The Internet of Collaborating Things: Agentic Edge AI for Autonomous Cross-Domain Collaboration

Walid A. Hanafy*, Nader Sehatbakhsh†, David Irwin*, Mani Srivastava†, Prashant Shenoy*
*University of Massachusetts Amherst, †University of California, Los Angeles

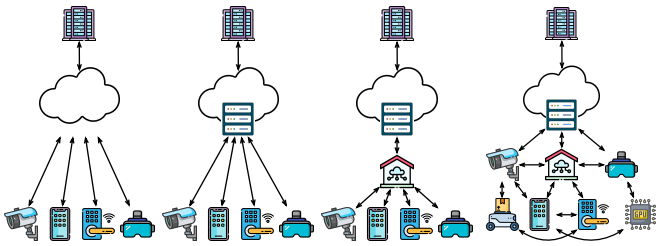
***Abstract*—The Internet of Things is on a trajectory toward a trillion connected devices deployed across multiple domains. These devices have evolved from simple sensing and actuation endpoints to mobile platforms with embedded processing and intelligent on-device services. Yet, the dominant paradigm today is a vertical device-to-server interaction model that cannot scale with this trajectory. What is needed instead is a horizontal paradigm in which devices directly communicate and collaborate, pooling their compute, sensing, and actuation into dynamic, cross-domain clusters rather than through a centralized infrastructure. We refer to this paradigm as the Internet of Collaborating Things (IoCT). Realizing this vision requires both a portable and secure execution substrate for heterogeneous hardware and an agentic control plane capable of contextual reasoning, transient trust establishment, and open-world adaptation throughout the device collaboration lifecycle without human intervention. In this position paper, we define this new communication, compute, and collaboration paradigm, articulate the role of agentic edge AI in realizing it, and outline research challenges and future directions toward trustworthy, autonomous IoCT collaboration.**

***Index Terms*—IoT, Cyber-Physical Systems, Agentic AI, LLMs, Access Control, Edge Computing, Autonomous Orchestration**

I. Introduction

The number of connected Internet of Things (IoT) devices exceeded 20 billion in 2025 [1], and some envision an Internet of a Trillion Things (IoTT) within the next decade [2]. Moreover, with recent advances in embedded hardware, low-power architectures, and domain-specific acceleration, IoT devices are becoming increasingly capable [3]. Modern IoT devices integrate multicore processors, AI accelerators, cameras, and LiDAR, often matching or exceeding the compute capabilities of the on-premises gateways that are supposed to serve them [4]–[7]. Despite this increase in device capabilities, today's IoT infrastructure is still organized around *device-to-server interaction*. Devices offload processing to a tiered hierarchy of on-premises, edge, and cloud servers, and even interactions between co-located devices are relayed through these servers or vendor clouds [5], [6], [8], [9] (see Fig. 1a). With continued growth, the number of IoT devices will dwarf the compute and bandwidth capabilities available to process their data by many orders of magnitude.

Consequently, the current *vertical* interaction model will not be able to sustain the growth in the number of IoT devices as well as the processing and bandwidth requirements of modern AI applications and high-resolution sensing modalities [5], [10]. This pressure is further amplified by the emergence of Physical AI workloads, including embodied agents, collaborative robotics, multimodal perception systems, and AR/VR



(a) On-premises-edge-cloud hierarchy (b) IoCT

Fig. 1: From a **vertical** interaction paradigm, in which devices funnel data up through an on-premises-edge-cloud hierarchy (a), to a **horizontal** *Internet of Collaborating Things*, in which heterogeneous devices communicate and collaborate directly, pooling their compute, sensing, and actuation capabilities (b).

platforms. These workloads increasingly require low-latency coordination across distributed sensors, actuators, and accelerators in the physical environment, while generating data volumes and inference demands that centralized infrastructure alone cannot sustainably handle.

These issues are exacerbated for mobile and transient devices, *e.g.*, drones and robots navigating unfamiliar territories, frequent visitors, and collaborative tasks, as detailed in Section II-A. In these scenarios, devices need to use the compute, sensing, and actuation capabilities of nearby devices. Under a centralized *vertical* interaction model, even these device-to-device interactions must be relayed through an edge or on-premises server, incurring unnecessary overhead. Instead, this position paper argues for a new paradigm that enables *horizontal* interactions between nearby devices, pooling the compute, sensing, and actuation capabilities they already carry (Fig. 1b).

We refer to this new paradigm as the *Internet of Collaborating Things* (IoCT). IoCT embraces the dispersed availability and sensing capabilities of IoT devices, and builds *dynamic*, *cross-domain* collaborative clusters that extend the conventional cloud-edge continuum downward to the device layer. In this model, IoT sensors, devices, and gateway servers in a physical space constitute a *shared pool* of sensing, actuation, computing, and networking capabilities, available to multiple applications and visiting devices simultaneously. Importantly, cross-device collaboration and the associated *perception–cognition–action* loops get organized (and reorganized) across

an amorphous collection of IoT devices that span different trust domains.

In IoCT, each application sees a dynamic and heterogeneous cluster of IoT sensors, devices, and on-premises servers, with wide-area connectivity to the edge and cloud infrastructure. To enable seamless cross-device collaboration and cooperative execution, IoCT applications are composed as portable dataflow graphs whose leaf nodes are either *nanotasks*, platform-independent computation mapped at runtime to the CPUs, GPUs, and accelerators available on nearby devices [11], or *nanoservices* that expose device-native sensing, actuation, and communication primitives.

Providing this flexible, on-demand computational substrate raises four interrelated challenges:

1) **Device Discovery.** The IoCT environment is diverse and fluid. Thus, the substrate must continuously accommodate devices as they arrive and depart, identify the compute, sensing, actuation, and other shareable capabilities they offer, maintain an up-to-date network view, and auto-configure devices and networks for seamless access, all without human intervention.
2) **Cross-Domain Security and Access Control.** Because devices roam across administrative domains, every collaboration session requires authenticating visiting devices, dynamically attesting *both* code and shared data, and enforcing fine-grained access control policies that are generated on demand, scoped to a specific interaction, and revoked upon departure.
3) **Heterogeneity-Aware Orchestration.** Unlike homogeneous edge servers, the device layer incorporates a high degree of hardware heterogeneity, making it challenging to determine how an application's dataflow graph should be composed across devices, which device hosts each nanotask, and which nanoservices fulfill each role. Further, as workloads and the device population change rapidly, the substrate must continuously recompose these decisions and rebalance the resources devoted to nanotasks and nanoservices.
4) **Constrained Execution.** Beyond mapping nanotasks to heterogeneous devices, the substrate must execute compute-intensive and latency-sensitive workloads under tight compute, memory, energy, and thermal constraints. This challenge is especially pronounced for AI workloads, where models and agents may need to be partitioned, compressed, migrated, or adapted at runtime to balance accuracy, latency, energy consumption, and availability.

Realizing IoCT and overcoming these challenges will require rethinking current edge architectures, including their data and control planes. On the *data plane*, IoCT applications require a portable and secure execution substrate that maps nanotasks to heterogeneous hardware [11]–[13] and provides an adaptive, trusted, and secure execution platform [14]–[17], that ensures secure and trusted communication and access control for nanoservices between IoCT devices from different administrative domains [18]–[20]. On the *control plane*, a unified autonomous orchestration layer must discover devices, validate cross-domain credentials and attestations, allocate resources, and adapt continuously as clusters form and dissolve in open-world physical environments [21]–[24].

Our Position. Realizing the IoCT vision requires a fundamentally new edge architecture governed by autonomous agents that manage the full device collaboration lifecycle across administrative trust domains without human intervention at runtime. This lifecycle spans discovering nearby devices, negotiating scoped access, composing shared dataflows, and adapting and revoking access as devices depart. These agents provide the contextual reasoning, transient trust establishment, and open-world adaptation needed when devices arrive with unfamiliar identities, capabilities, policies, and task needs. In this architecture, human administrators express high-level intent, while agents act as orchestrators that inspect newly discovered devices' identities, capabilities, and task needs, and evaluate credential and attestation results. Agents then produce validated policy and placement decisions, deciding where nanotasks execute on the portable substrate and which nanoservices a visiting device may invoke, and adapt these decisions as devices arrive, depart, or move through a space. This allows devices to shift from standalone operation to cooperative use of nearby resources when context and policy permit, without requiring manual configuration for each encounter. Without this autonomy, IoT orchestration cannot scale to the frequency, heterogeneity, and cross-domain mobility expected in future trillion-device deployments.

This position paper makes three contributions. First, we define the IoCT paradigm, present motivating scenarios, and show why existing approaches fall short (Section II). Second, we propose an initial IoCT architecture with a data plane for portable cross-device execution and an agentic control plane for autonomous orchestration (Section III). Third, we identify open research challenges spanning verifiable agentic decisions, trust negotiation, and resource management (Section IV).

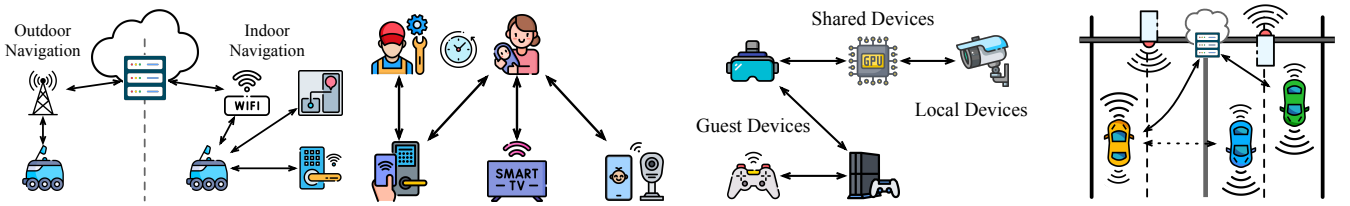
II. Why Is IoCT Hard to Achieve?

This section presents motivating examples for IoCT and explains why the state of the art falls short.

A. Motivational Examples

We begin with future IoT scenarios where the conventional vertical interaction model shown in Fig. 1a is no longer adequate and IoCT offers a more effective alternative. The following scenarios illustrate what this horizontal paradigm looks like in practice.

Delivery Robot. Food delivery robots (see Fig. 2a) are now common on many university campuses [25], and delivery providers and online retailers are increasingly experimenting with autonomous robots or drones to deliver packages [26], [27]. A robot may navigate *autonomously* outdoors using local vision, LiDAR, GPS, and optional computation offload to 5G-integrated edge servers such as AWS Wavelength [28]. Once near or inside a building, the robot can switch from *standalone* navigation to *cooperative* operation with the building's available sensors and devices. To do so, the robot must connect to the building's Wi-Fi network, query the building's indoor map,



(a) Delivery Robot (b) Frequent Visitors (c) Collaborative Gaming (d) Vehicular Communication

Fig. 2: Four motivational scenarios for IoCT. In each case, a device accesses and collaborates with devices across trust domain boundaries.

request authorized entry through smart-lock services [29], and place computation on on-premises servers to benefit from lower latency, improved reliability, and higher-quality prediction and control.

Frequent Visitors. Another example is a frequent visitor (see Fig. 2b), such as a babysitter or maintenance personnel, who requires access to the environment's devices. The visitor's device must be granted access to the door locks and internal appliances relevant to their expected duties. For example, the babysitter requires access to the baby monitor camera and limited access to entertainment devices (*e.g.*, a TV or wireless speakers). This scenario poses challenges for fine-grained, time-bounded access control that current models do not address [30]. For example, for privacy reasons, the babysitter should be able to access the baby monitor only during the visit, without being able to reconfigure it.

Collaborative Gaming. Consider a group of friends who meet at one of their homes to play together (see Fig. 2c). They bring their AR headsets (*e.g.*, Apple Vision Pro [31]), wireless game controllers, and mobile devices, and gain access to the home's smart TV and gaming console for multiplayer gaming, while controlling the living room's lights and music for an enhanced experience. The AR headsets can also use an on-premises PC or server for AR processing. This scenario requires applications to coordinate resources and services while maintaining performance isolation among users, applications, and background home services.

Vehicular Communication. Finally, consider autonomous vehicles driving on roads instrumented with city-operated cameras, LiDAR units, and edge compute (see Fig. 2d). Each vehicle perceives only part of its surroundings due to occlusions and limited sensor range, and fusing views across nearby vehicles and roadside sensors improves coverage and safety [32]. A single autonomous vehicle can generate terabytes of sensor data per day [33], so streaming this high-dimensional perception data to distant servers would saturate wireless uplinks, while cooperative maneuvers require fused results within tens of milliseconds. Instead, vehicles and roadside units must exchange and fuse intermediate representations locally, offloading perception and fusion tasks across one another's onboard accelerators and to the roadside edge. This scenario requires sharing high-dimensional perceptual data under continuous device churn and across trust domains.

These scenarios span different physical domains, device types, and application requirements, yet they share a common structure in which a *visiting* device enters a *space* (an administrative domain such as a building, home, or road segment) that governs a population of *resident* devices. The visiting device discovers available resources, obtains scoped access across *trust domain* boundaries, collaborates, and eventually departs. Together, the scenarios surface the demands an IoCT substrate must meet. These include offloading to and accessing resources across trust domains, enforcing fine-grained and time-bounded access control, managing shared resources among concurrent users, and sustaining high-bandwidth, low-latency collaboration under continuous device churn. For simplicity, we use the delivery robot as an example throughout this position paper.

B. Limitations of State of the Art

Current approaches fall short on one or more IoCT requirements.

Mobile Clusters. Dynamic clustering in ad hoc networks [34], delay-tolerant networking [35], opportunistic computing over encountered devices [36], and mobile device clouds [37] pool the communication and compute resources of nearby devices under churn, and address cluster formation and disruption tolerance. However, these approaches assume a single trust domain, mutually cooperative participants, and a common runtime, and their execution model is compute-centric and delay-tolerant.

Cloud and Edge Platforms. Earlier offloading approaches have utilized nearby edge resources to augment the computational and communication capabilities of IoT devices through static and dynamic offloading techniques [6], [38]–[40]. Commercial IoT platforms such as AWS IoT [8], Azure IoT Edge [9], and AWS Greengrass [41] position computation across the cloud-edge continuum. AWS Wavelength [28] pushes services onto 5G edge nodes for lower latency. However, these approaches focus on the vertical interaction model, treating devices as clients of provisioned infrastructure rather than peers that both consume and contribute compute, sensing,

and actuation. More importantly, device registries are pre-provisioned, trust is anchored in a single vendor's identity system, and there is no mechanism for cross-domain resource pooling.

IoT Composition Frameworks. Systems for composing IoT applications across distributed devices, such as declarative dataflow frameworks [13] and service-oriented IoT middleware [42], partition applications statically at deployment time. They do not handle dynamic device populations, visiting devices from foreign domains, or runtime reallocation as the environment changes.

Secure Isolation Mechanisms. WebAssembly (WASM) [12] and Unikernels [43] provide lightweight, sandboxed execution. However, these runtimes do not yet support the cross-tenant performance isolation and secure multi-tenant sharing of sensors that the IoCT device layer requires [19]. In addition, trusted execution environments (TEEs) such as ARM TrustZone and Intel SGX offer hardware-backed isolation but are hardware-specific, unavailable on many low-end devices, and do not address cross-domain authentication, attestation validation, or scoped access-control decisions by themselves [44].

Attestation and Device Integrity. Remote attestation methods, whether hardware-based (TPMs, SGX enclaves), software-based, or hybrid [44], [45], can verify device integrity, but are designed for static deployments with pre-established verifier–prover relationships. They do not support the transient, cross-domain authentication and attestation workflows that feed scoped access-control decisions. Privacy-preserving techniques such as differential privacy [46] and homomorphic encryption [47] can limit data disclosure in specific processing settings, but do not dynamically establish data or code integrity or guarantee the trustworthiness of sensor inputs or actuator outputs.

Service Discovery. Protocols such as UPnP [48], Zeroconf [22], and Jini [49] provide plug-and-play device registration within a local network but assume implicit trust among participants. They offer no access control, capability profiling, or cross-domain identity verification, and are not designed for untrusted or multi-tenant environments.

III. Realizing IoCT

Before presenting the architecture and the role of agentic edge AI, we list the core design principles guiding IoCT:

- **Horizontal collaboration over vertical communication.** Rather than treating IoT devices solely as clients that offload computation to edge or cloud infrastructure and interact through vendor clouds, IoCT enables nearby devices to communicate directly and pool sensing, computation, networking, and actuation resources.
- **Transient cross-domain federation.** IoCT collaborations are dynamic and short-lived, forming opportunistically as devices enter a physical space and dissolving when they depart, often across administrative trust boundaries with no prior relationship.
- **Portable execution across heterogeneous devices.** Applications must execute seamlessly across diverse hardware platforms, including microcontrollers, embedded processors, GPUs, and specialized accelerators, without requiring device-specific deployment logic.
- **Policy-driven autonomous orchestration.** Because IoCT environments are too dynamic and heterogeneous for manual management, orchestration must be driven by high-level intent and automated reasoning over device capabilities, trust relationships, and runtime context, operating in the control plane rather than on the data path.
- **Continuous adaptation to mobility and resource dynamics.** The system must continuously reconfigure placement, access control, and resource allocation as devices move, workloads fluctuate, and network conditions evolve.
- **Least-privilege, time-bounded collaboration.** Access to resources and services should be scoped to the minimum permissions required for a specific collaboration session and revoked automatically when the interaction terminates.

A. An Initial IoCT Architecture

The IoCT architecture considers heterogeneous IoT devices in a physical space, including resident and visiting devices that may belong to different administrative domains. These devices vary in compute, sensing, and actuation capabilities, and some have wide-area connectivity to edge and cloud infrastructure. IoCT uses two complementary layers: (1) a portable and secure communication and execution substrate that handles heterogeneous hardware, and (2) a new control plane that orchestrates the full device collaboration lifecycle without human intervention.

Data Plane. Rather than devices acting only as clients of the cloud-edge continuum, an application sees a dynamic and heterogeneous cluster of IoT devices and on-premises servers with wide-area network connectivity to edge and cloud servers. IoT devices are equipped with sensors, actuators, embedded processors, and specialized accelerators, while on-premises, edge, and cloud servers offer general-purpose computing resources, including CPUs, GPUs, and FPGAs. The IoCT architecture augments the current interaction model by enabling discovery and use of the sensing, actuation, computation, and connectivity available from devices and servers within the visited space.

Device collaboration requires a communication substrate that spans the heterogeneous wireless links (Wi-Fi, 5G/6G, BLE [50], LoRa [51]) present in a typical physical space. Rather than requiring applications to manage each link individually, the data plane should provide a unified messaging abstraction, such as a publish-subscribe model, that transparently bridges communication across device boundaries. For example, an embedded message broker on each device can provide local module-to-module messaging, while brokers on different devices bridge to one another over the network using lightweight IoT protocols such as MQTT [52]. The broker also mediates access control by checking capability-based tokens

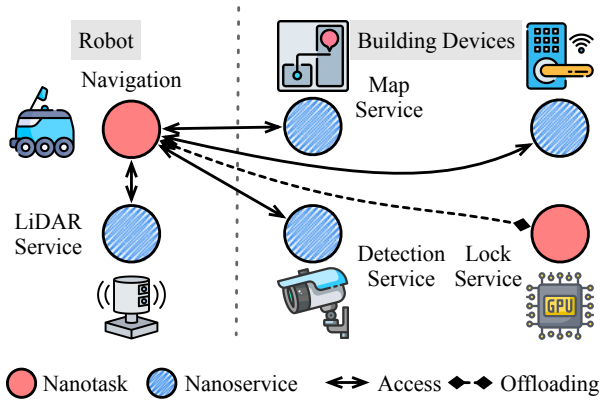


Fig. 3: Dataflow graph for the delivery-robot scenario. The robot's mission decomposes into nanotasks and nanoservices mapped across two trust domains.

at wire speed [21], ensuring that policies generated by the control plane are enforced on every interaction.

In addition to communication, the data plane supports a hardware-independent deployment substrate. For portable execution, IoCT can use WebAssembly (WASM), which provides a language- and hardware-independent binary format that offers compact representation, efficient validation, and safe low-overhead execution [12], and has been shown to be viable on resource-constrained microcontrollers [19], [53]. To protect not only the host device from misbehaving tenant code, but also the tenant's code from a misbehaving host, the WASM runtime can be placed inside a hardware trusted execution environment (TEE) such as ARM TrustZone [53], [54], creating a two-way isolation boundary. Beyond CPU and memory, the data plane must also support fine-grained, multi-tenant sharing of sensors, actuators, and hardware accelerators, despite the lack of conventional virtualization for such IoT devices.

Application Model. An IoCT application is expressed as a portable dataflow graph whose nodes are either *nanotasks* or *nanoservices*. Nanotasks are lightweight, platform-independent computation units that can be dynamically placed across nearby heterogeneous devices at fine granularity [11]. Nanoservices expose device-native sensing, actuation, and communication primitives as composable runtime services.

To enable this mapping, each device advertises its available resources and nanoservices using structured capability descriptions (*e.g.*, protocol-buffer payloads over MQTT), allowing the runtime to discover available hardware and services.

In the delivery-robot scenario, the robot can use its own resources as well as resources provided by the building. As shown in Fig. 3, the robot's mission can be fulfilled using a set of nanotasks (*e.g.*, navigation), onboard nanoservices, and building-provided nanoservices. For instance, the navigation task can use onboard LiDAR to plan a path, and use building nanoservices, including a map nanoservice that returns the building layout and a lock-actuation nanoservice that triggers the building's smart lock upon successful verification. The robot can execute these tasks locally, use the building's resources, or fall back to nearby edge servers or its home cloud for compute-heavy and latency-tolerant work, based on device capabilities, resource availability, and access permissions.

Control Plane. Complementary to the data plane, a control plane is needed to decide *which* device runs *what*, under *whose* authority, and to *adapt* those decisions as conditions change. The control plane implements several interrelated functions that operationalize the challenges of Section I, with constrained execution handled by the data plane substrate described above, namely: *(i) discovery and capability profiling:* detecting new devices and learning what resources and services they offer, extending zero-configuration networking to cross-domain settings [22], *(ii) resource management:* mapping nanotasks onto the available device pool under latency and capacity constraints, and re-mapping when devices arrive or depart, *(iii) cross-domain authentication and attestation:* authenticating visiting devices and validating credentials and attestations across administrative domains, *(iv) fine-grained access control:* generating and enforcing time-bounded policies that govern which devices may use which resources, and *(v) runtime adaptation:* continuously monitoring and adjusting all of the above as the physical context evolves.

To realize these functions, the control plane maintains a device registry that tracks the capabilities, locations, and credential and attestation metadata of all devices in the space, enabling both discovery and resource allocation. When a visiting device from a different trust domain arrives, the control plane performs credential exchange and attestation checks for that visit. The validation results inform the fine-grained access-control policies for the visit, which are encoded as short-lived capability tokens and pushed to the data plane message brokers for enforcement. The control plane maps nanotasks onto available devices, and continuous monitoring triggers re-orchestration when the context changes, whether a device departs, a workload shifts, or a visiting device moves to a different zone. This separation keeps reasoning and control decisions in the control plane while leaving low-latency enforcement to the data plane. However, because the device combinations and contexts involved cannot be anticipated at design time, and new collaboration sessions arise too frequently for manual configuration, the control plane must operate autonomously. We propose an agentic control plane in which agents reason over intent, context, trust evidence, and resource state to handle these functions.

B. Leveraging Edge AI Agents

The IoCT control plane must continuously discover devices, validate cross-domain credentials and attestations, generate policies, and adapt to changing conditions, all without human intervention at runtime. While these functions are implemented as structured, deterministic mechanisms, static rules cannot decide how to *compose* them for each collaboration session, because the device combinations, credential requirements, and environmental conditions are impossible to anticipate at design

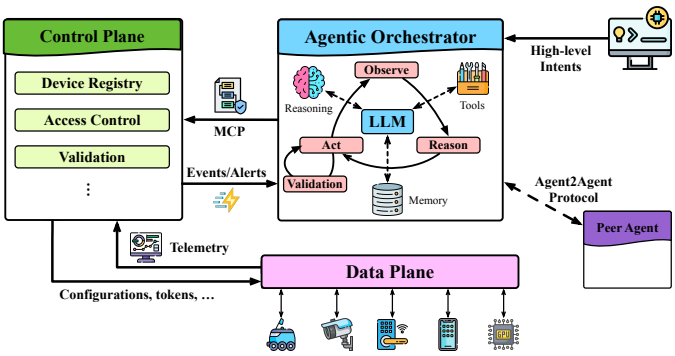


Fig. 4: Reference architecture of the IoCT agentic orchestrator.

time. Further, the challenge is not merely optimization complexity, but open-world contextual reasoning under incomplete and continuously evolving information, where rigid rule-based orchestration becomes brittle and unscalable. Thus, the IoCT control plane requires an *agentic orchestrator* that can perform contextual reasoning, establish transient trust, and adapt in open-world physical environments. Recent LLM-based agent architectures provide one promising way to implement this reasoning layer [55]–[59]. The agent receives high-level intent from human administrators, and supports the collaboration lifecycle by producing validated access-control, placement, and reconfiguration decisions for the control plane to apply. This builds on the LLM-as-orchestrator paradigm [58], but extends it with continuous reactivity and domain-grounded validation. The orchestrator maintains a persistent observe-reason-act loop over the physical environment rather than executing a one-shot plan. The IoCT orchestrator is event-driven, remaining idle until the control plane raises an alert (*e.g.*, a device arrives or departs) and then reasoning over the current state to produce a targeted decision rather than re-planning from scratch.

Fig. 4 illustrates the reference architecture for the IoCT agentic orchestrator. Human administrators define high-level standing intents (*e.g.*, "delivery robots may use lobby cameras, but not residential devices"), while the control plane maintains the device registry, policy state, credential and attestation state, and token issuer. The data plane reports telemetry to the control plane and receives configurations, policies, and capability tokens for enforcement. Then, based on events and alerts from the control plane, the agent evaluates the visiting device's credentials and attestation results, administrator intent, device capabilities, resource availability, and policy constraints to decide what access-control policies, placements, token requests, or reconfigurations are needed. Because these decisions affect cyber-physical resources, they must be validated before the control plane applies them. The agent therefore uses a multi-stage validation pipeline with syntactic and semantic checks for generated policies, placements, and token requests, where only outputs that pass all validation stages are applied by the control plane [21].

The agent communicates these validated decisions to the control plane through the Model Context Protocol (MCP) [60], which serves as the standardized interface between the agent and the control plane. MCP decouples the agent's reasoning logic from the control plane implementation, while data plane brokers enforce the resulting policies and capability tokens. Finally, because each administrative domain operates its own orchestrator, and neither has authority over the other, the two orchestrators must negotiate as peers when a visiting device crosses a domain boundary. This inter-agent communication uses semantics *inspired* by the Agent-to-Agent (A2A) protocol [61], where each agent publishes an Agent Card describing its domain's capabilities and policies, and negotiation proceeds through a structured task lifecycle.

C. The Collaboration Lifecycle

Figure 5 summarizes the delivery-robot collaboration lifecycle, from arrival to departure. When the robot enters the building lobby, the lifecycle begins with *discovery and capability profiling*. The building control plane then observes a newly associated device through the deployment's discovery mechanism and raises an event to the building's agentic orchestrator. The orchestrator queries the device registry, inspects the robot's advertised identity, capabilities, and task needs, and retrieves the relevant administrator intent through the MCP interface to the control plane.

The control plane then performs the required cross-domain security checks, including *credential and attestation validation*. These checks authenticate the robot's identity and validate claims such as its operator, device type, declared capabilities, firmware, and delivery purpose. The orchestrator uses the validated identity and claims, together with the administrator intent, current building context, available resources, and the robot's delivery task, to derive *scoped access control* and *resource placement* decisions. In this example, it may decide that the robot can use the building's map, elevator, and lock-actuation nanoservices, as well as on-premises compute resources needed for the delivery task.

The control plane applies these decisions by issuing short-lived signed capability tokens to the robot. The robot presents these tokens when accessing resident resources, and the data plane validates them and only permits authorized interactions, without involving the orchestrator. As the visit progresses, data plane telemetry flows back to the control plane. If the robot moves to a different part of the building or resource availability changes, the control plane raises an alert to the orchestrator, which evaluates the updated context and decides whether policies, placements, or tokens should be changed. Next, *runtime adaptation* keeps the data plane enforcement state aligned with the robot's changing physical location and the building's current operating conditions. Finally, when the robot departs, the control plane triggers *revocation*, invalidating any visit-specific policies and tokens.

IV. Open Research Challenges

Realizing the IoCT vision poses research challenges that span the device-layer substrate, cross-domain security, and

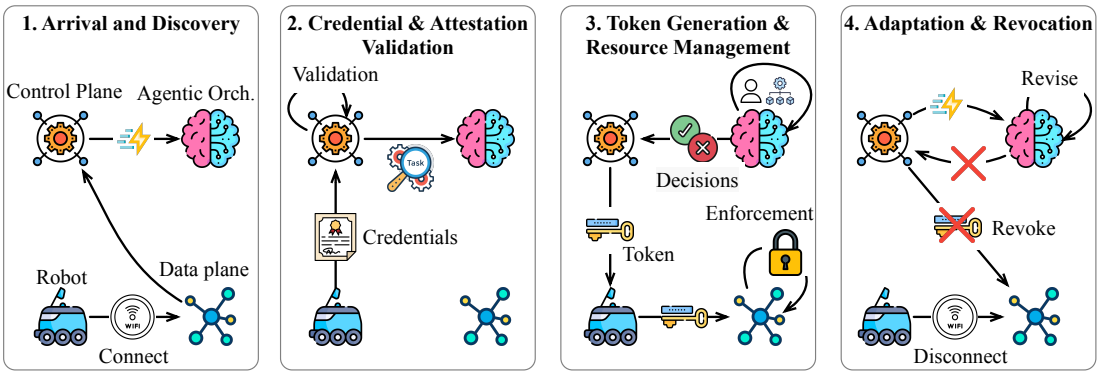


Fig. 5: Collaboration lifecycle for the delivery-robot scenario.

autonomous orchestration.

Safe and Verifiable Agentic Decisions. The agentic orchestrator derives resource-management and access-control decisions from context, device capabilities, and domain policies, while the control plane applies them. When implemented with LLM-based reasoning, however, the orchestrator may generate hallucinated or inconsistent outputs [62], risking excessive access, incorrect placement, or unsafe actuation. Existing LLM-based policy generation and validation pipelines address parts of this problem but do not guarantee correctness for broader IoCT decisions [21], [63]–[66]. Future work must combine structured output schemas, satisfiability checks, and formal safety verification.

Cross-Domain Trust Negotiation. When a visiting device crosses a domain boundary, the two domains must establish trust and agree on resource-sharing terms without a prior relationship [38]. Unlike traditional federated systems with long-lived agreements, IoCT interactions are transient and may involve heterogeneous credentials, attestations, and policy models. Existing trust-negotiation and agent-to-agent protocols do not cover the credential exchange, attestation validation, and policy reconciliation needed by IoCT [61], [67]. Open questions include resolving policy conflicts, scoping delegation, and assigning accountability when a visiting device causes harm.

Bidirectional Resource Sharing Under Asymmetric Trust. IoCT enables visiting devices not only to consume host resources but also to contribute capabilities, such as a robot's GPU accelerating building inference tasks. This creates asymmetric trust, since the host may benefit from a visitor's resources without fully trusting the visitor. Existing isolation and cryptographic mechanisms are often hardware-specific, unavailable on many devices, or too costly for latency-sensitive workloads [44], [47]. Future systems must support varying levels of trust [68], partition workloads to protect sensitive state, verify results from less-trusted executors, and update trust levels as evidence changes during a visit.

Dynamic and Heterogeneous Resource Management. IoCT environments are fluid, with devices arriving, departing, moving between zones, and failing without warning. The device pool is also large and highly heterogeneous, spanning microcontrollers, embedded processors, GPUs, and specialized accelerators that often lack standard virtualization and resource-management abstractions. The control plane must therefore utilize lightweight mechanisms that re-orchestrate placement, policies, and resource allocations under latency and safety constraints as workloads and device availability change. Open challenges include managing execution state after unexpected departures and sharing sensors, actuators, and accelerators across tenants without conventional memory-protection mechanisms.

Incentives and Economic Models for Collaboration. IoCT assumes devices and environments will voluntarily contribute sensing, compute, networking, and actuation resources to transient collaborations. However, these resources incur energy, wear, privacy, and opportunity costs. Future systems require incentive and accounting mechanisms to quantify resource contributions, prioritize competing workloads, and prevent abuse or free-riding behavior [36]. Open questions include how to value resource usage, enforce fairness, and choose among cooperative, contractual, or market-driven models.

V. Conclusion

The IoT is approaching a scale and mobility regime that static, human-configured orchestration cannot sustain. As devices become more capable and mobile, they must collaborate directly with nearby devices across administrative trust boundaries, pooling compute, sensing, and actuation into dynamic clusters rather than offloading to centralized infrastructure. In this position paper, we presented the Internet of Collaborating Things (IoCT), a new paradigm that shifts IoT devices from isolated endpoints to autonomous participants in a continuously evolving computational collective. We proposed an initial architecture of IoCT that combines a portable execution substrate with an agentic control plane, and identified open research challenges spanning verifiable agentic decisions, cross-domain trust negotiation, bidirectional resource sharing under asymmetric trust, and resource management over dynamic, heterogeneous device populations.

ACKNOWLEDGMENTS

We thank the anonymous SEC reviewers for their valuable feedback. This research was supported by the National Science Foundation (NSF) grants 2213636, 2211302, 2211888, 2325956, 2105494, and 2211301; U.S. Army grant W911NF-17-2-0196; Sandia National Laboratories award 2169310; and National Institutes of Health (NIH) award P41EB028242.

REFERENCES

[1] IoT Analytics, "State of IoT 2025: Number of connected IoT devices growing 14% to 21.1 billion globally," https://iot-analytics.com/number-connected-iot-devices/, October 2025, accessed: 2026-04-20.

[2] I. Scales, "Arm predicts 1 trillion IoT devices by 2035 with new end-to-end platform," https://www.itu.int/hub/2020/04/arm-predicts-1-trillion-iot-devices-by-2035-with-new-end-to-end-platform/, April 2020, accessed: 2026-04-20.

[3] L. D. Xu, W. He, and S. Li, "Internet of Things in Industries: A Survey," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 4, pp. 2233–2243, 2014.

[4] P. Gupta and S. S. Iyer, "Goodbye, motherboard. Bare chiplets bonded to silicon will make computers smaller and more powerful: Hello, silicon-interconnect fabric," *IEEE Spectr.*, vol. 56, no. 10, pp. 28–33, Oct. 2019. [Online]. Available: https://doi.org/10.1109/MSPEC.2019.8847587

[5] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge Computing: Vision and Challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016.

[6] M. Satyanarayanan, "The Emergence of Edge Computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[7] J. Lin, W.-M. Chen, Y. Lin, J. Cohn, C. Gan, and S. Han, "MCUNet: Tiny Deep Learning on IoT Devices," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS '20. Red Hook, NY, USA: Curran Associates Inc., 2020.

[8] Amazon Web Services. (2026) AWS IoT. [Online]. Available: https://aws.amazon.com/iot/

[9] Microsoft, "Azure IoT Edge Documentation — Microsoft Learn," https://learn.microsoft.com/en-us/azure/iot-edge/, 2026, accessed: 2026-04-20.

[10] J. Meng, Z. J. Kong, Y. C. Hu, M. G. Choi, and D. Lal, "Do We Need Sophisticated System Design for Edge-assisted Augmented Reality?" in *Proceedings of the 5th International Workshop on Edge Systems, Analytics and Networking*, ser. EdgeSys '22, 2022, pp. 7–12. [Online]. Available: https://doi.org/10.1145/3517206.3526267

[11] M. Kotsifakou, P. Srivastava, M. D. Sinclair, R. Komuravelli, V. Adve, and S. Adve, "HPVM: Heterogeneous Parallel Virtual Machine," in *Proceedings of the 23rd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, ser. PPoPP '18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 68–80. [Online]. Available: https://doi.org/10.1145/3178487.3178493

[12] A. Haas, A. Rossberg, D. L. Schuff, B. L. Titzer, M. Holman, D. Gohman, L. Wagner, A. Zakai, and J. Bastien, "Bringing the Web up to Speed with WebAssembly," in *Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation*, ser. PLDI 2017. New York, NY, USA: Association for Computing Machinery, 2017, pp. 185–200. [Online]. Available: https://doi.org/10.1145/3062341.3062363

[13] J. Noor, H.-Y. Tseng, L. Garcia, and M. Srivastava, "DDFlow: Visualized Declarative Programming for Heterogeneous IoT Networks," in *Proceedings of the International Conference on Internet of Things Design and Implementation*, ser. IoTDI '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 172–177. [Online]. Available: https://doi.org/10.1145/3302505.3310079

[14] J. Wang, A. Li, H. Li, C. Lu, and N. Zhang, "RT-TEE: Real-time System Availability for Cyber-physical Systems using ARM TrustZone," in *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022, pp. 352–369.

[15] F. Alder, J. Van Bulck, F. Piessens, and J. T. Mühlberg, "Aion: Enabling open systems through strong availability guarantees for enclaves," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 1357–1372.

[16] F. Arkannezhad, J. Feng, and N. Sehatbakhsh, "IDA: Hybrid Attestation with Support for Interrupts and TOCTOU," in *Network and Distributed System Security (NDSS) Symposium*, 2024.

[17] N. Sehatbakhsh, A. Nazari, H. Khan, A. Zajic, and M. Prvulovic, "EMMA: Hardware/Software Attestation Framework for Embedded Systems Using Electromagnetic Signals," in *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO-52. New York, NY, USA: Association for Computing Machinery, 2019, p. 983–995. [Online]. Available: https://doi.org/10.1145/3352460.3358261

[18] M. Shakarami, J. O. Benson, and R. S. Sandhu, "Scenario-Driven Device-to-Device Access Control in Smart Home IoT," *2022 IEEE 4th International Conference on Trust, Privacy and Security in Intelligent Systems, and Applications (TPS-ISA)*, pp. 217–228, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:257537545

[19] R. Liu, L. Garcia, and M. Srivastava, "Aerogel: Lightweight Access Control Framework for WebAssembly-Based Bare-Metal IoT Devices," in *2021 IEEE/ACM Symposium on Edge Computing (SEC)*, 2021, pp. 94–105.

[20] A. Alkhresheh, K. Elgazzar, and H. S. Hassanein, "DACIoT: Dynamic Access Control Framework for IoT Deployments," *IEEE Internet of Things Journal*, vol. 7, no. 12, pp. 11 401–11 419, 2020.

[21] H. Shastri, W. Hanafy, L. Wu, D. Irwin, M. Srivastava, and P. Shenoy, "LLM-Driven Auto Configuration for Transient IoT Device Collaboration," in *Proceedings of the Tenth ACM/IEEE Symposium on Edge Computing*, ser. SEC '25. New York, NY, USA: Association for Computing Machinery, 2025. [Online]. Available: https://doi.org/10.1145/3769102.3770619

[22] D. Stirling and F. Al-Ali, "Zero Configuration Networking," *XRDS*, vol. 9, no. 4, pp. 19–23, Jun. 2003. [Online]. Available: https://doi.org/10.1145/904080.904084

[23] P. C. Ccori, L. C. C. De Biase, M. K. Zuffo, and F. S. C. da Silva, "Device discovery strategies for the IoT," in *2016 IEEE International Symposium on Consumer Electronics (ISCE)*. IEEE, 2016, pp. 97–98.

[24] M. Achir, A. Abdelli, L. Mokdad, and J. Benothman, "Service discovery and selection in IoT: A survey and a taxonomy," *Journal of Network and Computer Applications*, vol. 200, p. 103331, 2022.

[25] Associated Students UCLA (ASUCLA), "UCLA Food Delivery with Starship Robots," 2024, accessed: 2024-11-05. [Online]. Available: https://www.asucla.ucla.edu/starship

[26] DoorDash, "DoorDash Unveils Dot, the Delivery Robot Powered by its Autonomous Delivery Platform to Accelerate Local Commerce," https://about.doordash.com/en-us/news/doordash-unveils-dot, Sep. 2025, accessed: 2026-04-30.

[27] Amazon, "Prime Air Drone Delivery," https://www.amazon.com/Prime-Air-Drone-Delivery, 2026, accessed: 2026-04-30.

[28] Amazon Web Services, Inc., "5G Edge Computing Infrastructure – AWS Wavelength," https://aws.amazon.com/wavelength/, 2026, accessed: 2026-04-20.

[29] S. Wolfson, "Amazon Key In-Garage Delivery: What It Is and How It Works," 2023, accessed: 2024-11-10. [Online]. Available: https://www.aboutamazon.com/news/amazon-prime/what-is-amazon-key-in-garage-delivery

[30] W. He, M. Golla, R. Padhi, J. Ofek, M. Dürmuth, E. Fernandes, and B. Ur, "Rethinking Access Control and Authentication for the Home Internet of Things (IoT)," in *27th USENIX Security Symposium (USENIX Security 18)*, Baltimore, MD, Aug. 2018, pp. 255–272.

[31] Apple Inc., "Apple Vision Pro - Apple," 2024. [Online]. Available: https://www.apple.com/apple-vision-pro/

[32] Q. Chen, S. Tang, Q. Yang, and S. Fu, "Cooper: Cooperative Perception for Connected Autonomous Vehicles Based on 3D Point Clouds," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. Los Alamitos, CA, USA: IEEE Computer Society, Jul. 2019, pp. 514–524. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/ICDCS.2019.00058

[33] S. Lu and W. Shi, "The Emergence of Vehicle Computing," *IEEE Internet Computing*, vol. 25, no. 3, pp. 18–22, 2021.

[34] C. Lin and M. Gerla, "Adaptive clustering for mobile wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 7, pp. 1265–1275, 1997.

[35] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '03. New York, NY, USA: Association for Computing Machinery, 2003, p. 27–34. [Online]. Available: https://doi.org/10.1145/863955.863960

[36] M. Conti and M. Kumar, "Opportunities in Opportunistic Computing," *Computer*, vol. 43, no. 1, pp. 42–50, 2010.

[37] K. Habak, M. Ammar, K. A. Harras, and E. Zegura, "Femto clouds: Leveraging mobile devices to provide cloud service at the edge," in *2015 IEEE 8th International Conference on Cloud Computing*, 2015, pp. 9–16.

[38] M. Satyanarayanan, "Pervasive computing: vision and challenges," *IEEE Personal Communications*, vol. 8, no. 4, pp. 10–17, 2001.

[39] R. Balan, J. Flinn, M. Satyanarayanan, S. Sinnamohideen, and H.-I. Yang, "The Case for Cyber Foraging," in *Proceedings of the 10th Workshop on ACM SIGOPS European Workshop*, ser. EW 10, 2002, p. 87–92. [Online]. Available: https://doi.org/10.1145/1133373.1133390

[40] P. Mach and Z. Becvar, "Mobile Edge Computing: A Survey on Architecture and Computation Offloading," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[41] AWS, "AWS IoT Greengrass Documentation," https://docs.aws.amazon.com/greengrass/, 2023, accessed: 2023-10-23.

[42] N. K. Giang, M. Blackstock, R. Lea, and V. C. Leung, "Developing IoT applications in the Fog: A Distributed Dataflow Approach," in *2015 5th International Conference on the Internet of Things (IOT)*, 2015, pp. 155–162.

[43] A. Madhavapeddy, R. Mortier, C. Rotsos, D. Scott, B. Singh, T. Gazagnaire, S. Smith, S. Hand, and J. Crowcroft, "Unikernels: library operating systems for the cloud," in *Proceedings of the Eighteenth International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '13, 2013, p. 461–472. [Online]. Available: https://doi.org/10.1145/2451116.2451167

[44] F. Brasser, B. El Mahjoub, A.-R. Sadeghi, C. Wachsmann, and P. Koeberl, "TyTAN: Tiny Trust Anchor for Tiny Devices," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2015, pp. 1–6.

[45] J. Noorman, J. V. Bulck, J. T. Mühlberg, F. Piessens, P. Maene, B. Preneel, I. Verbauwhede, J. Götzfried, T. Müller, and F. Freiling, "Sancus 2.0: A Low-Cost Security Architecture for IoT Devices," *ACM Transactions on Privacy and Security (TOPS)*, vol. 20, no. 3, Jul. 2017. [Online]. Available: https://doi.org/10.1145/3079763

[46] C. Dwork and A. Roth, "The Algorithmic Foundations of Differential Privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, no. 3–4, p. 211–407, Aug. 2014. [Online]. Available: https://doi.org/10.1561/0400000042

[47] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes: Theory and implementation," *ACM Comput. Surv.*, vol. 51, no. 4, Jul. 2018. [Online]. Available: https://doi.org/10.1145/3214303

[48] Open Connectivity Foundation, "UPnP device architecture 2.0," Open Connectivity Foundation, Tech. Rep., April 2020, accessed: 2026-04-20. [Online]. Available: https://openconnectivity.org/upnp-specs/UPnP-arch-DeviceArchitecture-v2.0-20200417.pdf

[49] J. Waldo, *The Jini Specifications*. Addison-Wesley Longman Publishing Co., Inc., 2000.

[50] J. J. Schussmann, L. Saxton, A. Testa, D. K. Sayre, and K. B. Leboeuf, "Bluetooth low energy (BLE) communication between a mobile device and a vehicle," US Patent Application Publication US20 170 164 192A1, Jun. 8, 2017. [Online]. Available: https://patents.google.com/patent/US20170164192A1/en

[51] S. Devalal and A. Karthikeyan, "LoRa Technology - An Overview," in *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, 2018, pp. 284–290.

[52] A. Banks, E. Briggs, K. Borg, and R. D. Benedetti, "MQTT Version 5.0," https://www.oasis-open.org/standard/mqtt-v5-0-cs02/, 2019, accessed: 2026-04-29.

[53] J. Ménétrey, M. Pasin, P. Felber, and V. Schiavoni, "WaTZ: A Trusted WebAssembly Runtime Environment with Remote Attestation for TrustZone," in *2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2022, pp. 1177–1189.

[54] P. Yuhala, J. Ménétrey, P. Felber, M. Pasin, and V. Schiavoni, "Fortress: Securing IoT Peripherals with Trusted Execution Environments," in *Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing*, 2024, pp. 243–250.

[55] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models," in *Proceedings of the 36th International Conference on Neural Information Processing Systems*, ser. NIPS '22. Red Hook, NY, USA: Curran Associates Inc., 2022.

[56] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing Reasoning and Acting in Language Models," in *International Conference on Learning Representations (ICLR)*, 2023.

[57] T. R. Sumers, S. Yao, K. Narasimhan, and T. L. Griffiths, "Cognitive Architectures for Language Agents," 2024. [Online]. Available: https://arxiv.org/abs/2309.02427

[58] Y. Shen, K. Song, X. Tan, D. Li, W. Lu, and Y. Zhuang, "HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face," in *Proceedings of the 37th International Conference on Neural Information Processing Systems*, ser. NIPS '23. Red Hook, NY, USA: Curran Associates Inc., 2023.

[59] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, W. X. Zhao, Z. Wei, and J. Wen, "A survey on large language model based autonomous agents," *Frontiers of Computer Science*, vol. 18, no. 6, p. 186345, 2024. [Online]. Available: https://doi.org/10.1007/s11704-024-40231-1

[60] Model Context Protocol, "Model Context Protocol Specification," https://modelcontextprotocol.io/specification/2025-11-25, Nov. 2025, accessed: 2026-04-24.

[61] A2A Project, "Agent2Agent Protocol," https://a2a-protocol.org/latest/, 2026, accessed: 2026-04-24.

[62] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, and T. Liu, "A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions," *ACM Trans. Inf. Syst.*, vol. 43, no. 2, Jan. 2025. [Online]. Available: https://doi.org/10.1145/3703155

[63] P. Sonune, R. Rai, S. Sural, V. Atluri, and A. Kundu, "LMN: A Tool for Generating Machine Enforceable Policies from Natural Language Access Control Rules using LLMs," 2025. [Online]. Available: https://arxiv.org/abs/2502.12460

[64] A. Vatsa, P. Patel, and W. Eiers, "Synthesizing Access Control Policies using Large Language Models," 2025. [Online]. Available: https://arxiv.org/abs/2503.11573

[65] Y. Cheng, M. Xu, Y. Zhang, K. Li, H. Wu, Y. Zhang, S. Guo, W. Qiu, D. Yu, and X. Cheng, "Say What You Mean: Natural Language Access Control with Large Language Models for Internet of Things," 2025. [Online]. Available: https://arxiv.org/abs/2505.23835

[66] S. H. Jayasundara, N. A. Gamagedara Arachchilage, and G. Russello, "SoK: Access Control Policy Generation from High-level Natural Language Requirements," *ACM Computing Surveys*, vol. 57, no. 4, Dec. 2024. [Online]. Available: https://doi.org/10.1145/3706057

[67] W. H. Winsborough, K. E. Seamons, and V. E. Jones, "Automated Trust Negotiation," in *DARPA Information Survivability Conference and Exposition*, vol. 2. Los Alamitos, CA, USA: IEEE Computer Society, Jan. 2000. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/DISCEX.2000.824965

[68] J. Richer and L. Johansson, "Vectors of Trust," RFC 8485, Oct. 2018. [Online]. Available: https://www.rfc-editor.org/rfc/rfc8485