POLITECNICO DI TORINO
Repository ISTITUZIONALE

Securing the Cloud Continuum: A Communication Management Perspective

Original
Securing the Cloud Continuum: A Communication Management Perspective / Pizzato, F., Bringhenti, D., Sisto, R., Valenza, F.. - In: COMPUTER. - ISSN 0018-9162. - ELETTRONICO. - 59:10(2026), pp. 26-35. [10.1109/MC.2026.3703844]

Availability:
This version is available at: 11583/3011927 since: 2026-09-25T07:45:08Z

Publisher:
IEEE

Published
DOI:10.1109/MC.2026.3703844

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2026 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

28 November 2026

Securing the Cloud Continuum: a communication management perspective

Francesco Pizzato, *Politecnico di Torino, Turin, 10129, Italy*
Daniele Bringhenti, *Politecnico di Torino, Turin, 10129, Italy*
Riccardo Sisto, *Politecnico di Torino, Turin, 10129, Italy*
Fulvio Valenza, *Politecnico di Torino, Turin, 10129, Italy*

Abstract—The computing continuum paradigm is set to shape the future of distributed systems by seamlessly integrating cloud, edge, and IoT infrastructures. This paradigm enables dynamic workload distribution across heterogeneous environments, improving scalability, latency, and resource utilization compared to cloud-centric models. However, the nature of the computing continuum introduces security management challenges that existing mechanisms do not address. Ensuring network-level workload isolation across heterogeneous layers and administrative domains remains a significant open problem. This paper investigates this challenge from a communication management perspective and proposes a structured classification of continuum communication patterns, along with an intent-based blueprint for a secure, robust, and scalable solution.

The computing continuum is an emerging paradigm driving the latest innovations in the broader field of cloud computing. Rather than simply extending cloud capabilities to edge and IoT devices, it unifies computation, network, and storage resources across all layers, from data centers to the far edge, into a seamless operational fabric [1]. This homogenizing abstraction transforms application deployment by removing the constraint of isolated data centers and enabling dynamic workload orchestration across heterogeneous environments. As a result, the continuum promises improved resiliency, seamless workload migration, and continuous optimization in terms of latency, costs, and energy consumption [2], [3].

While powerful, the computing continuum conceals significant complexity, introducing several security management challenges that go beyond those addressed in traditional cloud settings. Specifically, network-level workload isolation becomes significantly more complex when workload units deployed by different participants move across heterogeneous infrastructures owned by different administrative domains. Ensuring logical separation in such environments is essential to prevent lateral movements, cross-tenant interference, and data leakage, yet existing isolation mechanisms were not designed for this multi-domain, highly dynamic context.

Several factors contribute to this complexity. First, the extreme heterogeneity of the infrastructure components and security primitives (e.g., Kubernetes Network Policies, access-control lists) significantly increases the complexity of consistent security configuration across the continuum. Second, workloads frequently migrate across the continuum's fabrics, and resources are dynamically borrowed or lent among participants, making it unclear where and how isolation should be enforced. Third, the continuum's scale and dynamism make traditional approaches based on manual, static configuration impractical, leading to misconfiguration errors and weak protections. Lastly, multiple participants, each with distinct and potentially conflicting isolation requirements, coexist within the continuum, requiring dedicated mechanisms to reconcile different security objectives.

Existing research partially addresses these aspects. Several solutions leverage existing technologies, such as service meshes and zero-trust, to address workload isolation. However, these approaches

XXXX-XXX © 2021 IEEE
Digital Object Identifier 10.1109/XXX.0000.0000000

do not accommodate the nature of the continuum, as service meshes introduce overheads that are incompatible with edge and IoT devices, and zero-trust architectures focus on single-domain authentication and authorization flows rather than reconciling potentially conflicting communication requirements across independently governed infrastructures [4], [5]. To this end, Intent-Based Systems (IBSs) provide abstraction mechanisms and autonomic loops to translate high-level goals into configurations, thus coping with heterogeneity and dynamicity [6], [7]. However, IBSs also primarily assume a single administrative domain and do not explicitly address conflicting isolation goals across multiple participants. Similarly, studies on cross-tenant cloud security focus on access control and permissions to allocate new resources, rather than on the management of communications among workload units once deployed [8], [9], [10]. Moreover, recent proposals adapting IBSs to the computing continuum remain security-unaware or limit conflict analysis to goals defined by a single tenant defining incompatible goals (e.g., latency and availability) [11].

This gap motivates the need for an analysis of network-level workload isolation across the continuum and for the definition of guidelines that foster new IBSs explicitly designed to address its challenging nature. To this end, this paper addresses these existing problems and their associated challenges with the following contributions:

- an in-depth analysis of the problem of network-level workload isolation in the computing continuum, including a classification of communication patterns and the identification of the key design dimensions governing their management;
- a blueprint for a secure, robust, and scalable communication management framework tailored to the computing continuum's requirements, derived from the previous analysis.

Computing Continuum

While the term "computing continuum" is gaining traction, its interpretation varies, and there is not yet a uniquely accepted definition of this term. This paper adopts the definition coordinated under the EUCEI initiative[1], which advances the Cloud-Edge-IoT (CEI) Continuum, fostering cooperation among various stakeholders. This vision aims to overcome the fragmentation of current cloud deployments, advocating for a homogenized virtual computing space. Rather than merely interconnecting resources, the continuum creates a transparent and unified network, data, and computation fabric. Applications operate within this fabric without any perceived difference regarding geographical location or resource ownership, dynamically migrating to the optimal execution environment based on runtime conditions or user-defined policies.

The Resource Sharing Vision

A key aspect of this ambitious vision is its usage model. Participants, i.e., individuals or organizations, contribute to the continuum as resource consumers or providers, as part of a dynamic resource-sharing ecosystem. Within this space, dynamic relationships form among participants as these resources are acquired: some participants provide resources, others buy them, and execute workloads on the rented remote resources. By creating and tearing down relationships on demand, this model departs from static federation and enables more agile infrastructure utilization. For instance, a participant with underutilized on-premise infrastructure can rent resources to another participant facing a temporary need, e.g., traffic spike or scheduled maintenance, enabling the first to cover part of infrastructure management costs, and the second to offload computation without permanent physical expansion.

Several projects are developing powerful abstractions to transparently handle the underlying complexity of this "resource sharing" vision[2], including network and data fabrics offering mechanisms such as transparent cross-cluster connectivity, resolution of overlapping address spaces, and location-independent data access. Currently, there is no established standard technology for implementing the computing continuum. However, to facilitate a general analysis of the topic, this work uses the term "*computing continuum enabler*" to refer to any technological stack that supports this usage model.

The involved subjects

To ensure clarity and consistency, the following definitions specify how key terms, with a possible ambiguous meaning, are interpreted within the context of the computing continuum.

- *Participant* - an entity (e.g., organization, application owner, user) that operates workloads

[1] https://eucloudedgeiot.eu/

[2] Liqo (liqo.io), FLUIDOS (fluidos.eu), NEMO (meta-os.eu)

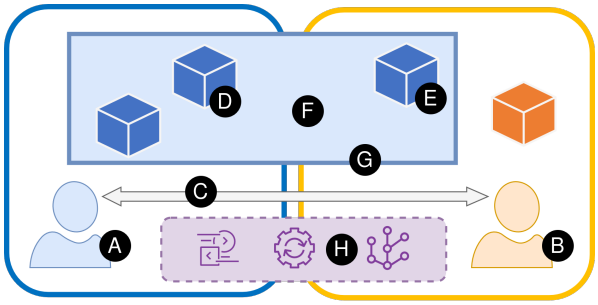

FIGURE 1: Graphical representation of the subjects involved in the computing continuum.

within the shared infrastructure of the computing continuum, or owns a share of the continuum's infrastructure, made available to others.

- *Consumer* - role taken by a participant when actively requesting and using resources offered by another participant owning them.
- *Provider* - role taken by a participant offering resources for rent to be bought and consumed by others.
- *Peering* - relationship established by connecting two entities, typically clusters, in the computing continuum. This results from a negotiation process in which a consumer discovers a provider and acquires resources, interconnecting the local infrastructure with the rented remote one.
- *Offloading* - deployment of a workload, or part of it, to a remote entity within the continuum via an established peering.
- *Workload unit(s)* - the smallest possible unit(s) of work managed within the computing continuum (e.g., Pods in Kubernetes or Tasks in Docker Swarm) and subject to offloading. This is the smallest granularity considered for an endpoint of a network communication.
- *Virtual Cluster* - logical computing environment resulting from the composition of a consumer's local resources and remote resources rented from one or more providers.
- *Virtual Cluster Border* - logical perimeter of the virtual cluster, which might span across multiple physical clusters.

Figure 1 illustrates these concepts in a simple scenario involving two physical clusters, identified by the two rounded-corner boxes, belonging to two participants, distinguished with blue and orange colors. In this scenario, Participant A acts as a consumer requiring to buy some resources, while Participant B is a provider willing to rent excess resources. Therefore, A contacts B to negotiate and rent some of her resources, resulting in the creation of a new peering relationship C. Then, A deploys some workload units (e.g., D), represented with colored cubes, in her own physical cluster, and offloads another one (i.e., E) to the remote resources of the provider's cluster. Consequently, A's logical computing environment expands to include part of B's physical cluster, forming the virtual cluster F, represented with a light-blue box. Its border G spans two physical clusters. The computing continuum enabler H, represented with a light-purple box, manages the necessary abstractions, including the network and data fabrics required for transparent resource consumption over the virtual cluster.

Communications in the Continuum

A significant security challenge in the computing continuum is network-level workload isolation. As workload units are offloaded from the consumer's cluster to the provider's one, multiple participants are involved, making the definition of valid communication paths among workload units and external entities non-trivial [12]. To systematically reason about this problem, we model communication management along two orthogonal dimensions:

- *Communication Control Authority*. This dimension captures which participant has the right to specify requirements or restrictions on communications involving workload units she owns, independently of their location within the continuum. When communications involve workload units owned by different participants, their requirements may conflict.
- *Low-level Enforcement Responsibility*. This dimension concerns which participant is technically responsible for the application of isolation primitives to block or allow communications, such as Kubernetes Network Policies or service meshes. Enforcement depends on the workload unit's location within the continuum, falling under the responsibility of who manages the physical clusters where they are deployed. Therefore, it may not coincide with control authority.

The separation between authority and enforcement naturally fits the continuum, where ownership of workloads and ownership of infrastructure do not necessarily overlap. Moreover, this distinction aligns with established architectural patterns, such as Zero Trust, which separates the Policy Enforcement Point (PEP) and the Policy Decision Point (PDP), or Intent-Based Networking (IBN), which distinguishes intents from configuration [13], [6].

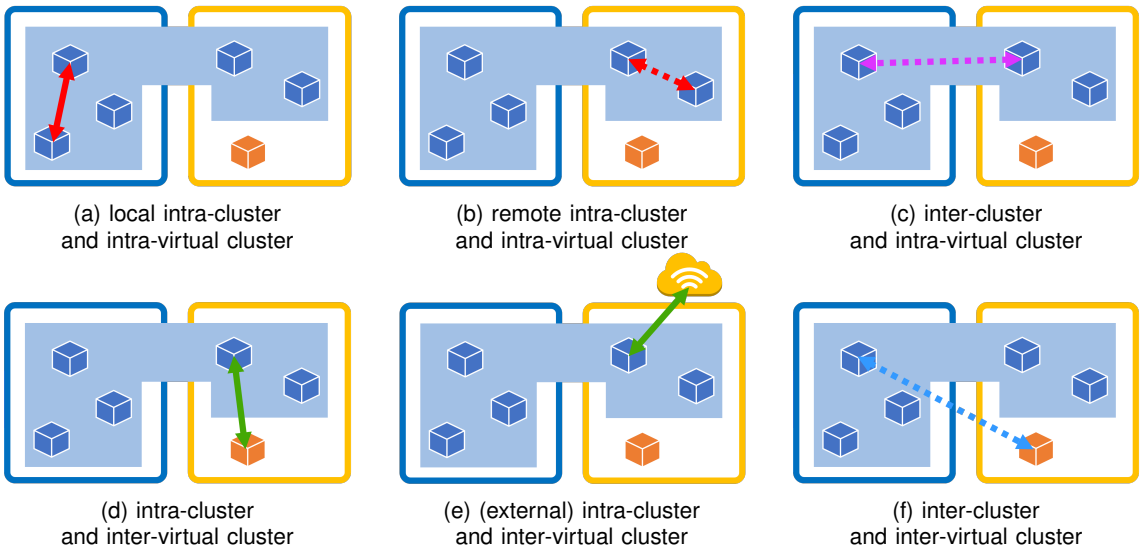

FIGURE 2: Communication types in the continuum

Based on this two-dimensional model, we classify the possible communication types in the computing continuum context. The six identified communication types are represented in the sub-figures of Figure 2, sharing the same exemplifying continuum scenario but focusing on different aspects. Specifically, the scenario comprises two physical clusters: the blue one owned by the consumer, the yellow one by the provider. A virtual cluster spans both physical clusters and contains five blue workload units belonging to the consumer, of which three are hosted in the consumer's local cluster, and the other two are offloaded to the remote one. The provider's cluster also includes a yellow workload unit, which is not part of the virtual cluster. Instead, the communications on which each sub-figure focuses are represented by arrows.

The identified communication types can be classified into two macro-categories: intra-virtual cluster and inter-virtual cluster communications.

Intra-virtual cluster communications

These communications occur between workload units belonging to the same virtual cluster and do not cross its virtual border. In this case, the consumer holds control authority as the communicating units belong to a workload she owns. However, enforcement responsibility depends on the units' location.

In particular, for any pair of units belonging to the same virtual cluster, between which the consumer wants to allow or block a communication, three different positioning schemes exist.

1) Both units are located in the consumer's local physical cluster (Figure 2a). The workload isolation primitives are directly enforced by the consumer.
2) Both units are located in the provider's remote physical cluster (Figure 2b). The consumer retains control authority but must rely on the provider for enforcement, as the provider is responsible for the underlying infrastructure.
3) One unit is located in the consumer's physical cluster, the other in the provider's one (Figure 2c). In this case, enforcement responsibility is shared, requiring coordination between consumer and provider to ensure consistent policy application.

Beware that for those scenarios in which communication control authority and enforcement responsibility do not coincide, participants have to trust that the other party is indeed enforcing what is requested. In this sense, multiple strategies have already been studied for implementing trust-based mechanisms in similar use cases, based on formal cross-tenant trust models and automated trust negotiation frameworks grounded in reputation scores [14], [15]. Complementarily, when a consumer delegates enforcement to a provider, remote attestation mechanisms can produce cryptographic evidence that the consumer can verify. In summary, investigating how to integrate already established remote attestation schemas and solutions represents an orthogonal research direction, aiming to strengthen assurance in the computing continuum [16].

TABLE 1: Summary of the main characteristics of the analyzed communication types

Communication Type	Communication Control Authority	Low-Level Enforcement Responsibility	Examples of Enforcement Primitives
Local intra-cluster intra-virtual cluster (Figure 2a)	Consumer	Consumer	Kubernetes Network Policies, VPC rules, AWS/Azure Security Groups, iptables/nftables rules
Remote intra-cluster intra-virtual cluster (Figure 2b)	Consumer	Provider	Kubernetes Network Policies, VPC rules, AWS/Azure Security Groups, iptables/nftables rules
Inter-cluster intra-virtual cluster (Figure 2c)	Consumer	Shared	Service Mesh (e.g., Istio, Linkerd), inter-VPC peering policies (e.g., AWS VPC Peering), private links (e.g., AWS PrivateLink, Azure Private Link)
Intra-cluster inter-virtual cluster (Figure 2d)	Shared	Provider	Kubernetes Network Policies, VPC rules, AWS/Azure Security Groups, iptables/nftables rules
(External) intra-cluster inter-virtual cluster (Figure 2e)	Shared	Provider	Kubernetes Network Policies, iptables/nftables rules, ingress/egress cloud-native gateways, web proxy rules (e.g., Squid Proxy)
Inter-cluster inter-virtual cluster (Figure 2f)	Shared	Shared	Service Mesh (e.g., Istio, Linkerd), inter-VPC peering policies (e.g., AWS VPC Peering), private links (e.g., AWS PrivateLink, Azure Private Link)

Inter-virtual cluster communications

These communications involve a workload unit of a virtual cluster and an external entity (e.g., a unit controlled by a different participant or an endpoint on the Internet), and consequently cross the virtual cluster border. Their control authority is inherently shared between consumer and provider, since the communication affects resources under different ownership domains, and conflicts between their goals may arise. Specifically, on the one hand, the consumer has the right to manage communications for her workload units. On the other hand, the provider, who manages the physical resources where the consumer's units are offloaded, has the right to establish how those units can communicate with elements not owned by the consumer. Instead, low-level enforcement depends solely on unit positions.

Three positioning schemes are identified:

1) Both units are located in the provider's remote physical cluster, but one belongs to the consumer's virtual cluster while the other is external and under the provider's control (Figure 2d). In this scenario, the two participants may have different management needs. For example, the consumer may request connectivity for an offloaded unit (e.g., to access a provider-hosted database), but the provider may restrict it due to service-level agreements or licensing issues. Therefore, control authority is shared and may be a possible source of conflicts. In any case, the provider is responsible for enforcing workload isolation primitives.
2) A consumer's unit is offloaded to the provider's cluster, while the other entity is on the Internet (Figure 2e). In this scenario, the consumer may request that the offloaded unit contact the remote endpoint, e.g., to fetch updates from a package repository or send telemetry to a third-party monitoring system. Instead, the provider may decide to restrict such Internet access if she does not trust the external entity, as the traffic coming from it would actually reach her physical cluster. Also in this case, control authority is shared, and the provider is responsible for low-level isolation enforcement.
3) A unit is located in the consumer's physical cluster, the other in the provider's cluster, but they are controlled by different participants (Figure 2f). This scenario is the least typical of the computing continuum, because it also appears in traditional cloud environments, when multiple tenants or organizational units operate within federated infrastructures. It can also be reconducted to the case of Figure 2e by offloading the consumer's workload unit to the provider's cluster.

Analysis Summary: Table 1 provides a concise view on the main characteristics of the management process for analyzed communication types, listing possible examples of low-level enforcement primitives.

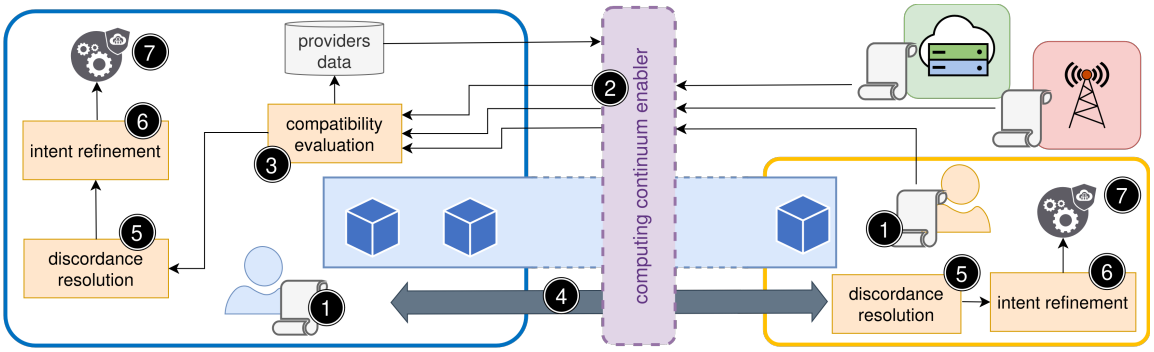

FIGURE 3: blueprint of network-level workload isolation solution for the continuum

Blueprint: an intent-based approach

To address the workload isolation challenges analyzed above, we propose a foundational blueprint grounded in intent-based management practice and adapted to the multi-administrative nature of the computing continuum. Some aspects of this blueprint are partially inspired by the authors' preliminary study in this field, i.e., an initial solution to partially solve this problem within the use cases of one of the above-mentioned European projects [17]. Figure 3 depicts the envisioned blueprint, highlighting its main phases.

Although the blueprint is presented as a sequence of phases, the computing continuum is inherently dynamic, as isolation requirements and infrastructure conditions evolve over time. Therefore, these phases should be re-triggered when monitoring detects deviations or when participants submit updated intents.

As discussed for intra-virtual cluster communications, trust is also required whenever control authority and enforcement responsibility diverge. When enforcement is delegated, the decisions and configurations produced by these phases should therefore be backed by verifiable evidence, such as remote attestation [16], or trust negotiation frameworks, such as [14], [15].

Intent specification

Initially, consumers and providers express their communication control authority throughout intents (1). These statements are specified with high-level, user-friendly languages, abstracting from the technologies actually used for virtual cluster management. They also must be highly expressive to provide fine-grained identification of the traffic classes on which their authority is expressed. Examples of selectors for traffic identification include IP 5-tuple fields or workload unit labels.

Following reference documentation, the two key elements for intent classification are the intent user, i.e., who issues the intent to the IBS, and the intent solutions, i.e., the specific scenarios to which the intent applies [7]. Based on the previous analysis, the involved intent users are consumers and providers who define intents for the computing continuum across the possible scopes identified by the communication types. As a result, the following two intent types are envisioned:

- Intents for intra-virtual cluster communication management have connectivity scope, i.e., specify which workload units should communicate and how, and are specified by the consumer, who holds control authority over her workloads, and must be accepted by the hosting provider.
- Intents for inter-virtual cluster communication management may be specified by both consumer and provider. The former defines connectivity-scoped intents, to express her desires about how workload units of her virtual cluster should communicate with external entities. The latter employs security-scoped intents, to restrict the consumer's requests, or to make impositions (e.g., for monitoring).

This dual-intent model captures both desired communications and admissible communications, enabling explicit reasoning over potential conflicts. Intent dissemination between the consumer and a set of potential providers should be managed externally to the IBS, ideally through an abstraction provided by the computing continuum enabler (2) [6]. The scientific literature proposes several approaches that commonly involve a centralized component where collection and processing occur [18]. However, the marketplace usage model for the continuum better suits a hierarchical approach with different levels of resource brokerage and an intent negotiation protocol [19].

Compatibility evaluation

The compatibility evaluation phase (3) determines whether a consumer's requests can be satisfied under a provider's authorizations. This operation is performed by the consumer on the intents received from each possible provider, to identify which options best meet her needs. Since consumers and providers have different isolation goals, the provider might not accept some inter-virtual cluster communications requested by the consumer, or some connections imposed by the provider might be incompatible with the consumer's will. Formally, this can be interpreted as an intent-analysis problem, where the permitted communication set defined by the provider must include the communication set requested by the consumer.

Several evaluation strategies are possible, such as a binary compatibility measurement or a more complex scoring system to rank all possible providers depending on how closely the consumer's needs match the provider's authorizations. Research in this area is not well developed, leaving space for several approaches exploring both architectural choices, like a centralized and trusted mediator service, and optimization problems, for instance exploring the combination of network isolation with others features such as carbon emission, latency, or cost, mutating the compatibility evaluation to a multi-optimization problem with possibly conflicting optimization goals [10], [11]. Whatever the approach, the result is the selection of a specific provider and the establishment of a peering relationship (4).

Discordance resolution

Once the consumer has selected a provider and a peering has been created, the next phase is discordance resolution (5). Compatibility evaluation assesses whether consumers' isolation needs fit providers' authorizations. Then, discordance resolution removes remaining conflicts and produces conflict-free intents.In this context, conflicts do not necessarily arise from condition or action mismatch, but rather from a misalignment between connectivity intents, i.e., the consumer's requests for a service, and security intents, i.e., the provider's constraints on what is allowed. The problem of reconciliating such intents has already been encountered in similar contexts and has been shown to be computationally demanding [20]. Therefore, discordance resolution is executed only once a provider has been selected to avoid wasted computation.

Based on the previous analysis, discordance resolution follows two guiding principles: participants have authority over the resources they own or rent, and providers retain authority over their infrastructure. This second principle creates most conflicts, since inter-virtual cluster communications occur under communication control authority of both consumer and provider. More specifically, two main categories exist:

- A consumer defines intents for inter-virtual cluster communications which are not authorized by the provider;
- A provider defines security intents imposing additional inter-virtual cluster communications which are not explicitly requested by the consumer.

Concerning the first discordance type, the provider's authority prevails. The relationship between the two is limited to the acquired resources; any communication beyond this boundary may be rightfully blocked by the provider. For example, a consumer may offload a workload unit and request egress connectivity for that unit to an external package repository or telemetry endpoint. If the provider's intents prohibit Internet egress from offloaded units, or explicitly mark that destination as unauthorized, e.g., an untrusted package repository, the request exceeds the provider's admissible policy and the communication is denied.

Concerning the second discordance type, deciding which authority should prevail is less immediate, since the provider may impose additional inter-virtual cluster communications not explicitly requested by the consumer. For example, the provider may require a monitoring agent to reach offloaded units to collect logs or accounting data. These communications may be legitimate for operating and securing the acquired resources, but may also conflict with the consumer's isolation preferences. Therefore, multiple resolution strategies may be envisioned. In extreme cases, the authority of a party prevails over the other. If the provider's authority prevails, the consumer is forced to accept some restrictions. Otherwise, the consumer may refuse the provider and opt for a new one. More balanced strategies may instead rely on cost-function optimization via autonomous reasoning, incentivizing schemas, or negotiation protocols to iteratively revise the intents [8], [9].

Intent refinement and orchestration

Once the intents have been harmonized, they must be refined into the configuration for the low-level communication management primitives (6). In IBN terminology, these operations correspond to intent refinement and orchestration [6]. Importantly, refinement in IBSs is not merely a syntactic decomposition of high-level goals into lower-level policies, but it may include algorithms

and heuristics that select among multiple feasible realizations and optimize outcomes under dynamic conditions.

In the computing continuum, this step must account for the heterogeneity of the enforcement mechanism (e.g., Kubernetes Network Policies, virtual private cloud controls, service mesh rules). Moreover, differences in semantics and capabilities are common within the same type of primitive. For instance, the Kubernetes Network Policy behavior highly depends on the container network interface in use, with some of them only partially supporting them and, in some cases, silently ignoring their enforcement [12].

Beyond refinement, this phase also includes intent orchestration. Depending on the communication type and on the enforcement responsibility, the primitive may be enforced on the consumer side, the provider side, or both. The resulting configurations are enforced by communicating them to the orchestration components in charge, e.g., the Kubernetes API server (7).

Conclusion: Future research directions

This paper provided an in-depth analysis of the network-level workload isolation problem, and proposed a blueprint for a secure automatic mechanism to address it. Even if the envisioned blueprint represents a starting point in this research direction, the computing continuum poses additional side challenges that should be faced in the future.

A first security challenge is that, whenever the provider has isolation enforcement responsibility, the consumer must trust that the low-level primitives are applied to the remote cluster according to the previously specified intents and their discordance resolution. To address this issue, transparent attestation protocols could be designed so as to allow consumers to verify, in an auditable way, that the primitives have been correctly enforced.

Another challenge is that both consumers' and providers' communication control intents may evolve over time. However, in the proposed blueprint, compatibility evaluation and discordance resolution are performed only before the peering, so the created virtual cluster fulfills only the initial intents. Future research could investigate how the proposed approach can be extended with renegotiation protocols, through which the continuum participants can reevaluate their offloading strategy.

ACKNOWLEDGMENTS

This work was partly supported by EU Horizon project FLUIDOS, under grant agreement 101070473.

REFERENCES

1. M. Tortonesi, "The Compute Continuum: Trends and Challenges," *Computer*, vol. 58, no. 3, pp. 105-108, 2025. doi: 10.1109/MC.2024.3520255
2. M. Zambianco, S. Cretti and D. Siracusa, "Cost Minimization in Multi-cloud Systems with Runtime Microservice Re-orchestration," *Proc. of the 27th Conf. on Innovation in Clouds, Internet and Networks (ICIN)*, 2024, pp. 65–72. doi: 10.1109/ICIN60470.2024.10494463
3. A. Taghinezhad-Niar and J. Taheri, "Security, Reliability, Cost, and Energy-Aware Scheduling of Real-Time Workflows in Compute-Continuum Environments," *IEEE Trans. on Cloud Computing*, vol. 12, pp. 954-965, 2024. doi: 10.1109/TCC.2024.3426282
4. Y. Elkhatib and J. P. Povedano, "An Evaluation of Service Mesh Frameworks for Edge Systems," *Proc. of the 6th Int. Workshop on Edge Systems, Analytics and Networking (EdgeSys)*, pp. 19–24, 2023. doi: 10.1145/3578354.3592867
5. M. A. Azad, S. Abdullah, J. Arshad, H. Lallie and Y. H. Ahmed, "Verify and Trust: A Multidimensional Survey of Zero-Trust Security in the Age of IoT," *Internet of Things*, vol. 27, Art. no. 101227, 2024. doi: 10.1016/j.iot.2024.101227
6. A. Clemm, L. Ciavaglia, L. Z. Granville and J. Tantsura, "Intent-Based Networking: Concepts and Definitions," RFC 9315, 2022. doi:10.17487/RFC9315.
7. Li, C., Havel, O., Olariu, A., Martinez-Julia, P., Nobre, J., and D. Lopez, "Intent Classification", RFC 9316, 2022. doi:10.17487/RFC9316,.
8. B. Shafiq, J. Joshi, E. Bertino and A. Ghafoor, "Secure Interoperation in a Multidomain Environment Employing RBAC Policies," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 11, pp. 1557–1577, 2005. doi: 10.1109/TKDE.2005.185
9. Y. Guo et al., "Resolving Policy Conflicts for Cross-Domain Access Control: A Double Auction Approach," *Proc. of the 21st Int. Conf. on Computational Science (ICCS)*, vol. 12742, pp. 525–539, 2021. doi: 10.1007/978-3-030-77961-0_43
10. Q. Alam et al., "A Cross Tenant Access Control (CTAC) Model for Cloud Computing: Formal Specification and Verification," *IEEE Trans. Inf. Forensics Secur.*, vol. 12, no. 6, pp. 1259–1268, 2017. doi: 10.1109/TIFS.2016.2646639

11. A. Morichetta, N. Spring, P. Raith and S. Dustdar, “Intent-based Management for the Distributed Computing Continuum,” *Proc. of the Int. Conf. on Service-Oriented System Engineering (SOSE)*, pp. 239–249, 2023. doi: 10.1109/SOSE58276.2023.00035
12. F. Minna, A. Blaise, F. Rebecchi, B. Chandrasekaran and F. Massacci, “Understanding the Security Implications of Kubernetes Networking,” *IEEE Security & Privacy*, vol. 19, no. 5, pp. 46-56, 2021. doi: 10.1109/MSEC.2021.3094726
13. S. Rose, O. Borchert, S. Mitchell and S. Connelly, “Zero Trust Architecture,” NIST SP 800-207, 2020. doi:10.6028/NIST.SP.800-207.
14. A. Sayler, E. Keller and D. Grunwald, “Jobber: Automating Inter-Tenant Trust in the Cloud,” *Proc. of the 5th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud)*, 2013.
15. B. Tang and R. Sandhu, “Cross-tenant Trust Models in Cloud Computing,” *Proc. of the IEEE 14th Int. Conf. on Information Reuse & Integration (IRI)*, pp. 129–136, 2013. doi: 10.1109/IRI.2013.6642463
16. F. Zaritto, E. Bravi, S. Sisinni and A. Lioy, “Extending Kubernetes for Pods Integrity Verification,” *J. Netw. Syst. Manag.*, vol. 34, no. 1, Art. no. 14, 2026. doi: 10.1007/S10922-025-09988-Z
17. F. Pizzato, D. Bringhenti, R. Sisto and F. Valenza, “An intent-based solution for network isolation in Kubernetes,” *Proc. of 10th Int. Conf. on Network Softwarization (NetSoft)*, 2024, pp. 24–28. doi: 10.1109/NETSOFT60951.2024.10588939
18. J. Murcia, A. Zarca, A. Skármeta, “BASTION: Beyond automated service and security orchestration for next-generation networks”, *Computer Networks, Vol. 267*, 2025, doi: 10.1016/j.comnet.2025.111352.
19. Y. Sharma, D. Bhamare, N. Sastry, B. Javadi and R. Buyya, “SLA Management in Intent-Driven Service Management Systems: A Taxonomy and Future Directions,” *ACM Comput. Surv.*, vol. 55, no. 13s, Art. no. 292, 2023. doi: 10.1145/3589339
20. E. Dehghan Biyar et al., “Autonomous Conflict Handling in Intent-Based Management,” *Computer Networks*, vol. 271, Art. no. 111561, 2025. doi: 10.1016/j.comnet.2025.111561

Francesco Pizzato is a Ph.D. student at Politecnico di Torino, Italy. His research interests are network security automation and cloud. He received his M.Sc. degree in Computer Engineering in 2023 from Politecnico di Torino, Italy. Contact him at francesco.pizzato@polito.it.

Daniele Bringhenti is a fixed-term Assistant Professor at Politecnico di Torino, Italy. His research interests include networking technologies, automatic security configuration, formal verification of security policies. He received his Ph.D. degree in Computer and Control Engineering in 2022 from Politecnico di Torino, Italy. Contact him at daniele.bringhenti@polito.it.

Riccardo Sisto is a Full Professor at Politecnico di Torino, Italy. His main research interests are formal methods, distributed systems, and computer security. He received his Ph.D. degree in Computer Engineering in 1992 from Politecnico di Torino, Italy. Contact him at riccardo.sisto@polito.it.

Fulvio Valenza is an Associate Professor at Politecnico di Torino, Italy. His research interests include security policies, network security orchestration, threat modeling. He received his Ph.D. degree in Computer and Control Engineering in 2017 from Politecnico di Torino, Italy. Contact him at fulvio.valenza@polito.it.