inter-MEC data sharing (I3). Additionally, FL alleviates the training burden on resource-constrained MEC systems (R2, R3) by enabling distributed and collaborative learning [16].

Despite these advantages, implementing AI/ML models in mobile, heterogeneous, and distributed MEC environments comes with challenges. Dressler *et al.* [20] emphasize the critical considerations for designing ML-based orchestration models tailored to MEC, particularly in ad-hoc resource environments (R3). These models must address diverse service requirements (C1) and the heterogeneity of network and infrastructure (C3, C4). They should also be continuously trained and updated in response to the changing demand dynamics caused by user mobility (C2). Finally, trust remains a significant challenge for federated learning approaches in heterogeneous and multi-stakeholder MEC environments [79]. That is, it should be ensured that no untrustworthy MEC system can poison the global model. For this case, bilateral trust assessment mechanisms has an additional importance to prevent compromised MEC system and users contributing to the development of FL schemes.

System and network hardening

Although we do not specifically focus on these aspects, system and network hardening are fundamental to the resilience of MEC environments. First, strict hardware security and isolation mechanisms are essential, as MEC services increasingly depend on secure cryptographic operations. Hardware Security Modules (HSMs) have long served as a trusted foundation, but they face scalability challenges in multi-tenant and shared execution environments—especially with the rise of microservices requiring frequent cryptographic operations. This underscores the need for scalable and efficient hardware security platforms that maintain strong isolation in dynamic MEC systems [28].

Second, virtualization-level vulnerabilities pose significant threats, not only enabling malicious attacks but also causing potential contamination in shared resources due to inadequate isolation at the OS, hypervisor, or container layers [10]. Additionally, software vulnerabilities in MEC services can serve as entry points for attackers to exploit other MEC components, leading to unauthorized access and privilege escalation. Here, with suitable hardware support, Trusted Execution Environments (TEEs) can ensure the confidentiality and integrity of computations within virtual instances. For instance, confidential virtual machines (CVMs) enable encryption of the (virtual) OS and application-specific data, remote attestation, and verifiable isolation for critical services within TEEs [81]. Trusted containers with integrity protection can also be a lightweight alternative to heavier VM solutions for multi-tenant and dynamic MEC systems (R2, R3) hosting mixed-criticality services [50]. Note that the networking and computational overhead of remote and continuous attestation for such solutions must be carefully considered, as they necessitate additional cryptographic operations and data exchange between MEC systems, controllers, and third-party verifiers [14], [89].

Third, robust network hardening measures—such as network segmentation, traffic monitoring, and separation—are crucial [36], particularly in heterogeneous MEC environments with multiple providers (C4) and mixed-criticality applications (C1). Finally, ensuring the physical resilience of MEC infrastructure is challenging due to its geographically

distributed nature (C5). Without proper isolation, protection, and monitoring, MEC systems remain vulnerable to physical threats. Especially in remote areas, additional safeguards such as earthquake-proofing, fire detection and control, breach monitoring, and physically secure perimeters are critical [75]. To address these challenges, hardening measures must ensure that all hardware, software, and network configurations as well as the deployment of physical infrastructure are properly set, continuously monitored, and regularly tested for compliance with certain MEC resilience standards, which have not been formally defined so far.

Zero-trust architecture

In heterogeneous MEC environments with numerous stakeholders, authentication mechanisms verify the identities of MEC systems and users (see Section 5.7). However, trust extends beyond authentication by incorporating confidence in an entity's behavior, reliability, and intentions over time. This is essential in scenarios requiring repeated interactions, data integrity, fairness, or secure collaboration between stakeholders with varying motivations.

In Section 5.6, we reviewed several reputation assessment mechanisms for involving trust in MEC environments. Beyond these standalone mechanisms, a proper architecture with standardized facilities is needed to enable secure interactions between diverse MEC systems and users. The National Institute of Standards and Technology (NIST) defines a reference model for zero trust architecture (ZTA) [74], which assumes no implicit trust between users and systems. It manages trust through components like the policy engine, administrator, enforcement mechanisms, and identity and access management. Integrating these ZTA components into existing reference models, such as ETSI in Figure 3.2 or into a more comprehensive system model in Figure 3.1, remains an active research area. For instance, Dhanapala *et al.* [17] implement a policy engine, incorporating reputation assessment methods like those reviewed in Section 5.6. In [2], ZTA is integrated with the 5G network stack, leveraging R2 resources deployed at base stations for trust assessment functions. Similarly, open networking paradigms such as Open Radio Access Networks (O-RAN) facilitate synergies between network

infrastructure, ZTA, and MEC to implement tightly coupled access control mechanisms [39].

The implementation of policy engine and enforcement components should also be aligned with the unique MEC characteristics. For example, MEC users with diverse and dynamically changing service requirements (C1) cannot be evaluated solely through static behavioral trust assessments. This necessitates secure authentication methods integrated with trust-based access control policies to ensure that highly mobile users with critical tasks (U3) are not unfairly penalized by potential false negatives in trust evaluations. AI/ML models could provide more accurate assessments of these changing behavioral patterns, as briefly discussed in the previous section.

Multi-dimensional space-air-ground MEC environments

In Section 3.1, we described R3 resources as mobile and ad-hoc MEC systems and provided examples of UAV-based flying MEC servers that collaborate with ground-based MEC systems [33]. The integration of air-ground MEC systems creates a comprehensive MEC environment, where users can leverage the robust capabilities of ground-based MEC systems (R1, R2) alongside the flexibility of dynamic air systems (R3), which can serve various locations on-demand. Here, it is also possible to imagine UAV-based base stations with integrated MEC systems (e.g., a combination of R2 and R3) [12]. However, such integrated systems and the combination of air-ground MEC environments bring several research challenges.

One key challenge is determining the mobility trajectories and optimal placement of flying MEC systems. These must align with the mobility patterns of users and adapt dynamically to changing demands. Additionally, placement strategies should ensure efficient load balancing between air and ground MEC systems. For instance, mobile MEC systems can act as a last resort when ground-based systems cannot meet user demands, to moderate their utilization and frequent relocation, and eventually avoid their energy depletion [104]. Reliable networking is another critical issue. Communication and channel modeling, along with network coverage in three-dimensional space, differ significantly

from terrestrial networking [32], adding complexity to joint computation (e.g., task offloading) and communication optimization problems (see Section 5.3). These challenges are further amplified in inter-MEC collaboration scenarios, requiring consideration of UAV-to-UAV, UAV-to-ground MEC, and UAV-to-user interfaces simultaneously. Addressing these complexities necessitates advanced AI/ML models or effective heuristics rather than traditional constrained optimization approaches.

Air-ground MEC environments can also extend into space by incorporating satellite systems to enhance connectivity [71]. Space-air-ground MEC systems align with emerging 6G networks, which aim to seamlessly integrate satellite systems, high- and low-altitude platforms (HAPS/LAPS), and terrestrial base stations to maximize network coverage [5]. A comprehensive picture covering all potential components of space-air-ground MEC is shown in Figure 6.2. Initially, integrating MEC with satellite systems might appear counterintuitive, as the primary goal of MEC is to bring computational resources closer to users. However, satellite-based MEC nodes can directly process computational tasks for users in areas with limited or no terrestrial coverage. This reduces the need to transmit large data volumes to distant cloud data centers. Satellites can also serve as a last resort during disasters, adding another level of MEC resources, as discussed in Section 5.4, and thus enhancing the resilience of the system. Despite these advantages, space-based MEC systems face distinct challenges, such as connectivity constraints, computational resource limitations, and issues related to security and synchronization [26], [52]. Finally, viewing space-based MEC systems as a multi-layered resource, with nodes placed at different altitudes, such as low earth, medium earth, and geostationary orbits (LEO, MEO, and GEO), introduces additional modeling opportunities. While these setups increase complexity, they also offer greater flexibility for task offloading and data sharing in orbital MEC environments [109].

Energy efficiency and sustainability

Sustainability in MEC has been an ongoing trend, with its relevance spanning various aspects of MEC, from the hardware design of MEC hosts to energy-efficient resource allocation and load balancing [6],

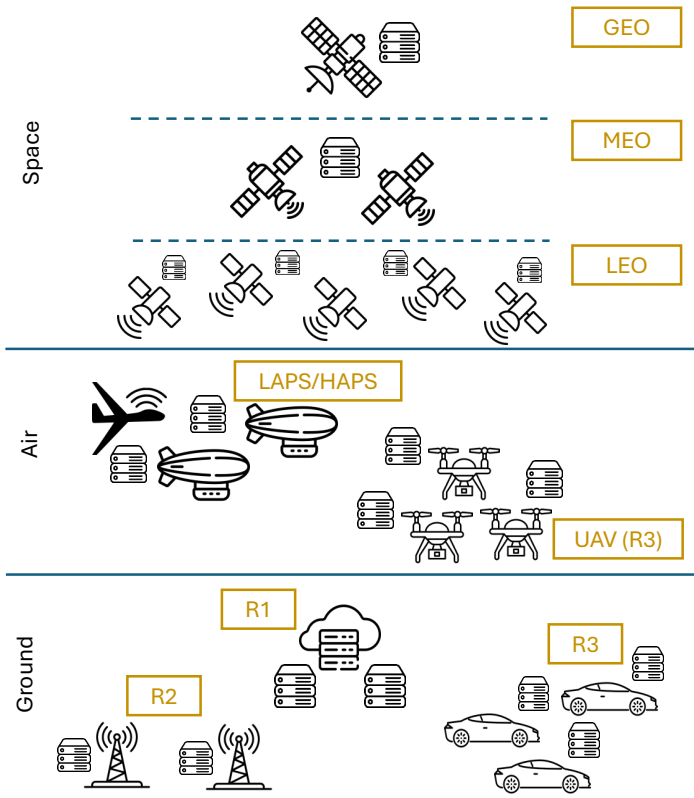


Figure 6.2: Multiple layers of the integrated space-air-ground MEC environments in terms of different altitudes.

[62]. Although not extensively discussed among our resilience measures, energy efficiency and sustainability have direct and indirect implications for building resilient MEC systems. This is particularly crucial for ensuring the dependability of mobile MEC resources (R3) with limited energy sources, such as batteries in UAVs or electric vehicles, as well as for low-power MEC users who cannot afford the energy costs of transmitting large volumes of data or performing local computations.

By optimizing energy consumption, MEC systems reduce the likelihood of resource depletion, which could otherwise lead to service outages. For instance, energy-aware task offloading algorithms can distribute

workloads across MEC systems based on their energy availability, ensuring continuous service delivery [22]. This principle can be extended to task scheduling, where energy constraints impose additional time-critical requirements, where the battery life of MEC resources and users must be considered to complete tasks with high reliability. This is particularly important for UAV-based MEC systems (R3) as they require frequent recharging and redeployment after completing their allocated tasks. Some approaches also utilize energy harvesting and wireless charging for such systems, which is an additional dimension for planning their mobility trajectory [106].

Incorporating energy efficiency adds further complexity to the already challenging problems of task allocation and scheduling. Estimating task completion time, including computation and communication time, has always been crucial to accommodate diverse service requirements (C1). However, energy consumption must also be considered. Here, energy cost of a task depends not only on its computation time but also on its computational intensity and the specifications of the MEC system executing it, such as the capabilities of its I/O devices and processors. AI/ML models can play a pivotal role in modeling these complex relationships between different application types and heterogeneous MEC systems, enabling energy-efficient task allocation and scheduling. Eventually, this can prolong resource availability and improve system resilience.

From an MEC user's perspective, energy consumption is a compromise between communication and computation [65]. Specifically, the energy cost of data transmission for task offloading can outweigh the energy-saving benefits of remote computation when the tasks require heavy data transmissions or the links are unreliable with high packet loss. Energy-aware offloading strategies that balance the amount of local and remote computations, based on the energy cost of data transmission, could help reduce the energy footprint of individual users [114]. Dependability and trustworthiness of MEC systems also interact with energy efficiency. For instance, regardless of its proximity, an unreliable (e.g., due to frequent faults) or untrustworthy MEC system can force a user to re-offload their task to another MEC server or revert to local computation. This introduces additional energy overhead. In this sense,

exploring the interplay between resilience goals (e.g., dependability and trustworthiness) and energy efficiency presents an intriguing research direction.

7

Conclusion

Cloud and edge computing paradigms have significantly transformed the scope of modern applications, enabling resource-constrained devices to perform powerful and computationally intensive tasks by offloading them to virtualized, remote servers. The multi-access (or mobile) edge computing (MEC) paradigm takes this a step further by allowing highly mobile and wirelessly connected users to execute applications in a time-sensitive manner. However, the heterogeneous nature of computational resources, the dynamic demands of users, and the overall unpredictability of the MEC environment due to mobility and the stochastic nature of wireless communication make it challenging to ensure ubiquitous resource availability and deliver high-reliability services. Additionally, the presence of multiple stakeholders who can dynamically join or leave the MEC environment creates a potentially hostile setting, where issues of security, trust, and privacy become paramount. As such, resilience emerges as one of the most pressing concerns in the design and maintenance of MEC systems. In this work, we explored challenges, concepts, and measures necessary for establishing resilient MEC systems.

The first key contribution is the analysis of structure, components, and actors within the MEC ecosystem to better understand their general

requirements. This effort resulted in a comprehensive system model that illustrates the various types of MEC resources, users, and interfaces, as well as their interdependencies. From this foundation, we identified seven key challenges that reflect the heterogeneous, dynamic, and distributed nature of MEC environments. These challenges underscore both the necessity of resilience measures and the difficulties associated with developing them. The proposed system model and challenges allowed us to present a holistic view of the MEC landscape and to define essential resilience goals and techniques. We introduced two primary resilience goals: dependability and trustworthiness. These goals encompass multiple objectives, such as reliability and availability within dependability, and security, trust, and privacy within trustworthiness, but also reflect more classic performance measures like energy efficiency, data rate, and latency. While these objectives helped highlight specific resilience issues, the overarching goals provided a framework to highlight the interdependencies among them. For instance, resource availability and service reliability should be guaranteed together to have a fully functioning and dependable MEC system; they cannot be addressed in isolation. These goals and objectives were carefully selected not only to address general resilience concerns in MEC but also to address gaps in the literature. Additionally, we emphasized the importance of employing both proactive and reactive techniques to achieve these goals effectively.

The second key contribution of this work is the presentation of eight resilience measures. We defined these measures conceptually and provided several examples for each, derived from our extensive literature analysis. These examples are further associated to (i) the challenges they address, (ii) the MEC components and actors they target, (iii) the resilience goals they aim at, and (iv) the techniques they employ. Through these examples, we discussed potential trade-offs and alternative approaches to implementing the proposed measures.

Finally, we outlined directions for designing more comprehensive, intelligent, sustainable, and seamlessly connected resilient MEC systems. We are certain that this work provides researchers with a comprehensive understanding of resilience in MEC, highlighting its challenges and potential solutions based on latest research and technological advancements.

References

[1] S. Ahmad, J. Zhang, A. Khan, U. A. Khan, and B. Hayat, "JO-TADP: Learning-Based Cooperative Dynamic Resource Allocation for MEC-UAV-Enabled Wireless Network," *Drones*, vol. 7, no. 5, 2023. DOI: 10.3390/drones7050303.

[2] B. Ali, M. A. Gregory, S. Li, and O. A. Dib, "Zero Trust Security Framework for 5G MEC Applications: Evaluating UE Dynamic Network Behaviour," in *33rd International Telecommunication Networks and Applications Conference (ITNAC 2023)*, pp. 140–144, Melbourne, Australia: IEEE, 2023. DOI: 10.1109/itnac59571.2023.10368551.

[3] A. Alioua, N. Bouchemal, R. Mati, and M.-L. Messai, "Blockchain-inspired Incentive Mechanism for Trust-aware Offloading in Mobile Edge Computing," in *49th IEEE Conference on Local Computer Networks (LCN 2024)*, pp. 1–8, Caen, France: IEEE, 2024. DOI: 10.1109/lcn60385.2024.10639661.

[4] A. Alwarafy, K. A. Al-Thelaya, M. Abdallah, J. Schneider, and M. Hamdi, "A Survey on Security and Privacy Issues in Edge-Computing-Assisted Internet of Things," *IEEE Internet of Things Journal*, vol. 8, no. 6, 2021, pp. 4004–4022. DOI: 10.1109/jiot.2020.3015432.

[5] M. Andjelkovic, N. Maletic, N. Miglioranza, M. Krstic, E. Koeck, J. Buchholz, M. Taddiken, M. Fehrenz, S. Baradie, D. Wübben, and M. Breitbach, "6G-TakeOff: Holistic 3D Networks for 6G Wireless Communications," in *27th Euromicro Conference on Digital System Design (DSD 2024)*, pp. 435–442, Paris, France: IEEE, 2024. DOI: 10.1109/dsd64264.2024.00064.

[6] P. Arroba, R. Buyya, R. Cárdenas, J. L. Risco-Martín, and J. M. Moya, "Sustainable edge computing: Challenges and future directions," *Software: Practice and Experience*, vol. 54, no. 11, 2024, pp. 2272–2296. DOI: 10.1002/spe.3340.

[7] S. Bagchi, M.-B. Siddiqui, P. Wood, and H. Zhang, "Dependability in edge computing," *Communications of the ACM*, vol. 63, no. 1, 2019, pp. 58–66. DOI: 10.1145/3362068.

[8] A. A. Baktayan, A. Thabit Zahary, and I. Ahmed Al-Baltah, "A Systematic Mapping Study of UAV-Enabled Mobile Edge Computing for Task Offloading," *IEEE Access*, vol. 12, 2024, pp. 101 936–101 970. DOI: 10.1109/access.2024.3431922.

[9] L. Baresi, D. F. Mendonça, M. Garriga, S. Guinea, and G. Quattrocchi, "A Unified Model for the Mobile-Edge-Cloud Continuum," *ACM Transactions on Internet Technology*, vol. 19, no. 2, 2019, pp. 1–21. DOI: 10.1145/3226644.

[10] M. Caprolu, R. D. Pietro, F. Lombardi, and S. Raponi, "Edge Computing Perspectives: Architectures, Technologies, and Open Security Issues," in *IEEE International Conference on Edge Computing (EDGE 2019)*, pp. 116–123, Milan, Italy, 2019. DOI: 10.1109/EDGE.2019.00035.

[11] J. Cheng, D. T. Nguyen, and V. K. Bhargava, "Resilient Edge Service Placement Under Demand and Node Failure Uncertainties," *IEEE Transactions on Network and Service Management*, vol. 21, no. 1, 2024, pp. 558–573. DOI: 10.1109/tnsm.2023.3290137.

[12] N. Cheng, W. Xu, W. Shi, Y. Zhou, N. Lu, H. Zhou, and X. Shen, "Air-Ground Integrated Mobile Edge Networks: Architecture, Challenges, and Opportunities," *IEEE Communications Magazine*, vol. 56, no. 8, 2018, pp. 26–32. DOI: 10.1109/mcom.2018.1701092.

[13] G. Cui, Q. He, F. Chen, H. Jin, Y. Xiang, and Y. Yang, "Location Privacy Protection via Delocalization in 5G Mobile Edge Computing Environment," *IEEE Transactions on Services Computing*, vol. 16, no. 1, 2023, pp. 412–423. DOI: 10.1109/tsc.2021.3112659.

[14] G. Cui, Q. He, B. Li, X. Xia, F. Chen, H. Jin, Y. Xiang, and Y. Yang, "Efficient Verification of Edge Data Integrity in Edge Computing Environment," *IEEE Transactions on Services Computing*, vol. 15, no. 6, 2022, pp. 3233–3244. DOI: 10.1109/TSC.2021.3090173.

[15] J. Cui, L. Wei, H. Zhong, J. Zhang, Y. Xu, and L. Liu, "Edge Computing in VANETs - An Efficient and Privacy-Preserving Cooperative Downloading Scheme," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 6, 2020, pp. 1191–1204. DOI: 10.1109/jsac.2020.2986617.

[16] S. Deng, H. Zhao, W. Fang, J. Yin, S. Dustdar, and A. Y. Zomaya, "Edge Intelligence: The Confluence of Edge Computing and Artificial Intelligence," *IEEE Internet of Things Journal*, vol. 7, no. 8, 2020, pp. 7457–7469. DOI: 10.1109/jiot.2020.2984887.

[17] I. Dhanapala, S. Bharti, A. McGibney, and S. Rea, "Toward a Performance-Based Trustworthy Edge-Cloud Continuum," *IEEE Access*, vol. 12, 2024, pp. 99 201–99 212. DOI: 10.1109/access.2024.3429197.

[18] C. Dong, Y. Tian, Z. Zhou, W. Wen, and X. Chen, "Joint Power Allocation and Task Offloading for Reliability-Aware Services in NOMA-Enabled MEC," *IEEE Transactions on Wireless Communications*, vol. 23, no. 7, 2024, pp. 7537–7551. DOI: 10.1109/twc.2023.3342434.

[19] L. Dong, Q. Ni, W. Wu, C. Huang, T. Znati, and D. Z. Du, "A Proactive Reliable Mechanism-Based Vehicular Fog Computing Network," *IEEE Internet of Things Journal*, vol. 7, no. 12, 2020, pp. 11 895–11 907. DOI: 10.1109/jiot.2020.3007608.

[20] F. Dressler, C. F. Chiasserini, F. H. P. Fitzek, H. Karl, R. Lo Cigno, A. Capone, C. E. Casetti, F. Malandrino, V. Mancuso, F. Klingler, and G. A. Rizzo, "V-Edge: Virtual Edge Computing as an Enabler for Novel Microservices and Cooperative Computing," *IEEE Network*, vol. 36, no. 3, 2022, pp. 24–31. DOI: 10.1109/MNET.001.2100491.

[21] F. Dressler, F. Klingler, M. Segata, and R. Lo Cigno, "Cooperative Driving and the Tactile Internet," *Proceedings of the IEEE*, vol. 107, no. 2, 2019, pp. 436–446. DOI: 10.1109/JPROC.2018.2863026.

[22] M. Falcão, C. B. Souza, A. Balieiro, and K. Dias, "Resource allocation for UAV-enabled multi-access edge computing," *The Journal of Supercomputing*, vol. 80, no. 15, 2024, pp. 22 770–22 802. DOI: 10.1007/s11227-024-06314-3.

[23] E. Fazeldehkordi and T.-M. Grønli, "A Survey of Security Architectures for Edge Computing-Based IoT," *IoT*, vol. 3, no. 3, 2022, pp. 332–365. DOI: 10.3390/iot3030019.

[24] K. Fu, W. Zhang, Q. Chen, D. Zeng, and M. Guo, "Adaptive Resource Efficient Microservice Deployment in Cloud-Edge Continuum," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 8, 2022, pp. 1825–1840. DOI: 10.1109/tpds.2021.3128037.

[25] S. Ghanavati, J. Abawajy, and D. Izadi, "Automata-Based Dynamic Fault Tolerant Task Scheduling Approach in Fog Computing," *IEEE Transactions on Emerging Topics in Computing*, vol. 10, no. 1, 2022, pp. 488–499. DOI: 10.1109/tetc.2020.3033672.

[26] M. Giordani and M. Zorzi, "Non-Terrestrial Networks in the 6G Era: Challenges and Opportunities," *IEEE Network*, vol. 35, no. 2, 2021, pp. 244–251. DOI: 10.1109/mnet.011.2000493.

[27] M. Guo, X. Huang, W. Wang, B. Liang, Y. Yang, L. Zhang, and L. Chen, "HAGP: A Heuristic Algorithm Based on Greedy Policy for Task Offloading with Reliability of MDs in MEC of the Industrial Internet," *Sensors*, vol. 21, no. 10, 2021, p. 3513. DOI: 10.3390/s21103513.

[28] J. Han, I. Yun, S. Kim, T. Kim, S. Son, and D. Han, "Scalable and Secure Virtualization of HSM With ScaleTrust," *IEEE/ACM Transactions on Networking*, vol. 31, no. 4, 2023, pp. 1595–1610. DOI: 10.1109/TNET.2022.3220427.

[29] Y. Harchol, A. Mushtaq, V. Fang, J. McCauley, M. Panda, and S. Shenker, "Making edge-computing resilient," in *11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID 2011)*, pp. 253–266, Newport Beach, CA, 2011. DOI: 10.1145/3419111.3421278.

[30] Q. He, H. Lin, J. Hu, and X. Wang, "A Novel Cross-domain Access Control Protocol in Mobile Edge Computing," in *IEEE Global Communications Conference (GLOBECOM 2021)*, pp. 1–6, Madrid, Spain, 2021. DOI: 10.1109/GLOBECOM46510.2021.9685650.

[31] M. Heydari, A. Mylonas, V. Katos, E. Balaguer-Ballester, V. H. F. Tafreshi, and E. Benkhelifa, "Uncertainty-aware authentication model for fog computing in IoT," in *4th International Conference on Fog and Mobile Edge Computing (FMEC 2019)*, pp. 52–59, Rome, Italy, 2019. DOI: 10.1109/FMEC45842.2019.

[32] A. Al-Hourani, S. Kandeepan, and S. Lardner, "Optimal LAP Altitude for Maximum Coverage," *IEEE Wireless Communications Letters*, vol. 3, no. 6, 2014, pp. 569–572. DOI: 10.1109/lwc.2014.2342736.

[33] S. A. Huda and S. Moh, "Survey on computation offloading in UAV-Enabled mobile edge computing," *Elsevier Journal of Network and Computer Applications*, vol. 201, 2022, p. 103 341. DOI: 10.1016/j.jnca.2022.103341.

[34] J.-H. Huh and Y.-S. Seo, "Understanding Edge Computing: Engineering Evolution with Artificial Intelligence," *IEEE Access*, vol. 7, 2019, pp. 164 229–164 245. DOI: 10.1109/ACCESS.2019.2945338.

[35] A. Javed, A. Malhi, and K. Främling, "Edge Computing-based Fault-tolerant Framework: A Case Study on Vehicular Networks," in *International Wireless Communications and Mobile Computing (IWCMC 2020)*, pp. 1541–1548, Limassol, Cyprus, 2020. DOI: 10.1109/IWCMC48107.2020.9148269.

[36] S. Jeuk, G. Salgueiro, F. Baker, and S. Zhou, "Network segmentation in the cloud a novel architecture based on UCC and IID," in *4th IEEE International Conference on Cloud Networking (CloudNet 2015)*, pp. 58–63, Niagara Falls, Canada, 2015. DOI: 10.1109/CloudNet.2015.7335280.

[37] X. Jia, D. He, N. Kumar, and K.-K. R. Choo, "A Provably Secure and Efficient Identity-based Anonymous Authentication Scheme for Mobile Edge Computing," *IEEE Systems Journal*, vol. 14, no. 1, 2020, pp. 560–571. DOI: 10.1109/JSYST.2019.2896064.

[38] C. Jiang, T. Fan, H. Gao, W. Shi, L. Liu, C. Cérin, and J. Wan, "Energy aware edge computing: A survey," *Elsevier Computer Communications*, vol. 151, 2020, pp. 556–580. DOI: 10.1016/j.comcom.2020.01.004.

[39] H. Jiang, H. Chang, S. Mukherjee, and J. Van der Merwe, "OZTrust: An O-RAN Zero-Trust Security System," in *IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN 2023)*, pp. 129–134, Dresden, Germany: IEEE, 2023. DOI: 10.1109/NFV-SDN59219.2023.10329620.

[40] K. Jiang, H. Zhou, X. Chen, and H. Zhang, "Mobile Edge Computing for Ultra-Reliable and Low-Latency Communications," *IEEE Communications Standards Magazine*, vol. 5, no. 2, 2021, pp. 68–75. DOI: 10.1109/MCOMSTD.001.2000045.

[41] A. Jøsang, *Subjective Logic: A Formalism for Reasoning Under Uncertainty*, vol. 3, 1st ed. Springer, 2016.

[42] W. Z. Khan, E. Ahmed, S. Hakak, I. Yaqoob, and A. Ahmed, "Edge Computing: A Survey," *Elsevier Future Generation Computer Systems*, vol. 97, no. C, 2019, pp. 219–235. DOI: 10.1016/j.future.2019.02.050.

[43] L. Kong, J. Tan, J. Huang, G. Chen, S. Wang, X. Jin, P. Zeng, M. Khan, and S. K. Das, "Edge-computing-driven Internet of Things: A Survey," *ACM Computing Surveys*, vol. 55, no. 8, 2022, pp. 1–41. DOI: 10.1145/3555308.

[44] X. Kong, Y. Wu, H. Wang, and F. Xia, "Edge Computing for Internet of Everything: A Survey," *IEEE Internet of Things Journal*, vol. 9, no. 23, 2022, pp. 23 472–23 485. DOI: 10.1109/JIOT.2022.3200431.

[45] I. Korontanis, A. Makris, T. Theodoropoulos, and K. Tserpes, "Real-time Monitoring and Analysis of Edge and Cloud Resources," in *32nd International Symposium on High-Performance Parallel and Distributed Computing (HPDC 2023), 3rd Workshop on Flexible Resource and Application Management on the Edge (FRAME 2023)*, pp. 13–18, Orlando, FL: ACM, 2023. DOI: 10.1145/3589010.3594892.

[46] K. Kumaran and E. Sasikala, "Learning based Latency Minimization Techniques in Mobile Edge Computing (MEC) systems: A Comprehensive Survey," in *International Conference on System, Computation, Automation and Networking (ICSCAN 2021)*, pp. 1–6, Puducherry, India: IEEE, 2021. DOI: 10.1109/ICSCAN53069.2021.9526410.

[47] M. Laroui, B. Nour, H. Moungla, M. A. Cherif, H. Afifi, and M. Guizani, "Edge and Fog Computing for IoT: A Survey on Current Research Activities & Future Directions," *Elsevier Computer Communications*, vol. 180, 2021, pp. 210–231. DOI: 10.1016/j.comcom.2021.09.003.

[48] K. Li, Y. Cui, W. Li, T. Lv, X. Yuan, S. Li, W. Ni, M. Simsek, and F. Dressler, "When Internet of Things meets Metaverse: Convergence of Physical and Cyber Worlds," *IEEE Internet of Things Journal*, vol. 10, no. 5, 2023, pp. 4148–4173. DOI: 10.1109/JIOT.2022.3232845.

[49] Q. Li, S. Meng, S. Zhang, J. Hou, and L. Qi, "Complex Attack Linkage Decision-making in Edge Computing Networks," *IEEE Access*, vol. 7, 2019, pp. 12 058–12 072. DOI: 10.1109/ACCESS.2019.2891505.

[50] W. Li, B. Zhao, L. Zhu, Y. Wang, Q. Haizhong, and S. Yu, "TCEC: Integrity Protection for Containers by Trusted Chip on IoT Edge Computing Nodes," *IEEE Sensors Journal*, 2024. DOI: 10.1109/JSEN.2024.3445576.

[51] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated Learning in Mobile Edge Networks: A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, 2020, pp. 2031–2063. DOI: 10.1109/comst.2020.2986024.

[52] Y. Lin, W. Feng, Y. Wang, Y. Chen, Y. Zhu, X. Zhang, N. Ge, and Y. Gao, "Satellite-MEC Integration for 6G Internet of Things: Minimal Structures, Advances, and Prospects," *IEEE Open Journal of the Communications Society*, vol. 5, 2024, pp. 3886–3903. DOI: 10.1109/OJCOMS.2024.3418860.

[53] H. Liu, P. Zhang, G. Pu, T. Yang, S. Maharjan, and Y. Zhang, "Blockchain Empowered Cooperative Authentication with Data Traceability in Vehicular Edge Computing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, 2020, pp. 4221–4232. DOI: 10.1109/TVT.2020.2969722.

[54] L. Liu, C. Chen, Q. Pei, S. Maharjan, and Y. Zhang, "Vehicular Edge Computing and Networking: A Survey," *ACM/Springer Mobile Networks and Applications*, vol. 26, 2021, pp. 1145–1168. DOI: 10.1007/s11036-020-01624-1.

[55] X. Liu, J. Jiang, and L. Li, "Computation Offloading and Task Scheduling with Fault-Tolerance for Minimizing Redundancy in Edge Computing," in *IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW 2021)*, pp. 198–209, Wuhan, China: IEEE, 2021. DOI: 10.1109/issrew53611.2021.00064.

[56] Y. Liu, X. Xing, Z. Tong, X. Lin, J. Chen, Z. Guan, Q. Wu, and W. Susilo, "Secure and Scalable Cross-Domain Data Sharing in Zero-Trust Cloud-Edge-End Environment Based on Sharding Blockchain," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 4, 2024, pp. 2603–2618. DOI: 10.1109/tdsc.2023.3313799.

[57] T. Long, Y. Ma, Y. Xia, X. Xiao, Q. Peng, and J. Zhao, "A Mobility-Aware and Fault-Tolerant Service Offloading Method in Mobile Edge Computing," in *IEEE International Conference on Web Services (ICWS 2022)*, pp. 67–72, Barcelona, Spain: IEEE, 2022. DOI: 10.1109/icws55610.2022.00024.

[58] Q. Luo, S. Hu, C. Li, G. Li, and W. Shi, "Resource Scheduling in Edge Computing: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, 2021, pp. 2131–2165. DOI: 10.1109/comst.2021.3106401.

[59] H. Ma, S. Li, E. Zhang, Z. Lv, J. Hu, and X. Wei, "Cooperative Autonomous Driving Oriented MEC-Aided 5G-V2X: Prototype System Design, Field Tests and AI-Based Optimization Tools," *IEEE Access*, vol. 8, 2020, pp. 54 288–54 302. DOI: 10.1109/access.2020.2981463.

[60] M. Ma and Z. Wang, "Distributed Offloading for Multi-UAV Swarms in MEC-Assisted 5G Heterogeneous Networks," *Drones*, vol. 7, no. 4, 2023, p. 226. DOI: 10.3390/drones7040226.

[61] P. Maciel, J. Dantas, C. Melo, P. Pereira, F. Oliveira, J. Araujo, and R. Matos, "A survey on reliability and availability modeling of edge, fog, and cloud computing," *Journal of Reliable Intelligent Environments*, vol. 8, no. 3, 2022, pp. 227–245. DOI: 10.1007/s40860-021-00154-1.

[62] M. P. J. Mahenge, C. Li, and C. A. Sanga, "Energy-efficient task offloading strategy in mobile edge computing for resource-intensive mobile applications," *Digital Communications and Networks*, vol. 8, no. 6, 2022, pp. 1048–1058. DOI: 10.1016/j.dcan.2022.04.001.

[63] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, 2017, pp. 2322–2358. DOI: 10.1109/comst.2017.2745201.

[64] M. B. Monir, T. Abdelkader, and E.-S. M. Ei-Horbaty, "Trust Evaluation of Service level Agreement for Service Providers in Mobile Edge Computing," in *9th International Conference on Intelligent Computing and Information Systems (ICICIS 2019)*, pp. 362–369, Cairo, Egypt: IEEE, 2019. DOI: 10.1109/icicis46948.2019.9014854.

[65] Q.-H. Nguyen and F. Dressler, "A Smartphone Perspective on Computation Offloading – A Survey," *Elsevier Computer Communications*, vol. 159, 2020, pp. 133–154. DOI: 10.1016/j.comcom.2020.05.001.

[66] Z. Ouyang, Y. Xia, J. Li, J. Feng, Y. Yu, K. Zhang, X. Xu, Y. Ma, P. Chen, and X. Li, "A Novel Redundant Service Caching and Task Offloading Method in Mobile Edge Computing," in *31st International Conference on Web Services (ICWS 2024)*, pp. 31–46, Bangkok, Thailand: Springer, 2024. DOI: 10.1007/978-3-031-77072-2_3.

[67] T. Park, M. You, J. Kim, and S. Lee, "Fatriot: Fault-tolerant MEC architecture for mission-critical systems using a SmartNIC," *Elsevier Journal of Network and Computer Applications*, vol. 231, 2024, p. 103 978. DOI: 10.1016/j.jnca.2024.103978.

[68] K. Peng, V. Leung, X. Xu, L. Zheng, J. Wang, and Q. Huang, "A Survey on Mobile Edge Computing: Focusing on Service Adoption and Provision," *Wireless Communications and Mobile Computing*, vol. 2018, 2018. DOI: 10.1155/2018/8267838.

[69] K. Peng, B. Zhao, M. Bilal, and X. Xu, "Reliability-Aware Computation Offloading for Delay-Sensitive Applications in MEC-Enabled Aerial Computing," *IEEE Transactions on Green Communications and Networking*, vol. 6, no. 3, 2022, pp. 1511–1519. DOI: 10.1109/tgcn.2022.3162584.

[70] X. Qi and G. Mei, "Network Resilience: Definitions, approaches, and applications," *Journal of King Saud University - Computer and Information Sciences*, vol. 36, no. 1, 2024, p. 101 882. DOI: 10.1016/j.jksuci.2023.101882.

[71] Y. Qiu, J. Niu, X. Zhu, K. Zhu, Y. Yao, B. Ren, and T. Ren, "Mobile Edge Computing in Space-Air-Ground Integrated Networks: Architectures, Key Technologies and Challenges," *Journal of Sensor and Actuator Networks*, vol. 11, no. 4, 2022, p. 57. DOI: 10.3390/jsan11040057.

[72] V. B. Reddy, A. Negi, S. Venkataraman, and V. R. Venkataraman, "A Similarity based Trust Model to Mitigate Badmouthing Attacks in Internet of Things (IoT)," in *5th IEEE World Forum on Internet of Things (WF-IoT 2019)*, pp. 278–282, Luimneach, Ireland: IEEE, 2019. DOI: 10.1109/wf-iot.2019.8767170.

[73] R. Roman, J. Lopez, and M. Mambo, "Mobile edge computing, Fog et al.: A survey and analysis of security threats and challenges," *Elsevier Future Generation Computer Systems*, vol. 78, 2018, pp. 680–698. DOI: 10.1016/j.future.2016.11.009.

[74] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero Trust Architecture," National Institute of Standards and Technology, United States Department of Commerce, NIST Special Publication 800-207, 2020. DOI: 10.6028/nist.sp.800-207.

[75] D. Sabella, A. Reznik, K. R. Nayak, D. Lopez, F. Li, U. Kleber, A. Leadbeater, K. Maloor, S. B. M. Baskaran, L. Cominardi, C. Costa, F. Granelli, V. Gazis, F. Ennesse, X. Gu, F. Naim, and D. Druta, "MEC Security: Status of Standards Support and Future Evolutions," European Telecommunications Standards Institute, Sophia Antipolis, France, ETSI White Paper 46, 2021.

[76] D. Sabella, A. Vaillant, P. Kuure, U. Rauschenbach, and F. Giust, "Mobile-Edge Computing Architecture: The role of MEC in the Internet of Things," *IEEE Consumer Electronics Magazine*, vol. 5, no. 4, 2016, pp. 84–91. DOI: 10.1109/mce.2016.2590118.

[77] A. Samanta, F. Esposito, and T. G. Nguyen, "Fault-Tolerant Mechanism for Edge-Based IoT Networks With Demand Uncertainty," *IEEE Internet of Things Journal*, vol. 8, no. 23, 2021, pp. 16 963–16 971. DOI: 10.1109/jiot.2021.3075681.

[78] Z. Sanaei, S. Abolfazli, A. Gani, and R. Buyya, "Heterogeneity in Mobile Cloud Computing: Taxonomy and Open Challenges," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, 2014, pp. 369–392. DOI: 10.1109/SURV.2013.050113.00090.

[79] P. M. Sánchez Sánchez, A. Huertas Celdrán, N. Xie, G. Bovet, G. Martínez Pérez, and B. Stiller, "FederatedTrust: A solution for trustworthy federated learning," *Elsevier Future Generation Computer Systems*, vol. 152, 2024, pp. 83–98. DOI: 10.1016/j.future.2023.10.013.

[80] D. Satria, D. Park, and M. Jo, "Recovery for overloaded mobile edge computing," *Elsevier Future Generation Computer Systems*, vol. 70, 2017, pp. 138–147. DOI: 10.1016/j.future.2016.06.024.

[81] G. Scopelliti, C. Baumann, and J. T. Mühlberg, "Understanding Trust Relationships in Cloud-Based Confidential Computing," in *IEEE European Symposium on Security and Privacy Workshops (EuroS&PW 2024)*, pp. 169–176, Vienna, Austria, 2024. DOI: 10.1109/EuroSPW61312.2024.00023.

[82] R. Sharma, C. A. Chan, and C. Leckie, "Hybrid Collaborative Architectures For Intrusion Detection In Multi-Access Edge Computing," in *IEEE/IFIP Network Operations and Management Symposium (NOMS 2022)*, pp. 1–7, Budapest, Hungary: IEEE, 2022. DOI: 10.1109/noms54207.2022.9789795.

[83] S. N. Shirazi, A. Gouglidis, A. Farshad, and D. Hutchison, "The Extended Cloud: Review and Analysis of Mobile Edge Computing and Fog From a Security and Resilience Perspective," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, 2017, pp. 2586–2595. DOI: 10.1109/jsac.2017.2760478.

[84] R. Singh and S. S. Gill, "Edge AI: A survey," *Internet of Things and Cyber-Physical Systems*, vol. 3, 2023, pp. 71–92. DOI: 10.1016/j.iotcps.2023.02.004.

[85] J. P. Sterbenz, D. Hutchison, E. K. Çetinkaya, A. Jabbar, J. P. Rohrer, M. Schöller, and P. Smith, "Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines," *Elsevier Computer Networks*, vol. 54, no. 8, 2010, pp. 1245–1265. DOI: 10.1016/j.comnet.2010.03.005.

[86] H. Sun, H. Yu, G. Fan, and L. Chen, "QoS-Aware Task Placement With Fault-Tolerance in the Edge-Cloud," *IEEE Access*, vol. 8, 2020, pp. 77 987–78 003. DOI: 10.1109/access.2020.2977089.

[87] L. Tang, B. Tang, L. Tang, F. Guo, and J. Zhang, "Reliable Mobile Edge Service Offloading Based on P2P Distributed Networks," *Symmetry*, vol. 12, no. 5, 2020, p. 821. DOI: 10.3390/sym12050821.

[88] Y. Tao, S. Chen, C. Zhang, D. Wang, D. Yu, X. Cheng, and F. Dressler, "Private Over-the-Air Federated Learning at Band-Limited Edge," *IEEE Transactions on Mobile Computing*, vol. 23, no. 12, 2024, pp. 12 444–12 460. DOI: 10.1109/TMC.2024.3411295.

[89] K. Tsampiras, A. Liontos, and V. Tenentes, "Evaluating Trusted Firmware Remote Attestation on ARM and RISC-V Edge Computing Prototypes," in *13th International Conference on Modern Circuits and Systems Technologies (MOCAST 2024)*, pp. 1–4, Sofia, Bulgaria, 2024. DOI: 10.1109/MOCAST61810.2024.10615972.

[90] S. Tuli, G. Casale, and N. R. Jennings, "PreGAN: Preemptive Migration Prediction Network for Proactive Fault-Tolerant Edge Computing," in *41st IEEE International Conference on Computer Communications (INFOCOM 2022)*, pp. 670–679, Virtual Conference: IEEE, 2022. DOI: 10.1109/infocom48880.2022.9796778.

[91] S. Tuli, S. R. Poojara, S. N. Srirama, G. Casale, and N. R. Jennings, "COSCO: Container Orchestration Using Co-Simulation and Gradient Based Optimization for Fog Computing Environments," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 1, 2022, pp. 101–116. DOI: 10.1109/tpds.2021.3087349.

[92] C. Wang, C. Gill, and C. Lu, "FRAME: Fault Tolerant and Real-Time Messaging for Edge Computing," in *39th IEEE International Conference on Distributed Computing Systems (ICDCS 2019)*, pp. 976–985, Dallas-Fort Worth, TX: IEEE, 2019. DOI: 10.1109/icdcs.2019.00101.

[93] C. Wang, Z. Yuan, P. Zhou, Z. Xu, R. Li, and D. O. Wu, "The Security and Privacy of Mobile-Edge Computing: An Artificial Intelligence Perspective," *IEEE Internet of Things Journal*, vol. 10, no. 24, 2023, pp. 22 008–22 032. DOI: 10.1109/jiot.2023.3304318.

[94] J. Wang, M. Wang, Z. Zhang, and H. Zhu, "Toward a Trust Evaluation Framework Against Malicious Behaviors of Industrial IoT," *IEEE Internet of Things Journal*, vol. 9, no. 21, 2022, pp. 21 260–21 277. DOI: 10.1109/jiot.2022.3179428.

[95] K. Wang, X. Wang, and X. Liu, "A High Reliable Computing Offloading Strategy Using Deep Reinforcement Learning for IoVs in Edge Computing," *Journal of Grid Computing*, vol. 19, no. 2, 2021. DOI: 10.1007/s10723-021-09542-6.

[96] L. Wang, S. Chen, F. Chen, Q. He, and J. Liu, "B-Detection: Runtime Reliability Anomaly Detection for MEC Services With Boosting LSTM Autoencoder," *IEEE Transactions on Mobile Computing*, vol. 23, no. 4, 2024, pp. 2599–2613. DOI: 10.1109/tmc.2023.3262233.

[97] R. Wang, N. Chen, X. Yao, and L. Hu, "FASDQ: Fault-Tolerant Adaptive Scheduling with Dynamic QoS-Awareness in Edge Containers for Delay-Sensitive Tasks," *Sensors*, vol. 21, no. 9, 2021, p. 2973. DOI: 10.3390/s21092973.

[98] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A Survey on Mobile Edge Networks: Convergence of Computing, Caching and Communications," *IEEE Access*, vol. 5, 2017, pp. 6757–6779. DOI: 10.1109/ACCESS.2017.2685434.

[99] X. Wang, B. Wang, Y. Wu, Z. Ning, S. Guo, and F. R. Yu, "A Survey on Trustworthy Edge Intelligence: From Security and Reliability To Transparency and Sustainability," *IEEE Communications Surveys & Tutorials*, 2024. DOI: 10.1109/comst.2024.3446585.

[100] Y. Wang and J. Zhao, "A Survey of Mobile Edge Computing for the Metaverse: Architectures, Applications, and Challenges," in *8th IEEE International Conference on Collaboration and Internet Computing (CIC 2022)*, pp. 1–9, Atlanta, GA: IEEE, 2022. DOI: 10.1109/cic56439.2022.00011.

[101] Y. Wang, H. Wang, S. Chen, and Y. Xia, "A Survey on Mainstream Dimensions of Edge Computing," in *5th International Conference on Information System and Data Mining (ICISDM 2021)*, pp. 46–54, Virtual Conference: ACM, 2021. DOI: 10.1145/3471287.3471295.

[102] Z. Wang, J. Liu, and W. Zhu, "Edge Intelligence-Empowered Immersive Media," *IEEE MultiMedia*, vol. 30, no. 2, 2023, pp. 8–17. DOI: 10.1109/mmul.2023.3247574.

[103] Z. Wang, Y. Sun, D. Liu, J. Hu, X. Pang, Y. Hu, and K. Ren, "Location Privacy-Aware Task Offloading in Mobile Edge Computing," *IEEE Transactions on Mobile Computing*, vol. 23, no. 3, 2024, pp. 2269–2283. DOI: 10.1109/tmc.2023.3254553.

[104] Z. Wang, W. Zhao, P. Hu, X. Zhang, L. Liu, C. Fang, and Y. Sun, "UAV-Assisted Mobile Edge Computing: Dynamic Trajectory Design and Resource Allocation," *Sensors*, vol. 24, no. 12, 2024, p. 3948. DOI: 10.3390/s24123948.

[105] D. Wu, G. Shen, Z. Huang, Y. Cao, and T. Du, "A Trust-Aware Task Offloading Framework in Mobile Edge Computing," *IEEE Access*, vol. 7, 2019, pp. 150 105–150 119. DOI: 10.1109/access.2019.2947306.

[106] X. Xia, S. M. M. Fattah, and M. A. Babar, "A Survey on UAV-Enabled Edge Computing: Resource Management Perspective," *ACM Computing Surveys*, vol. 56, no. 3, 2023, pp. 1–36. DOI: 10.1145/3626566.

[107] Y. Xiao, Y. Jia, C. Liu, X. Cheng, J. Yu, and W. Lv, "Edge Computing Security: State of the Art and Challenges," *Proceedings of the IEEE*, vol. 107, no. 8, 2019, pp. 1608–1631. DOI: 10.1109/jproc.2019.2918437.

[108] Y. Yang, Y. Hu, and M. C. Gursoy, "Reliability-Optimal Designs in MEC Networks with Finite Blocklength Codes and Outdated CSI: (Invited Paper)," in *17th IEEE International Symposium on Wireless Communication Systems (ISWCS 2021)*, pp. 1–6, Berlin, Germany: IEEE, 2021. DOI: 10.1109/iswcs49558.2021.9562237.

[109] Z. Yin, C. Wu, C. Guo, Y. Li, M. Xu, W. Gao, and C. Chi, "A comprehensive survey of orbital edge computing: Systems, applications, and algorithms," *Chinese Journal of Aeronautics*, 2024. DOI: 10.1016/j.cja.2024.11.026.

[110] H. Zeyu, X. Geming, W. Zhaohang, and Y. Sen, "Survey on Edge Computing Security," in *International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE 2020)*, pp. 96–105, Fuzhou, China, 2020. DOI: 10.1109/ICBAIE49996.2020.00027.

[111] J. Zhang, B. Chen, Y. Zhao, X. Cheng, and F. Hu, "Data Security and Privacy-Preserving in Edge Computing Paradigm: Survey and Open Issues," *IEEE Access*, vol. 6, 2018, pp. 18 209–18 237. DOI: 10.1109/access.2018.2820162.

[112] L. Zhang, H. Guo, X. Zhou, and J. Liu, "Trusted Task Offloading in Vehicular Edge Computing Networks: A Reinforcement Learning Based Solution," in *IEEE Global Communications Conference (GLOBECOM 2023)*, pp. 6711–6716, Kuala Lumpur, Malaysia: IEEE, 2023. DOI: 10.1109/globecom54140.2023.10437191.

[113] P. Zhang, H. Jin, H. Dong, W. Song, and A. Bouguettaya, "Privacy-Preserving QoS Forecasting in Mobile Edge Environments," *IEEE Transactions on Services Computing*, vol. 15, no. 2, 2022, pp. 1103–1117. DOI: 10.1109/tsc.2020.2977018.

[114] X. Zhang and S. Debroy, "Energy Efficient Task Offloading for Compute-intensive Mobile Edge Applications," in *IEEE International Conference on Communications (ICC 2020)*, pp. 1–6, Virtual Conference: IEEE, 2020. DOI: 10.1109/icc40277.2020.9149012.

[115] X. Zhang and S. Debroy, "Resource Management in Mobile Edge Computing: A Comprehensive Survey," *ACM Computing Surveys*, vol. 55, no. 13, 2023, pp. 1–37. DOI: 10.1145/3589639.