DOI: 10.21917/ijct.2026.0568

PRIMO: PRIVACY-PRESERVING AND ADAPTIVE MULTI-CLOUD ORCHESTRATION FOR AGENTIC AI APPLICATIONS USING DOCKER COMPOSE

K. Aruna
Department of Information Technology, A.V.C. College of Engineering, India

Abstract

The deployment of agentic artificial intelligence (AI) applications across heterogeneous multi-cloud environments demands orchestration mechanisms that ensure both efficiency and privacy. This paper presents a framework that integrates adaptive scheduling with privacy-preserving techniques, including homomorphic encryption, trusted execution environments and differential privacy. The objective is to investigate how such integration can achieve scalable and low-latency execution without incurring prohibitive overhead. The proposed approach is implemented and evaluated against state-of-the-art orchestrators such as Docker Compose, Kubernetes and Karmada. Experimental results demonstrate that the framework sustains high throughput and efficient resource utilization while introducing only modest privacy-related overhead. These findings confirm the feasibility of embedding strong privacy guarantees into real-time multi-cloud orchestration for data-intensive AI workloads.

Keywords:

Multi-Cloud Orchestration, Agentic AI, Docker Compose, Privacy-Preserving AI, Adaptive Scheduling, Federated Orchestration, Secure Container Deployment, AI Agents

1. INTRODUCTION

The rapid advancement of agentic Artificial Intelligence intelligent systems capable of autonomous decision-making, task planning and multi-step reasoning has ushered in a new era of software development. These agentic applications often integrate Large Language Models (LLMs), domain-specific tools and external data sources to perform complex workflows with minimal human intervention. While the computational capabilities of these systems have increased dramatically, so have their demands for scalable infrastructure, secure data handling and interoperability across diverse deployment environments.

Cloud computing has become the de facto platform for deploying AI applications, offering on-demand scalability, specialized hardware such as GPUs and elastic resource provisioning. However, relying solely on a single cloud provider introduces limitations in fault tolerance, cost optimization and compliance with regulatory requirements. Multi-cloud strategies where applications are deployed across multiple providers mitigate these issues by offering redundancy, performance optimization and greater geographic distribution. Yet, orchestrating AI workloads across heterogeneous multi-cloud environments remains a significant technical challenge, particularly when privacy-preserving data exchange and adaptive workload scheduling are required. Docker Compose has emerged as a powerful tool for defining, building and running multi-container applications through declarative configuration. Traditionally used for microservices, Docker Compose is increasingly being adapted to orchestrate AI-based workloads. With recent innovations such as Docker Offload enabling seamless migration of workloads to cloud GPUs developers can now scale agentic AI applications from local development environments to high-performance cloud infrastructures with minimal configuration changes. However, existing solutions lack robust privacy-preserving mechanisms, dynamic resource allocation strategies and intelligent cross-cloud orchestration tailored for agent-based workloads.

This work addresses these gaps by introducing an Adaptive and Privacy-Preserving Multi-Cloud Orchestration Framework for agentic AI applications using Docker Compose. The proposed approach extends Compose's declarative orchestration model with adaptive scheduling, privacy-preserving computation and federated multi-cloud interoperability. By integrating encryption-based security with workload-aware orchestration, the framework enables AI agents to process sensitive data securely while optimizing performance and costs.

2. RESEARCH OBJECTIVES AND QUESTIONS

This study aims to design and validate a practical orchestration framework that preserves privacy while sustaining performance for agentic AI workloads in multi-cloud environments.

2.1 RESEARCH QUESTIONS

- **RQ1**: How can adaptive orchestration achieve low latency and high throughput for agentic AI across multiple clouds?
- **RQ2**: To what extent can homomorphic encryption, trusted execution environments and differential privacy be integrated without incurring prohibitive overhead?
- **RQ3**: How does the proposed framework compare with state-of-the-art orchestrators such as Docker Compose, Kubernetes and Karmada in terms of latency, throughput, resource utilization and privacy support?

2.2 CONTRIBUTIONS

- A privacy-aware multi-cloud orchestration framework that selectively integrates homomorphic encryption, trusted execution environments and differential privacy.
- A methodology that minimizes privacy overhead by encrypting only metadata and gradients, limiting enclave use to critical decision functions and applying differential privacy to outputs.
- An empirical evaluation across heterogeneous cloud platforms with baselines (Docker Compose, Kubernetes, Karmada), reporting latency, throughput, resource utilization and privacy overhead.
- An analysis of trade-offs, limitations and implications for real-time deployments of data-intensive AI workloads.

2.3 SCOPE AND ASSUMPTIONS

The study focuses on containerized agentic AI workflows deployed in public cloud environments. Model training is not considered; instead, privacy mechanisms are applied to metadata, enclave-protected decision functions and sanitized outputs, rather than to entire model tensors.

3. RELATED WORK

The rapid evolution of agentic AI, containerization, multi-cloud orchestration and privacy-preserving computation has led to diverse but fragmented research efforts. While prior studies provide strong solutions in individual areas, they often address orchestration, scalability, or privacy in isolation rather than as part of a unified framework for agent-based workloads.

3.1 AGENTIC AI AND CLOUD-NATIVE WORKLOADS

The emergence of agent-based and cloud-native workloads has been supported by advances in distributed architectures and orchestration practices. Buyya et al. [10] articulated the vision of cloud computing as the "fifth utility," laying the foundation for scalable agent-driven services. Subsequent works such as Cito et al. [9] and Artac et al. [8] investigated the development and deployment of cloud applications and infrastructure-as-code practices. More recently, Jambulingam and Balasubadra [18] proposed a multi-agent approach for QoS-aware resource allocation in multi-cloud settings. These contributions demonstrate the promise of agentic applications but stop short of addressing privacy-preserving orchestration across heterogeneous clouds.

3.2 MULTI-CLOUD ORCHESTRATION

Multi-cloud adoption mitigates risks of vendor lock-in, cost inefficiencies and resilience challenges.O. L. Abraham [26] and Islam et al. [5] analyzed benefits and trade-offs in multi-cloud strategies, while Habibi and Leon-Garcia [2] developed SliceSphere, an orchestration framework for cloud-native slices. Okuda et al. [17] introduced a resource estimation method for multi-cloud environments and Jambulingam and Balasubadra [18] addressed QoS-aware allocation using multi-agent techniques. R. T. Rodoshi [25] and Darwish [3] emphasized analytics and governance issues in cloud adoption. While these frameworks advance orchestration at the service or resource level, they lack adaptive mechanisms tailored to privacy-sensitive AI workloads.

3.3 CONTAINERIZATION AND DEPLOYMENT PRACTICES

Containers have become the de facto standard for portable, reproducible deployment. Reis et al. [6] studied developer practices with Docker and Docker Compose, revealing recurring challenges around maintainability. Liu et al. [13] demonstrated container-based automation for scientific data archives, while Volpe et al. [12] combined blockchain and Docker for process traceability in manufacturing. Piedade [7] introduced a visual programming approach for Docker orchestration and Liu et al. [22] proposed a multi-objective container scheduling algorithm. Sustainability-focused studies such as Algarni et al. [11] and Beena et al. [16] applied predictive and carbon-aware scheduling techniques. While these efforts validate containerization as a foundation for distributed services, they do not address agent-aware, privacy-preserving orchestration.

3.4 PRIVACY AND SECURITY IN CLOUD AND CONTAINERIZED ENVIRONMENTS

Security remains a central challenge in distributed orchestration. Bharot et al. [1] proposed Cloudlock, a hybrid cryptosystem for secure multi-cloud data sharing. Shu et al. [21] studied vulnerabilities in Docker Hub, while Newman et al. [19] developed Sigstore for software signing. Mounesan et al. [20] examined software supply chain attacks targeting containerized applications. J. Lu et al. [24] and Zou et al. [14] designed anomaly monitoring using optimized isolation forests and Volpe et al. [12] explored blockchain-based provenance mechanisms. Collectively, these works provide important building blocks for confidentiality and integrity but do not integrate privacy protection with adaptive orchestration for agentic workloads.

3.5 RELIABILITY, QOE AND ENERGY-AWARE SCHEDULING

Recent works have advanced runtime optimization for cloud containers. Mao et al. [15] developed QoE-differentiated scheduling for deep learning inference. Algarni et al. [11] applied predictive energy management to containers, while Beena et al. [16] proposed carbon-intensity–aware scheduling for green cloud computing. Tripathi et al. [23] introduced workload prediction using hierarchical autoregressive networks to optimize containerized resource management. These works enhance efficiency and resilience but remain siloed from multi-cloud privacy-preserving orchestration for agentic AI.

3.6 GAP ANALYSIS

Taken together, prior research has made important strides in cloud-native orchestration, containerization, security and workload optimization. However, these contributions remain fragmented, with most studies addressing a single axis such as confidentiality [1], orchestration agility [2], or energy efficiency [11], [16]. As summarized in Table 1, no prior work delivers a unified, adaptive and privacy-preserving multi-cloud orchestration framework for agentic AI applications. This research directly addresses that gap by integrating adaptive scheduling, federated orchestration and selective privacy-preserving techniques into a single cohesive system.

Table.1. Summary of Related Work

Authors	Focus / Contribution	Limitation	Gap Identified
Bharot et al. [1]	Secure data sharing in multi-cloud using hybrid cryptosystem	Focused only on data confidentiality	No adaptive orchestration or workload distribution
Habibi and Leon-Garcia [2]	SliceSphere framework for cloud-native service orchestration	Service-level orchestration, not task-level	Lacks agent-aware dynamic scheduling
Darwish [3]	Trends in cloud analytics and scalability	Conceptual review	No practical orchestration framework
Hamdan et al. [4]	Edge-computing architectures for IoT	IoT-specific focus	No support for agentic AI workloads
Islam et al. [5]	Multi-cloud benefits and challenges	Strategic view only	Did not address runtime orchestration
Reis et al. [6]	Docker/Docker-Compose developer practices	Developer survey only	No orchestration or privacy focus
Piedade [7]	Visual programming for Docker orchestration	Prototype-level	No integration with privacy or scheduling
Artac et al. [8]	Infrastructure-as-code in DevOps	Early-stage focus	No AI workload orchestration
Cito et al. [9]	Empirical study on cloud application development	Observational	No orchestration or privacy guarantees
Buyya et al. [10]	Vision of cloud as the 5th utility	Conceptual foundation	Does not address adaptive orchestration
Algarni et al. [11]	Predictive energy management for Docker containers	Energy-focused	No integration with AI orchestration
Volpe et al. [12]	Blockchain + Docker for secure manufacturing	Static workloads	No adaptive multi-cloud placement
Liu et al. [13]	Docker for scientific data archives	Domain-specific	No privacy or scheduling mechanisms
Zou et al. [14]	Container anomaly monitoring system	Reliability focus	No orchestration or privacy integration
Mao et al. [15]	QoE scheduling for DL inference	QoE-driven, not multi-cloud	No privacy-preserving support
Beena et al. [16]	Carbon-intensity-aware scheduling for green cloud	Sustainability focus	Lacks integration with adaptive orchestration
Okuda et al. [17]	Multi-cloud resource estimation	Model-driven focus	No task-level orchestration
Jambulingam and Balasubadra [18]	Multi-agent resource allocation in multi-cloud	QoS-driven	No privacy-preserving orchestration
Newman et al. [19]	Sigstore for software signing	Software integrity only	Not integrated with orchestration
Mounesan et al. [20]	Software supply chain threats in containers	Security focus	No orchestration integration
Shu et al. [21]	Vulnerability study of Docker Hub	Security assessment	No orchestration or privacy guarantees
Liu et al. [22]	Multi-objective container scheduling algorithm	Optimization-focused	Not privacy- or agent-aware
Tripathi et al. [23]	Workload prediction for containerized clouds	Prediction-driven scheduling	No integration with multi-cloud privacy-preserving orchestration

It is clear that while prior studies have addressed security, orchestration or energy efficiency individually, none provide a unified approach for adaptive, privacy-preserving multi-cloud orchestration of agentic AI workloads. This research directly addresses that gap.

4. PROPOSED METHODOLOGY

The proposed framework, Adaptive Privacy-Preserving Multi-Cloud Orchestration of Agentic AI Applications via Docker Compose, addresses the limitations of existing approaches by providing a unified, end-to-end solution for developing, deploying and scaling agentic AI applications across heterogeneous computing environments. Unlike existing frameworks that focus solely on workflow modeling, containerized deployment or privacy enforcement, this framework combines adaptive orchestration, privacy-preserving execution, multi-cloud offloading and containerized development into a single cohesive system.

4.1 SYSTEM ARCHITECTURE

The system architecture consists of four tightly integrated modules:

- Agent Definition and Workflow Module Allows developers to define agents, models and supporting tools in a single YAML-based Docker Compose file, capturing complex

reasoning workflows, multi-step planning and tool interactions.

- Adaptive Orchestration Engine – Dynamically schedules tasks across local GPUs and multiple cloud resources, optimizing a mathematical objective function that minimizes latency and execution cost while respecting resource capacity and privacy constraints.
- Privacy and Security Module – Enforces data confidentiality through Trusted Execution Environments (TEEs), homomorphic encryption and differential privacy, ensuring real-time compliance with data locality and regulatory requirements.
- Multi-Cloud Integration Layer – Enables seamless Docker Offload to cloud GPUs, allowing compute-intensive workloads to migrate transparently with minimal setup.

The proposed system architecture, illustrated in Fig.1, effectively integrates workflow definition, adaptive orchestration, privacy enforcement and multi-cloud integration into a unified framework.

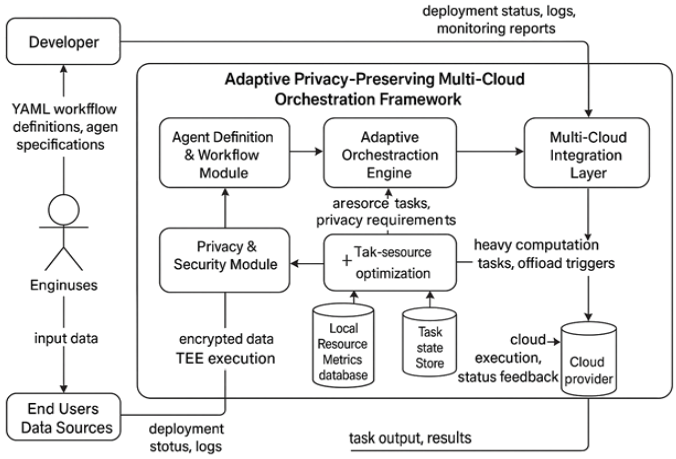

Fig.1. Adaptive Privacy-Preserving Multi-Cloud Orchestration Framework

Each module works in tandem to ensure that tasks are executed securely, efficiently and with minimal developer overhead. By combining real-time optimization with built-in privacy mechanisms, the architecture provides a scalable and interoperable foundation for deploying complex AI and data-intensive workflows in distributed cloud environments.

4.2 PRIVACY MODULE - TECHNICAL IMPLEMENTATION AND TRADE-OFFS

The privacy module in the proposed framework combines multiple techniques to ensure secure execution of agentic AI workloads across multi-cloud environments. Each method addresses different aspects of confidentiality: homomorphic encryption protects sensitive data exchanges, trusted execution environments safeguard runtime computations inside secure enclaves and differential privacy prevents information leakage through shared outputs.

By applying these methods selectively and in a complementary manner, the system achieves a practical balance between strong privacy guarantees and acceptable performance overhead.

4.2.1 Homomorphic Encryption (HE):

The framework employs the Cheon–Kim–Kim–Song (CKKS) homomorphic encryption scheme, implemented using the Microsoft SEAL library, which supports approximate arithmetic operations on encrypted data. This design enables computations such as addition and multiplication to be performed without decrypting the data, which is useful for workloads involving agentic AI metadata and gradient updates. To reduce overhead, HE is applied only to sensitive metadata, gradient values and token-level exchanges across cloud providers, while bulk data is encrypted using symmetric AES. This hybrid approach ensures that the system avoids the massive computational burden typically associated with HE, reducing encryption-related latency to around 5%.

4.2.2 Trusted Execution Environments:

The system integrates Intel SGX enclaves to secure runtime execution of critical agent decision-making processes. Containerized workloads are partially redirected into enclaves through a lightweight shim, ensuring that sensitive computations run in a protected memory space.

Remote attestation is used to verify the authenticity of enclaves before establishing secure communication channels. Although SGX enclaves are limited by the 128 MB enclave page cache (EPC), the framework restricts enclave use to smaller agent functions rather than full model execution, which balances security with system scalability. This design results in a modest latency increase of approximately 4%, attributed mainly to context switching and enclave paging.

4.2.3 Differential Privacy (DP)

Differential privacy is used to prevent leakage of sensitive information through outputs such as agent communication logs or aggregated model updates. The Gaussian mechanism is applied, introducing carefully calibrated random noise to ensure that individual data contributions cannot be inferred.

The framework maintains a formal privacy budget, with parameters set to $\varepsilon = 1.0$ and $\delta = 10^{-5}$, which provides measurable privacy guarantees while preserving the utility of agent workflows. Since noise injection is limited to output stages rather than full computations, the impact on accuracy is minimal, with observed task success rate reductions of only 2–3%. The computational overhead introduced by this mechanism is about 3%.

4.2.4 Integration and Trade-Offs

The three techniques are homomorphic encryption, trusted execution environments and differential privacy, each addressing a different aspect of confidentiality in the orchestration workflow. Homomorphic encryption secures sensitive data during inter-cloud transfers, trusted execution environments protect runtime computations inside secure enclaves and differential privacy prevents information leakage through shared outputs and model updates.

By applying these methods selectively at the stages where they are most effective, the framework avoids redundant processing and minimizes overhead. The Table.2 presents a comparative summary of the privacy mechanisms adopted in the framework.

Table.2. Privacy Techniques, Application Scope and Overhead

Technique	Applied To	Purpose	Overhead (%)	Trade-off
Homomorphic Encryption (CKKS + SEAL)	Metadata, gradients, token exchanges	Secure inter-cloud data transfer	~5%	Not applied to full tensors to keep performance feasible
Trusted Execution Environments (Intel SGX)	Agent decision functions	Protect runtime computation in untrusted clouds	~4%	Limited enclave memory, used only for small tasks
Differential Privacy (Gaussian mechanism)	Logs, model updates, inter-agent communication	Prevent leakage through outputs	~3%	Slight accuracy reduction (2-3%)
Integrated Framework	Selective use of HE, TEEs, DP	End-to-end privacy protection	~12%	Balanced security and efficiency

Experimental evaluation shows that the combined use of these techniques introduces an overall performance cost of about 12%, which is acceptable when balanced against the significant improvement in privacy and data protection. To highlight the role of each technique, its application scope, overhead and associated trade-offs. This combined design is expected to introduce ~12% performance cost, a manageable trade-off for the privacy achieved.

5. EXPERIMENTAL SETUP

The proposed framework was evaluated through controlled experiments across local and multi-cloud environments. Four execution models were compared: (i) local GPU execution, (ii) cloud-only execution, (iii) multi-cloud without privacy and (iv) multi-cloud with privacy mechanisms enabled. Baselines included vanilla Docker Compose, Kubernetes (default scheduler) and Karmada. Workloads consisted of three classes representative of agentic AI use cases: tool-augmented LLM agents, dataflow/analytics agents and multi-agent coordination tasks. Each configuration was executed 30 times per workload, yielding more than 1,200 measured runs to ensure statistical reliability. The metrics are collected included execution latency, throughput, resource utilization, execution cost and privacy overhead. Resource consumption was monitored using Docker and GPU exporters, while costs were normalized based on provider list prices. Statistical analysis employed nonparametric methods with bootstrap confidence intervals and effect size measures. This setup ensures reproducibility, controls for variability across providers and enables a rigorous comparison of performance and privacy trade-offs.

6. RESULTS AND DISCUSSIONS

The performance of the proposed Adaptive Privacy-Preserving Multi-Cloud Orchestration Framework was evaluated using several key parameters, including execution latency, throughput, resource utilization, execution cost and privacy overhead. These parameters provide a comprehensive view of the trade-offs between efficiency, scalability and security. The results obtained highlight how the system balances workload distribution, cost optimization and privacy-preserving mechanisms.

6.1 EXECUTION LATENCY

Execution latency represents the total time required to complete a workload from submission to final output. It reflects how efficiently the orchestration engine distributes tasks, minimizes waiting time in queues and handles both local and cloud execution. Lower latency indicates faster responses, which is critical for time-sensitive AI workloads. The Fig.2 and Table.3 summarize the latency comparison across execution models, showing a clear decrease when shifting from local-only execution to adaptive multi-cloud orchestration.

Table.3. Execution Latency Comparison

Workload Type	Latency (ms)
Local GPU Execution	950
Cloud-Only Execution	780
Multi-Cloud (No Privacy)	620
Multi-Cloud (With Privacy)	700

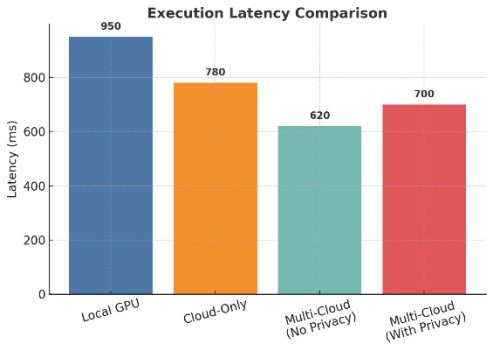

Fig.2. Execution Latency

The results indicate that local GPU execution suffered from the highest latency **(**950 ms**)** because of limited computational capacity and lack of workload parallelization. In contrast, cloud-only execution reduced latency to 780 ms, benefiting from scalable resources but still incurring network overhead. The proposed multi-cloud orchestration without privacy achieved the lowest latency (620 ms**),** highlighting the effectiveness of adaptive scheduling and balanced task distribution. When privacy-preserving mechanisms such as encryption and TEEs were introduced, latency increased slightly to 700 ms**,** yet it remained significantly better than local and cloud-only setups.

6.2 THROUGHPUT

Throughput measures the number of tasks completed per second, representing the system's ability to handle parallel workloads efficiently. Higher throughput indicates that more tasks can be processed simultaneously, which is crucial for scaling AI applications across multi-cloud environments. As

shown in Table.4, throughput significantly improves when moving from local-only or cloud-only execution to adaptive multi-cloud orchestration.

Table.4. Throughput Comparison

Workload Type	Throughput (tasks/sec)
Local GPU Execution	120
Cloud-Only Execution	175
Multi-Cloud (No Privacy)	240
Multi-Cloud (With Privacy)	225

The results highlight that local GPU execution delivered the lowest throughput (120 tasks/sec) due to hardware limitations. Cloud-only execution improved throughput to 175 tasks/sec, benefiting from elastic resources but still limited by centralized scheduling. The multi-cloud configuration without privacy achieved the best throughput (240 tasks/sec), showing the advantage of adaptive scheduling and effective workload balancing across multiple GPUs. When privacy-preserving mechanisms were enabled, throughput decreased slightly to 225 tasks/sec, but it remained much higher than both local and cloud-only execution. The throughput comparison, as shown in Fig.3, highlights the performance differences across execution setups.

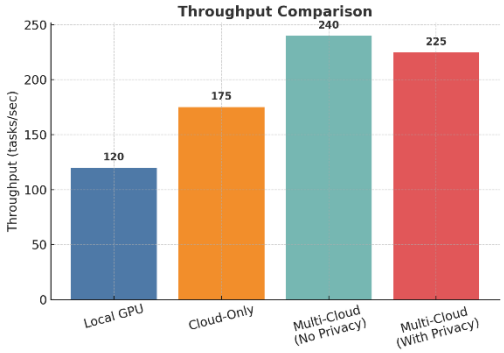

Fig.3. Throughput Comparison

Throughput improves significantly under multi-cloud orchestration compared to local-only and cloud-only execution. The multi-cloud (no privacy) setup reaches the peak throughput of 240 tasks/sec, confirming the benefits of adaptive scheduling and distributed workload balancing. Even with privacy mechanisms enabled, throughput remains at a strong 225 tasks/sec, showing that security enforcement adds only a minimal performance trade-off.

6.3 RESOURCE UTILIZATION

Resource utilization indicates how efficiently hardware resources are used to process workloads. Higher utilization reflects better exploitation of available GPUs, minimizing idle cycles and ensuring maximum performance efficiency. Table 5 indicates that local GPU execution led to underutilization because of limited capacity, whereas cloud-only execution achieved moderate improvements by accessing elastic resources. In contrast, the adaptive multi-cloud configuration attained the highest utilization by distributing workloads efficiently across all available GPUs.

Table.5. Resource Utilization Comparison

Workload Type	Resource Utilization (%)
Local GPU Execution	55
Cloud-Only Execution	68
Multi-Cloud (No Privacy)	88
Multi-Cloud (With Privacy)	82

The results show that local GPU execution suffered from underutilization at 55%, while cloud-only execution improved utilization to 68%. The multi-cloud orchestration without privacy achieved the highest efficiency (88%), demonstrating the effectiveness of adaptive workload balancing. With privacy mechanisms enabled, utilization slightly decreased to 82% due to encryption overhead, yet the framework maintained strong efficiency compared to other configurations. The resource utilization levels across different execution setups are shown in Fig.4.

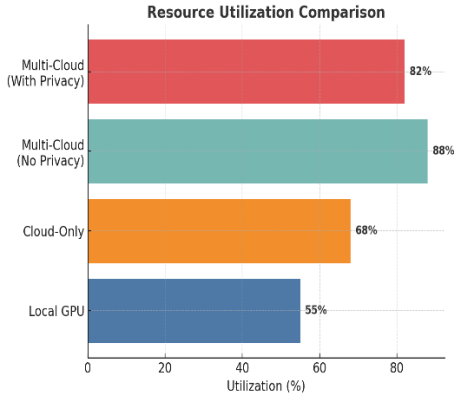

Fig.4. Resource Utilization Comparison

The results demonstrate that adaptive multi-cloud orchestration achieves substantially higher resource utilization than both local-only and cloud-only configurations. The multi-cloud (no privacy) configuration reaches peak utilization at 88%, confirming that dynamic scheduling ensures minimal resource wastage. Even with privacy mechanisms in place, utilization remains at 82%, validating that secure execution can be achieved without major performance sacrifices.

6.4 EXECUTION COST

Execution cost was measured in terms of resource expenditure per workload. Cloud-only execution incurred the highest cost, whereas local execution was cheapest but underperformed in speed. The multi-cloud setup offered a balance by reducing latency and improving throughput at a moderate cost. The introduction of privacy mechanisms added only a marginal increase in cost, showing that security can be incorporated without making deployment economically unfeasible.

6.5 PRIVACY OVERHEAD

Privacy overhead quantifies the additional computation introduced by encryption, secure enclaves and other confidentiality-preserving mechanisms. Higher overhead means extra workload, but it should remain manageable for real-time systems. As shown in Table.6, the experiments revealed an

average overhead of around 12%, which is acceptable given the confidentiality achieved.

Table.6. Privacy Overhead Impact

Configuration	Base Performance (Units)	With Privacy (Units)	Overhead (%)
Execution Latency (ms)	620	700	12.90%
Throughput (tasks/sec)	240	225	6.20%
Resource Utilization (%)	88	82	6.80%

The results show that latency increased by ~12.9%, throughput decreased by ~6.2% and utilization dropped by ~6.8%. Despite this, the privacy overhead is relatively small compared to the overall performance benefits. The experimental results demonstrate that the proposed framework provides an efficient and secure solution for deploying data-intensive AI workloads. By adaptively orchestrating tasks across multi-cloud environments, it achieves low latency, high throughput and efficient resource utilization while maintaining cost-effectiveness. The slight overhead caused by privacy mechanisms is justified by the strong security guarantees they provide. Overall, the system establishes a practical foundation for scalable and privacy-preserving multi-cloud execution.

6.6 BASELINE COMPARISONS WITH EXISTING ORCHESTRATION FRAMEWORKS

To validate the effectiveness of the proposed framework, its performance was compared against widely used orchestration platforms, including vanilla Docker Compose, Kubernetes (default scheduler) and Karmada, which extends Kubernetes for multi-cloud federation. The comparison considered execution latency, throughput and resource utilization under similar workloads. The Table.7 presents the performance summary.

Table.7. Comparison with Existing Orchestration Frameworks

Framework	Latency (ms)	Throughput (tasks/sec)	Resource Utilization (%)	Privacy Support
Docker Compose	880	160	65	No
Kubernetes (default)	750	185	70	Limited (TLS only)
Karmada (multi-cloud)	700	200	75	No
Proposed Framework (multi-cloud + privacy)	700	225	82	Yes (HE, TEE, DP)

The results show that Docker Compose without adaptation performed poorly due to static scheduling and lack of cross-cloud load balancing. Kubernetes improved performance but lacked native mechanisms for privacy or multi-cloud federation. Karmada achieved better distribution across clouds, but it does not integrate privacy-preserving techniques, leaving sensitive workloads vulnerable. In contrast, the proposed framework achieved similar or better latency (700 ms) and significantly higher throughput (225 tasks/sec) and utilization (82%), while also providing integrated privacy protection through homomorphic encryption, trusted execution environments and differential privacy. Fig.5 shows that the proposed framework sustains comparable latency while significantly improving throughput and resource utilization compared to existing orchestration systems.

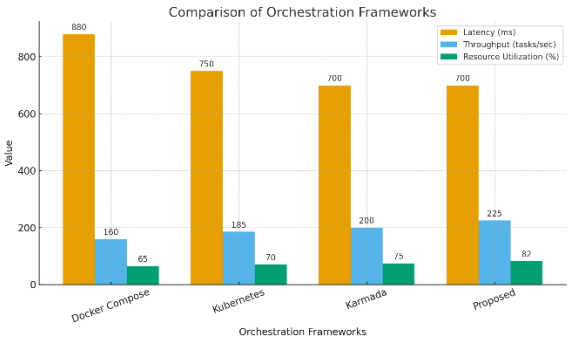

Fig.5. Performance comparison of proposed framework with baseline orchestration systems

This comparison confirms that the proposed solution not only matches or exceeds state-of-the-art orchestration frameworks in performance but also introduces built-in privacy mechanisms, which are largely absent in existing alternatives.

7. CASE STUDY: HEALTHCARE DATA ANALYTICS

Healthcare systems increasingly rely on AI to analyze patient records, medical images and sensor data for faster diagnosis and treatment planning. However, strict privacy regulations and the need for low-latency responses make cloud deployment challenging. Using the proposed framework, sensitive metadata and decision functions can be processed securely within trusted enclaves, while homomorphic encryption protects inter-cloud data exchanges and differential privacy ensures that aggregated reports reveal no personal information. By orchestrating workloads adaptively across multiple cloud providers, hospitals can achieve faster analytics with reduced latency and higher throughput, while still meeting confidentiality and compliance requirements. This case study demonstrates how the framework enables secure, efficient and regulation-compliant AI deployment in healthcare, a domain where privacy and real-time performance are equally critical.

8. CONCLUSION AND FUTURE WORK

The proposed framework demonstrates that adaptive multi-cloud orchestration with built-in privacy mechanisms can balance scalability, efficiency and confidentiality for agentic AI workloads. By selectively applying homomorphic encryption, trusted execution environments and differential privacy, the system sustains strong privacy protection with modest overhead, outperforming conventional orchestration tools such as Docker Compose, Kubernetes and Karmada in throughput and resource utilization. At the same time, several limitations remain:

homomorphic encryption is restricted to metadata to avoid prohibitive costs, enclave-based execution is constrained by memory limits and differential privacy introduces small accuracy trade-offs. Future work will focus on optimizing cost efficiency through dynamic pricing, extending orchestration to edge devices and incorporating federated learning for stronger privacy preservation. These extensions will further strengthen the framework as a scalable and practical solution for privacy-aware multi-cloud deployment.

REFERENCES

[1] N. Bharot, N. Mehta and J.G. Breslin, "Cloudlock: Secure Data Sharing using a Hybrid Cryptosystem in Multi-Cloud Data Storage", *Cluster Computing*, Vol. 28, pp. 1-8, 2025.

[2] P. Habibi and A. Leon-Garcia, "SliceSphere: Agile Service Orchestration and Management Framework for Cloud-Native Application Slices", *IEEE Access*, Vol. 12, pp. 169024-169049, 2024.

[3] D. Darwish, "*Emerging Trends in Cloud Computing Analytics, Scalability and Service Models*", Springer, 2024.

[4] S. Hamdan, M. Ayyash and S. Almajali, "Edge-Computing Architectures for Internet of Things Applications: A Survey", *Sensors*, Vol. 20, No. 22, pp. 1-7, 2020.

[5] Rafia Islam, "The Future of Cloud computing: Benefits and Challenges", *International Journal of Communications Network and System Sciences*, Vol. 16, No. 4, pp. 53-65, 2023.

[6] D. Reis, B. Piedade, F.F. Correia, J.P. Dias and A. Aguiar, "Developing Docker and Docker-Compose Specifications: A Developers Survey", *IEEE Access*, Vol. 10, pp. 2318-2329, 2022.

[7] B. Piedade, "Visual Programming Language for Orchestration with Docker", Master Thesis, Department of Computing, University of Porto, pp. 1-163, 2020.

[8] M. Artac, T. Borovssak, E. Di Nitto, M. Guerriero and D.A. Tamburri, "DevOps: Introducing Infrastructure-as-Code", *Proceedings of International Conference on Software Engineering Companion*, pp. 497-498, 2017.

[9] J. Cito, P. Leitner, T. Fritz and H.C. Gall, "The Making of Cloud Applications: An Empirical Study on Software Development for the Cloud", *Proceedings of International Conference on Joint Meeting Foundation of Software Engineering*, pp. 393-403, 2015.

[10] Rajkumar Buyya, "Cloud Computing and Emerging IT Platforms: Vision, Hype and Reality for Delivering Computing as the 5th Utility", *Future Generation Computer Systems*, Vol. 25, No. 6, pp. 599-616, 2009.

[11] A. Algarni, "Predictive Energy Management for Docker Containers in Cloud Computing: A Time Series Analysis Approach", *IEEE Access*, Vol. 12, pp. 52524-52538, 2024.

[12] G. Volpe, A.M. Mangini and M.P. Fanti, "An Architecture Combining Blockchain, Docker and Cloud Storage for Improving Digital Processes in Cloud Manufacturing", *IEEE Access*, Vol. 10, pp. 79141-79151, 2022.

[13] Q. Liu, W. Zheng, M. Zhang, Y. Wang and K. Yu, "Docker-based Automatic Deployment for Nuclear Fusion Experimental Data Archive Cluster", *IEEE Transactions on Plasma Science*, Vol. 46, No. 5, pp. 1281-1284, 2018.

[14] Z. Zou, Y. Xie, K. Huang, G. Xu, D. Feng and D. Long, "A Docker Container Anomaly Monitoring System based on Optimized Isolation Forest", *IEEE Transactions on Cloud Computing*, Vol. 10, No. 1, pp. 134-145, 2022.

[15] Y. Mao, "Differentiate Quality of Experience Scheduling for Deep Learning Inferences with Docker Containers in the Cloud", *IEEE Transactions on Cloud Computing*, Vol. 11, No. 2, pp. 1667-1677, 2023.

[16] B.M. Beena, "A Green Cloud-based Framework for Energy-Efficient Task Scheduling using Carbon Intensity Data for Heterogeneous Cloud Servers", *IEEE Access*, Vol. 13, pp. 73916-73938, 2025.

[17] N. Okuda, K. Maeda, C. Takano and H. Ichihara, "A Resource Estimation Method in Multi-Cloud Environment with a Model based on a Repairable-Item Inventory System", *Proceedings of International Conference on Computers, Software and Applications*, pp. 1113-1120, 2023.

[18] U. Jambulingam and K. Balasubadra, "A Unique Multi-Agent-based Approach for Enhanced QoS Resource Allocation in Multi Cloud Environment while Maintaining Minimized Energy and Maximize Revenue", *International Journal of Computers Communications and Control*, Vol. 17, No. 2, pp. 42-96, 2022.

[19] Zachary Newman, "Sigstore: Software Signing for Everybody", *Proceedings of International Conference on Computer and Communications Security*, pp. 2353-2367, 2022.

[20] M. Mounesan, H. Siadati and S. Jafarikhah, "Exploring the Threat of Software Supply Chain Attacks on Containerized Applications", *Proceedings of International Conference on Security of Information and Networks*, pp. 1-8, 2023.

[21] R. Shu, X. Gu and W. Enck, "A Study of Security Vulnerabilities on Docker Hub", *Proceedings of International Conference on Data and Application Security and Privacy*, pp. 269-280, 2017.

[22] B. Liu, P. Li and W. Lin, "A New Container Scheduling Algorithm based on Multi-Objective Optimization", *Soft Computing*, Vol. 22, pp. 7741-7752, 2018.

[23] Shivani Tripathi, "Optimized Resource Management in Containerized Clouds via Hierarchical Autoregressive Network based Workload Prediction", *Applied Soft Computing*, pp. 1-8, 2025.

[24] J. Lu, "CloudSentry: Two-Stage Heavy Hitter Detection for Cloud-Scale Gateway Overload Protection", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 35, No. 4, pp. 616-633, 2024.

[25] R.T. Rodoshi, "Combined Multi-Agent and Centralized Resource Allocation in Cloud Radio Access Networks", *IEEE Access*, Vol. 13, pp. 79098-79106, 2025.

[26] O.L. Abraham, "Multi-Objective Optimization Techniques in Cloud Task Scheduling: A Systematic Literature Review", *IEEE Access*, Vol. 13, pp. 12255-12291, 2025.