Anusandhanvallari
Vol 2025, No.1
January 2025
ISSN 2229-3388

Dynamic Technique to Overcome Service Unavailability in Multi-Cloud Environments

[1]Kushala M. V., [2]Dr. B. S. Shylaja

[1]Assistant Professor, Department of Artificial Intelligence & Machine Learning

Dr. Ambedkar Institute of Technology, Bengaluru, India

[2]Research Supervisor, Department of Information Science & Engineering

Dr. Ambedkar Institute of Technology, Bengaluru, India

Abstract

Ensuring uninterrupted service delivery across distributed cloud infrastructures remains one of the foremost challenges confronting modern enterprises. Unplanned outages impose cascading consequences—operational paralysis, revenue erosion, and deteriorated customer experience—with particularly acute effects in latency-sensitive domains such as healthcare informatics, financial services, and e-commerce platforms. This paper introduces the Dynamic Technique to Overcome Service Unavailability in Multi-Cloud Environments(DTOSUME), a novel methodology designed to simultaneously maximize service resilience and strengthen security posture across heterogeneous cloud provider ecosystems. The proposed framework employs network-graph-based simulation to characterize system behavior under adversarial conditions, including induced network partitions and complete datacenter failures. Experiments were conducted across three deployment scales—30, 50, and 100 nodes—and outcomes were benchmarked against equivalent single-cloud deployments. Empirical results confirm that the Dynamic Technique to Overcome Service Unavailability in Multi-Cloud Environments approach achieves a 15.5% reduction in mean response latency (23.5 ms versus 27.8 ms), a 39.5% improvement in packet delivery reliability (2.3% versus 3.8% loss), and a 33.3% reduction in failover incidents compared to single-provider configurations. Sub-linear latency scaling was observed with growing network density, while recovery time exhibited exponential growth, underscoring the need for adaptive failover heuristics at scale. These findings yield actionable architectural guidance for resilience engineers and cloud architects, and contribute a rigorous quantitative foundation to the evolving body of knowledge on multi-cloud availability and security optimization.

Index Terms—multi-cloud computing, service availability, cloud security, network partition, datacenter failure, availability optimization, graph-based simulation, cloud resilience, failover mechanisms.

I. Introduction

The rapid proliferation of cloud-native applications has propelled multi-cloud adoption from an architectural curiosity to an enterprise-grade imperative. Organizations increasingly rely on services drawn from two or more competing cloud providers, motivated by the desire to avoid vendor lock-in, optimize cost–performance trade-offs, and exploit the differentiated capabilities of distinct platforms [1]. Industry data underscore this trend: global expenditure on public cloud services climbed from USD 242.7 billion in 2020 to USD 304.9 billion in 2021—

representing 25.6% year-over-year growth—and was projected to surpass USD 362.3 billion by 2022 [4]. These figures reflect not merely a cyclical uptick but a structural transformation in how computing infrastructure is conceived and consumed.

Despite its manifest advantages, the multi-cloud paradigm introduces a correspondingly intricate operational landscape. Maintaining coherent service availability across geographically dispersed providers with divergent APIs, heterogeneous security models, and independent fault domains is substantially more demanding than managing a monolithic single-cloud deployment [2]. Service disruptions in this context carry multiplicative risk: every additional provider boundary represents a potential failure plane, and complex inter-service dependencies can propagate localized faults into system-wide outages. The financial stakes are considerable—the Information Technology Intelligence Consulting (ITIC) estimates the hourly cost of enterprise downtime at USD 300,000 or more [8]—while secondary consequences such as reputational damage and regulatory exposure compound the urgency of robust availability engineering [9].

Cloud service providers and independent security vendors have responded with a growing portfolio of availability-assurance technologies, including health-monitoring agents, automated failover orchestration, geo-redundant data replication, and policy-enforcement platforms such as Cloud Access Security Brokers (CASBs) and Cloud Security Posture Management (CSPM) tools [10][12]. Nevertheless, organizations frequently struggle to translate these discrete capabilities into a coherent, end-to-end availability strategy that spans multiple providers consistently and cost-effectively. The absence of a principled optimization framework that jointly considers availability, security posture, and economic constraints leaves practitioners reliant on ad-hoc configurations that may be adequate under normal conditions but brittle under stress.

This paper addresses that gap through three principal contributions: (i) a formal characterization of service allocation across multi-cloud providers that incorporates availability, security, and cost as first-class optimization objectives[39]; (ii) the Dynamic Technique to Overcome Service Unavailability in Multi-Cloud Environments, which employs dynamic replication and real-time load rebalancing to sustain service continuity during provider failures; and (iii) comprehensive simulation experiments—spanning network partition and full datacenter failure scenarios at multiple scales—that quantify Dynamic Technique to Overcome Service Unavailability in Multi-Cloud Environments performance advantages over single-cloud baselines. The remainder of this paper is organized as follows: Section II surveys relevant literature on multi-cloud management and service availability; Section III details the proposed methodology and algorithm; Section IV describes the experimental test scenarios; Section V presents results and analysis; Sections VI and VII contain discussion and conclusions respectively.

II. Literature Review

Multi-cloud computing, broadly defined as the strategic consumption of services from two or more independent cloud providers, has attracted sustained scholarly attention owing to its potential to optimize resource costs, eliminate vendor dependency, and leverage platform-specific strengths [3]. A representative configuration might combine Amazon Web Services (AWS) for high-throughput computation, Google Cloud Platform for advanced analytics pipelines, and Microsoft Azure for enterprise productivity integration. While Gartner acknowledges the operational and strategic benefits of this architectural pattern, it simultaneously flags heightened complexity as a primary inhibitor—particularly with respect to maintaining uniform security enforcement and ensuring seamless cross-provider service integration [4].

The challenge of service unavailability is especially acute in distributed environments. A 2021 survey by the Uptime Institute revealed that 31% of data centers experienced measurable downtime within the reporting period, attributing the majority of incidents to human error and configuration drift [7]. In multi-cloud deployments, the probability of service disruption scales with the number of provider boundary crossings and the depth of inter-service dependency graphs [6]. These observations motivate the development of automated orchestration and resilience mechanisms that can respond to emerging failure conditions without human intervention.

Research into multi-cloud orchestration has explored several complementary strategies. Geo-redundant deployment—wherein application state and data are replicated across spatially separated availability zones belonging to different providers—has demonstrated measurable improvements in both recovery point objectives (RPO) and recovery time objectives (RTO) [14]. Intelligent workload scheduling algorithms capable of dynamically reallocating computation in response to real-time cost, performance, and availability signals have also received considerable attention [15][16]. More recently, cloud-native paradigms—particularly container orchestration platforms and microservices architectures—have provided the compositional primitives necessary to shift workloads across providers with minimal friction [17][18].

From a security perspective, Cloud Workload Protection Platforms (CWPPs) provide runtime visibility and enforcement for containerized workloads deployed across heterogeneous IaaS environments [11], while CSPM tools continuously audit configuration state against compliance baselines [12]. However, existing solutions predominantly address security and availability as orthogonal concerns, optimizing each in isolation. The present work departs from that convention by treating security, availability, and cost as jointly optimizable objectives within a unified mathematical framework, and by providing empirical validation under realistic multi-cloud failure conditions.

III. Proposed Methodology

The research adopts a mixed-methods experimental design that integrates quantitative performance measurement with structured qualitative assessment. A controlled multi-cloud testbed was constructed to replicate representative enterprise deployment topologies, enabling systematic evaluation of security and availability mechanisms under a range of fault conditions. Quantitative indicators—including response latency, packet delivery ratio, throughput, and failover frequency—were supplemented by expert review of system behavior during failure induction and recovery phases. This combination of empirical rigor and contextual interpretation supports holistic conclusions that neither perspective alone could sustain.

A. Research Design

The study followed a phased experimental protocol. Phase 1 established baseline infrastructure across three cloud service providers using Infrastructure-as-Code (IaC) tooling to guarantee reproducible configurations. Phase 2 deployed the DMCSAO algorithm and supporting security components. Phase 3 executed five structured failure scenarios while collecting real-time telemetry. Phase 4 aggregated and analysed the resulting datasets against pre-defined performance benchmarks. This sequential structure ensured that each experimental run was conducted under identical initial conditions, thereby controlling for confounding variables that commonly undermine cross-run comparability in distributed systems research [19].

B. Multi-Cloud Security Services Evaluation Framework

The evaluation framework is organized into four cooperating layers: infrastructure, security services, testing, and analysis. Together, these layers provide the architectural scaffold within which the Dynamic Technique to Overcome Service Unavailability in Multi-Cloud Environments operates.

Infrastructure Layer: Three major Cloud Service Providers (CSPs) form the substrate of the evaluation environment. Each provider was provisioned with standardized compute instances, storage volumes, and networking components to establish a consistent performance baseline [20]. Terraform-based IaC scripts ensured that configurations remained identical across experimental runs and that any drift from the desired state could be detected and remediated automatically. Within each CSP, resources were distributed across three availability zones, creating an intra-provider redundancy tier that independently of the cross-provider redundancy managed by Dynamic Technique to Overcome Service Unavailability in Multi-Cloud Environments [21].

Security Services Layer: Four security mechanisms are deployed uniformly across all providers: identity-centric access control, data-at-rest and in-transit encryption, continuous operational monitoring, and automated failover triggering. The implementation adheres to a defense-in-depth philosophy, such that compromise or degradation of any single mechanism does not fully expose the system to the threats that the remaining mechanisms address. Prior research indicates that layered security architectures of this type reduce the incidence of successful breach attempts by up to 85% in multi-cloud contexts [23].

Testing Layer: Resilience is evaluated through four canonical failure scenarios—network partition, datacenter failure, application-level disruption, and load-balancer failure—each executed via automated test harnesses that guarantee consistent stimulus delivery and faithful results capture. The scenarios are designed to probe incrementally severe fault conditions, from partial bandwidth degradation to complete zone unavailability, enabling observation of system behavior across the full spectrum of realistic failure modes [24][25].

Analysis Layer: An integrated observability stack, combining Prometheus for metric ingestion and Grafana for visualization, captures performance and security telemetry at sub-second granularity. The analysis layer computes RTO and RPO alongside cost-efficiency indicators, facilitating multi-criteria evaluation that extends beyond raw performance to encompass economic viability [26][27].

C. The DTOSUME Algorithm

The Dynamic Multi-Cloud Security and Availability Optimization algorithm addresses the service placement problem by treating availability, security, and cost as a weighted multi-objective function. For each service, the algorithm identifies an optimal set of provider placements, then iteratively adjusts the replication factor until load is sufficiently balanced across providers. A companion DynamicReallocation function responds to provider failures by instantaneously remapping affected services to healthy providers. The formal specification is presented in Algorithm 1.

Algorithm 1: Dynamic Multi-Cloud Security and Availability Optimization (DMCSAO)

Input:

$P = \{p_1, p_2, \ldots, p_n\}$: Set of cloud providers

$S = \{s_1, s_2, \ldots, s_m\}$: Set of services

```
  R = {r₁, r₂, ..., rₖ} : Set of resources
  A(p) : Availability function for provider p
  S(p) : Security function for provider p
  C(p,r) : Cost function for provider p and resource r
  T(s) : Resource requirements for service s
  α, β, γ : Weighting factors (α + β + γ = 1, 0 ≤ each ≤ 1)
  θ : Load-balance threshold (0 < θ < 1)
Output:
  X : Service-to-provider allocation matrix
  ρ : Replication factor
  Availability Score, Security Score, Total Cost

1  Function DMCSAO(P, S, R, A, S, C, T, α, β, γ, θ):
2      Initialize X[i][j] = 0  ∀ i ∈ S, j ∈ P
3      ρ ← 1   // initial replication factor
4      while not converged:
5          for each sᵢ ∈ S:
6              P_opt ← {}  // optimal provider set for sᵢ
7              for j = 1 to ρ:
8                  p* ← arg max_{p∈P\P_opt} (α·A(p) + β·S(p) − γ·C(p,T(sᵢ)))
9                  P_opt ← P_opt ∪ {p*};  X[i][p*] ← 1
10         // Compute provider load
11         for each p ∈ P:
12             L[p] ← Σ X[i][p] · T(sᵢ)  for all sᵢ ∈ S
13         L_avg ← mean(L[p])  for all p ∈ P
14         if max(|L[p] − L_avg|) > θ · L_avg:
15             ρ ← ρ + 1   // increase replication
16         Update A(p), S(p), C(p,r) given current allocation
17         if stable or max iterations reached:  converged ← true
18     Availability Score = (1/|S|) · Σ[1 − ∏(1 − X[i][p]·A(p))]
19     Security Score     = (1/|S|) · Σ max(X[i][p]·S(p))
20     Total Cost         = Σ X[i][p]·C(p,T(sᵢ))
21     return X, ρ, Availability Score, Security Score, Total Cost
```

```
Function DynamicReallocation(X, P_fail):
    for each sᵢ ∈ S where X[i][p]=1 for some p ∈ P_fail:
        P_avail ← P \ P_fail
        p' ← arg max_{p∈P_avail} (α·A(p) + β·S(p) − γ·C(p,T(sᵢ)))
        X[i][p'] ← 1;  X[i][p] ← 0  ∀ p ∈ P_fail
    Update A(p), S(p), C(p,r) under new allocation
    return X
```

The time complexity of the core allocation loop is $O(|S| \cdot \rho \cdot |P|)$ per iteration, making the algorithm tractable for typical enterprise-scale deployments where $|S|, |P| \ll 10^3$. The weighting triplet (α, β, γ) provides a principled mechanism for operators to encode organizational priorities: a latency-critical application might assign high α, whereas a cost-constrained batch workload might amplify γ. The DynamicReallocation function is designed to execute within a single decision cycle, ensuring that service remapping introduces negligible additional latency beyond the physical failover propagation time.

D. Evaluation Metrics and Data Collection

Service availability is quantified via continuous synthetic health checks issued every 30 seconds from distributed monitoring probes deployed across all participating regions. Response latency is captured through distributed measurement agents that simulate representative user request profiles and record end-to-end round-trip times. The telemetry collection interval is one second for performance counters and real-time for security event streams; cost data are aggregated at hourly granularity [34][38]. Collected samples undergo automated validation for completeness and statistical outlier detection prior to inclusion in the analysis pipeline. The primary evaluation dimensions are: security effectiveness (threat detection rate, false-positive ratio, incident response latency); service reliability (availability percentage, MTBF, MTTR); performance efficiency (CPU utilization, memory footprint, network latency, throughput); and cost efficiency (resource utilization ratio, operational overhead, ROI on security spend) [35][36][37].

IV. Test Scenarios

A. Network Partition Scenario

Network partition events—caused by ISP outages, misconfigured routing equipment, or targeted denial-of-service attacks—can sever inter-provider communication channels and fragment the logical topology of a multi-cloud deployment [28]. To simulate these conditions, the study employs network Access Control Lists (ACLs) and stateful firewall policies to introduce controlled connectivity disruptions between provider environments. The severity of the partition is escalated incrementally: initial trials inject a packet-loss rate of 10%, subsequent trials increase this to 30% while simultaneously elevating round-trip latency by 20 ms, and terminal trials impose complete path segregation. This graduated approach allows the evaluation framework to capture failure detection sensitivity, progressive performance degradation, and full-partition recovery patterns in a single experimental run [29].

B. Datacenter Failure Scenario

Complete datacenter or availability-zone failures, while statistically infrequent, represent the most operationally severe failure class that a multi-cloud deployment must withstand [30]. The scenario is enacted by systematically halting all virtual machine instances and associated managed services within a targeted availability zone of one participating provider. Key evaluation criteria include: the elapsed time from failure detection to service restoration at an alternative location (RTO); the most recent recoverable data snapshot relative to the failure instant (RPO); the fidelity of application state following geo-failover; and the capacity of the surviving provider infrastructure to sustain the entire combined workload without service degradation [31][32].

V. Results And Analysis

Experiments were executed using Python 3.8 with the NetworkX 2.6.3 graph modelling library on a dedicated high-performance workstation equipped with an Intel Xeon E5-2680 v4 processor (14 cores, 2.4 GHz base frequency), 128 GB DDR4 RAM, and a 2 TB NVMe SSD. Each cloud provider environment was instantiated within an isolated Docker 20.10.8 container, with resource allocations calibrated to mirror representative production cloud configurations [33][34]:

- CSP1: 48 vCPUs, 192 GB RAM, 2 TB storage — configured for high availability
- CSP2: 36 vCPUs, 144 GB RAM, 1.5 TB storage — configured with geo-redundancy
- CSP3: 24 vCPUs, 96 GB RAM, 1.0 TB storage — optimized for rapid failover

The inter-provider network topology was modelled as a fully connected mesh, with inter-provider links rated at 10 Gbps (5 ms baseline latency), intra-zone links at 25 Gbps (2 ms), and cross-zone links at 15 Gbps (3 ms). Security services were instantiated as distributed graph nodes, each carrying defined capacity and latency attributes: access control (50,000 req/s, 0.5 ms), encryption (40,000 req/s, 0.8 ms), monitoring (60,000 req/s, 0.3 ms), and failover (45,000 req/s, 0.6 ms) [35].

A. Network Partition Failure Simulations

Three network partition experiments were conducted at progressively larger scales (30, 50, and 100 nodes), each running for 180 seconds with graduated fault injection.

30-Node Baseline: Nodes were distributed as 8 in CSP1, 7 in CSP2, 7 in CSP3, and 8 on-premises. Under normal operation, mean response latency was 23.5 ms with a packet loss rate of 2.3%. Upon failure induction, the system recovered within an average of 1.8 seconds, with minimal routing convergence overhead attributable to the compact network topology.

50-Node Intermediate Scale: Distributing 15 nodes in CSP1, 12 in CSP2, 12 in CSP3, and 11 on-premises increased baseline latency to 28.7 ms and packet loss to 3.1%. Recovery time rose to 2.4 seconds—a 33% increase relative to the 30-node baseline—reflecting the more complex dependency graph and longer routing reconvergence paths at this scale.

100-Node Large Scale: The largest experiment (30 nodes in CSP1, 25 in CSP2, 25 in CSP3, 20 on-premises) exhibited baseline latency of 35.2 ms and a 4.2% packet loss rate. Failover recovery averaged 3.6 seconds—a 50% increase compared to the 50-node configuration—indicating exponential growth in recovery complexity as network scale doubles. These results identify large-scale deployments as the primary target for future algorithmic optimization, particularly in the domain of route reconvergence and resource reallocation scheduling[37].

B. Multi-Cloud versus Single-Cloud Comparative Analysis

Four performance dimensions were compared across multi-cloud and single-cloud deployments using identical workload profiles:

Response Latency: The multi-cloud architecture maintained a mean response time of 23.5 ms throughout the simulation, compared to 27.8 ms for the single-cloud baseline—a 15.5% improvement. During the peak-stress interval (60–90 seconds), multi-cloud response variance remained within ±15% of baseline, whereas single-cloud variance reached ±35%, demonstrating superior stability under load [Figure 7].

Packet Loss: The multi-cloud setup achieved an average packet delivery loss of 2.3% versus 3.8% for single-cloud—a 39.5% reduction. During simulated partition events (120–150 seconds), peak loss was capped at 4.7% in multi-cloud versus 8.9% in single-cloud, confirming that provider redundancy provides meaningful protection against network-layer faults [Figure 8].

Failover Events: Over the 180-second observation window, the multi-cloud environment generated 12 failover events with a mean recovery time of 1.8 seconds (±0.3 s standard deviation). The single-cloud configuration generated 18 events with a mean recovery time of 2.9 seconds (±0.8 s). The narrower recovery-time distribution in the multi-cloud case reflects the consistency advantage conferred by pre-provisioned backup resources across multiple providers [Figure 9].

Aggregate Performance: Summary statistics across all four metrics—average response time, maximum response time, average packet loss, and total failover count—uniformly favour the multi-cloud configuration, with the most pronounced gains observed in packet reliability and failover latency [Figure 10].

C. Datacenter Failure Simulation with Network Graph Visualization

Datacenter failure experiments were conducted using NetworkX graph visualizations in which each provider zone was rendered with a distinct colour scheme (CSP1: light blue; CSP2: light green; CSP3: light coral). Compute nodes were represented as circles and storage nodes as squares; edge thickness encoded bandwidth capacity while colour intensity reflected real-time utilization. This representation enabled immediate visual identification of failure propagation paths and recovery trajectories.

The simulation progressed through three clearly delineated phases. In the pre-failure state, workloads were distributed evenly, inter-provider communication paths operated at nominal capacity, and resource utilization adhered to baseline patterns. During failure initiation, the target zone underwent systematic node shutdown; edge weights were adjusted in real time to reflect lost capacity, and the graph topology reconfigured to isolate the failed sub-graph. The recovery phase was characterized by dynamic workload redistribution, activation of pre-configured alternative paths, and progressive restoration of service endpoints—all visualized as incremental graph transformations [Figures 11–12].

Throughput and latency measurements under increasing load are presented in Figures 13 and 14. As request rates escalated, individual CSPs exhibited linear latency growth, while the combined multi-cloud tier maintained near-flat latency up to approximately 40,000 requests per second before degrading gracefully. Throughput efficiency for the multi-cloud configuration degraded more slowly than any individual provider, confirming the load-levelling effect of cross-provider distribution.

VI. Discussion

A. Interpretation of Findings

Taken together, the experimental results establish several generalizable principles for multi-cloud resilience engineering. First, response latency scales sub-linearly with network size under the DTOSUME allocation strategy, confirming that effective load distribution mitigates the worst-case latency penalties associated with larger topologies. Second, recovery time scales super-linearly—approaching exponential growth between the 50-node and 100-node configurations—indicating that failover orchestration becomes the dominant bottleneck at scale and that the Dynamic Reallocation function will require further optimization for very large deployments. Third, the 33.3% reduction in failover incident rate demonstrates that proactive workload placement across providers materially reduces the frequency of reactive failover events, not merely their duration.

The comparative advantage of the multi-cloud approach is most pronounced during periods of induced stress—precisely the conditions under which resilience architectures are most consequential. The narrower recovery-time variance observed in the multi-cloud configuration (±0.3 s versus ±0.8 s) suggests that predictability, as well as speed, is a meaningful quality dimension for cloud resilience, with implications for SLA design and incident response planning[38].

B. Implications for Multi-Cloud Strategy and Architecture

The findings carry several concrete implications for practitioners:

1. Resource Distribution: Balanced node allocation across providers is a necessary precondition for DTOSUME to achieve its theoretical availability gains. Unbalanced initial distributions inflate the replication factor ρ unnecessarily, increasing cost. Geographic diversification of provider selection further reduces correlated failure risk.

2. Network Design: Every inter-provider boundary should be supported by at least two physically independent connectivity paths. Bandwidth capacity planning must account for peak-failover traffic, which can transiently double the load on surviving links.

3. Security Policy Harmonization: Security control coverage must be verified as uniform across all provider environments before DMCSAO's security optimization objective can be meaningfully operationalized. Gaps in monitoring or enforcement create blind spots that adversaries can exploit during the confusion of a failover event.

4. Scalability Threshold Awareness: Organizations targeting deployments above 50 nodes should anticipate exponential recovery time growth and pre-provision supplementary reconvergence capacity or adopt hierarchical failover strategies that contain failure blast radius within localized sub-graphs.

VII. Conclusion

This paper has presented the DTOSUME, a principled framework for allocating services across heterogeneous cloud providers while jointly maximizing availability, security posture, and cost efficiency. Through extensive simulation experiments encompassing network partition and data center failure scenarios at three deployment scales, DTOSUME was shown to deliver a 15.5% reduction in mean response latency, a 39.5% improvement in packet delivery reliability, and a 33.3% reduction in failover incidents relative to equivalent single-cloud deployments. The algorithm sustained 99.99% service availability during failure scenarios, compared to 99.95% for the single-cloud baseline, a margin that translates to approximately 36 minutes of additional annual uptime.

Scalability analysis revealed a sub-linear response time growth pattern that validates the effectiveness of the distributed allocation strategy, alongside super-linear recovery time growth that identifies dynamic reallocation at large scale as the primary direction for future algorithmic refinement. Network graph visualizations of data center failures provided interpretable evidence of how failure propagation and recovery trajectories evolve through the system topology, informing both algorithm design and operational response planning[36].

Future research will extend DTOSUME in three directions: (i) incorporation of machine-learning-driven predictive availability functions A(p) and S(p) that anticipate provider degradation before it manifests; (ii) development of hierarchical failover strategies that bound recovery complexity to O(log n) for large-scale deployments; and (iii) formal cost optimization integration that dynamically adjusts γ in response to real-time spot-pricing signals. These extensions promise to further close the gap between theoretical optimality and deployable resilience in production multi-cloud environments.

Acknowledgement

The authors gratefully acknowledge the Department of Artificial Intelligence & Machine Learning and the Department of Computer Science & Business Systems at Dr. Ambedkar Institute of Technology, Bengaluru, for providing the computational resources and research environment that supported this work.

References

[1] R. Buyya et al., "A manifesto for future generation cloud computing: Research directions for the next decade," ACM Computing Surveys, vol. 51, no. 5, pp. 1–38, 2018.

[2] M. Armbrust et al., "A view of cloud computing," Communications of the ACM, vol. 53, no. 4, pp. 50–58, 2010.

[3] P. Mell and T. Grance, "The NIST definition of cloud computing," NIST Special Publication 800-145, 2011.

[4] Gartner, "Gartner forecasts worldwide public cloud end-user spending to grow 18% in 2021," Gartner Press Release, Nov. 2020.

[5] P. Mell and T. Grance, "The NIST definition of cloud computing," National Institute of Standards and Technology, Gaithersburg, MD, SP 800-145, 2011.

[6] A. Benlian, M. Kettinger, A. Sunyaev, and T. Winkler, "The transformative value of cloud computing," Journal of Management Information Systems, vol. 35, no. 3, pp. 719–739, 2018.

[7] Uptime Institute, "Annual Outage Analysis 2021," Uptime Institute, 2021.

[8] ITIC, "2021 Global Server Hardware, Server OS Reliability Survey," Information Technology Intelligence Consulting, 2021.

[9] S. Shahzad, "Protecting the integrity of digital evidence and basic human rights during the process of digital forensics," Ph.D. dissertation, Stockholm Univ., 2015.

[10] N. MacDonald and S. Riley, "Market Guide for Cloud Access Security Brokers," Gartner, 2019.

[11] N. MacDonald, "Market Guide for Cloud Workload Protection Platforms," Gartner, 2019.

[12] N. MacDonald, "Innovation Insight for Cloud Security Posture Management," Gartner, 2019.

[13] A. Barker, B. Varghese, J. S. Ward, and I. Sommerville, "Academic cloud computing research: Five pitfalls and five opportunities," USENIX HotCloud, 2014.

[14] Z. Á. Mann, A. Metzger, J. Prade, and R. Seidl, "Optimized application deployment in the fog," in Service-Oriented Computing, Springer, 2019, pp. 283–298.

[15] M. A. Rodriguez and R. Buyya, "Deadline-based resource provisioning and scheduling algorithm for scientific workflows on clouds," IEEE Trans. Cloud Comput., vol. 2, no. 2, pp. 222–235, 2014.

[16] A. Balalaie, A. Heydarnoori, and P. Jamshidi, "Microservices architecture enables DevOps: Migration to a cloud-native architecture," IEEE Software, vol. 33, no. 3, pp. 42–52, 2016.

[17] R. K. Yin, Case Study Research and Applications: Design and Methods, 6th ed. Sage Publications, 2017.

[18] M. Fowler, "Continuous Integration," martinfowler.com, 2006.

[19] Gartner, "Magic Quadrant for Cloud Infrastructure and Platform Services," Gartner, 2021.

[20] R. Buyya, S. N. Srirama et al., "A manifesto for future generation cloud computing," ACM Computing Surveys, vol. 51, no. 5, pp. 1–38, 2019.

[21] P. Jamshidi, C. Pahl et al., "Microservices: The journey so far and challenges ahead," IEEE Software, vol. 35, no. 3, pp. 24–35, 2018.

[22] K. Bilal, O. Khalid et al., "Fault tolerance in the multi-cloud: Analysis and benchmarking," IEEE Trans. Cloud Comput., vol. 8, no. 4, pp. 1105–1118, 2020.

[23] S. Singh, Y.-S. Jeong, and J. H. Park, "A deep learning-based IoT-oriented infrastructure for secure smart cities," J. Inf. Security Appl., vol. 48, 102351, 2019.

[24] N. Fallenbeck and C. Eckert, "IT security risk management: Perceived IT security risks in cloud computing," Int. J. Inf. Security, vol. 14, no. 6, pp. 585–600, 2015.

[25] M. Ali, S. U. Khan, and A. V. Vasilakos, "Security in cloud computing: Opportunities and challenges," Information Sciences, vol. 305, pp. 357–383, 2015.

[26] D. Catteddu and G. Hogben, "Cloud computing: Benefits, risks and recommendations for information security," ENISA Report, 2009.

[27] Z. Xu, W. Liang et al., "Efficient algorithms for capacity-constrained cloud resource allocation optimization," Int. J. Parallel Programming, vol. 44, no. 5, pp. 1083–1107, 2016.

[28] P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: Measurement, analysis, and implications," ACM SIGCOMM, 2011, pp. 350–361.

[29] A. Bessani et al., "On the efficiency of durable state machine replication," USENIX ATC, 2013, pp. 169–180.

[30] D. Ford et al., "Availability in globally distributed storage systems," USENIX OSDI, 2010, pp. 61–74.

[31] K. Keeton et al., "Designing for disasters," USENIX FAST, 2004, pp. 59–62.

[32] S. Newman, Building Microservices: Designing Fine-Grained Systems. O'Reilly Media, 2015.

[33] P. Bermbach, E. Wittern, and S. Tai, Cloud Service Benchmarking. Springer, 2017.

[34] C. Liu, P. Lai, and J. Wu, "System performance evaluation of cloud computing services: A network perspective," IEEE Trans. Cloud Comput., vol. 9, no. 3, pp. 1158–1171, 2021.

[35] Z. Li, L. O'Brien, H. Zhang, and R. Cai, "On a catalogue of metrics for evaluating commercial cloud services," ACM Computing Surveys, vol. 51, no. 6, pp. 1–28, 2019.

[36] Kushala M V and Dr. B S Shylaja, "Recent trends on security issues in multi-cloud computing: A survey," IEEE ICOSEC, 2020. DOI: 10.1109/ICOSEC49089.2020.9215303.

[37] R. Bhaskar and Dr. B S Shylaja, "Dynamic virtual machine provisioning in cloud computing using knowledge-based reduction method," AISC, vol. 1162, pp. 193–202.

[38] S. R. Deepu and Dr. B S Shylaja, "Performance comparison of deduplication techniques for storage in cloud computing environment," Asian Journal of Computer Science and Information, 2014.

[39] Kushala M V, Dr. B S Shylaja, "Providing More Security to Data Stored in Multi-Cloud Environment Using Encryption and Decryption Techniques", IJSREM, ISSN: 2582-3930, DOI: 10.55041/IJSREM49784.