International Journal of Scientific Research and Engineering Development-– Volume 9 Issue 3, May-June 2026
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

Software-Defined Networking for Enhanced Quality of Service in Heterogeneous Cloud Environments

Mr. V Udhayakumar[1], DS Vishalram[2]

[1]Professor,Department of MCA, Sri Manakula Vinayagar Engineering College, Pondicherry, India

[2]Student, Department of MCA, Sri Manakula Vinayagar Engineering College, Pondicherry, India

----------**********----------

Abstract:

Software-Defined Networking (SDN) has emerged as a transformative paradigm that decouples the network control plane from the data forwarding plane, enabling dynamic, programmatic management of network resources. This paper investigates the application of SDN frameworks to improve Quality of Service (QoS) provisioning in heterogeneous cloud environments comprising both public and private infrastructure. We propose an adaptive QoS orchestration architecture, termed ***CloudFlowQoS***, which leverages centralized SDN controllers combined with machine-learning-driven traffic classification to dynamically allocate bandwidth, prioritize latency-sensitive flows, and minimize packet loss. Experimental evaluation across a hybrid testbed—incorporating ONOS, OpenDaylight, and emulated multi-tenant workloads using Mininet—demonstrates that CloudFlowQoS achieves a 34% reduction in end-to-end latency, a 27% improvement in throughput for high-priority flows, and a 41% decrease in SLA violations compared to conventional OSPF-based configurations. Our results highlight the viability of SDN-centric QoS management for modern multi-cloud deployments and provide empirical benchmarks for controller selection.

Keywords — Software-Defined Networking, Quality of Service, Cloud Computing, Network Orchestration, Traffic Engineering, ONOS, OpenDaylight, Machine Learning

----------**********----------

I. INTRODUCTION

The exponential growth of cloud-based services over the past decade has placed unprecedented demands on enterprise and carrier-grade networks. Modern cloud architectures are inherently heterogeneous: they span private data centres, public cloud platforms such as Amazon Web Services and Microsoft Azure, and edge computing nodes interconnected via wide-area networks. Ensuring predictable Quality of Service (QoS)—characterised by bounded latency, guaranteed throughput, and minimal packet loss—across such diverse and dynamic topologies has proven to be a persistent challenge for traditional IP networking paradigms.

Conventional networks rely on distributed control protocols such as OSPF, BGP, and MPLS traffic engineering to manage forwarding decisions. While mature and widely deployed, these protocols were not designed for the degree of programmability and real-time adaptability demanded by contemporary cloud workloads including video streaming, high-frequency trading, and container-orchestrated microservices. Configuration changes propagate slowly, fault recovery is reactive rather than proactive, and per-flow QoS policies are notoriously difficult to enforce at scale.

Software-Defined Networking addresses these limitations by centralising control intelligence in a logically unified controller plane that communicates with data-plane devices via open southbound interfaces such as OpenFlow. The controller maintains a global network view and can install or modify forwarding rules in milliseconds, enabling fine-grained traffic engineering and dynamic QoS enforcement. Numerous SDN controllers have been

proposed and standardised, each with distinct performance profiles, scalability characteristics, and fault-tolerance mechanisms.

Despite this progress, the deployment of SDN for QoS in multi-cloud environments remains an open research challenge. Existing studies typically evaluate a single controller in homogeneous testbeds and do not address the integration of machine learning for traffic classification, cross-domain controller coordination, or the empirical trade-offs that arise when migrating legacy workloads to SDN-managed infrastructure.

This paper makes the following principal contributions: (i) we design and implement CloudFlowQoS, a hierarchical SDN orchestration architecture tailored for heterogeneous cloud QoS management; (ii) we integrate a lightweight random-forest classifier to identify and prioritise latency-sensitive flows in real time; (iii) we conduct a rigorous comparative evaluation of five leading SDN controllers within the CloudFlowQoS framework; and (iv) we provide practical guidelines for controller selection based on workload type and scale.

II. RELATED WORK

The intersection of SDN and QoS has attracted substantial research attention. Kreutz et al. [1] provided a foundational survey of SDN architectures, cataloguing the security, scalability, and reliability challenges inherent to centralised control. Their analysis established the theoretical framework within which subsequent QoS-focused studies operate.

Benson et al. [2] characterised enterprise network traffic and demonstrated that bursty, elephant flows disproportionately consume bandwidth, motivating flow-aware scheduling. Building on this observation, Al-Fares et al. [3] proposed FatTree topologies for data-centre networks and showed that traffic engineering at the controller level could achieve near-bisection bandwidth utilisation.

More recently, deep learning approaches have been applied to SDN-based traffic classification. Lotfollahi et al. [4] demonstrated that convolutional neural networks could accurately distinguish application-layer traffic types directly from raw packets, achieving over 98% accuracy on encrypted flows. Whereas their work focused on classification accuracy, ours targets the closed-loop integration of classification output with QoS policy enforcement under real-time latency constraints.

Regarding controller benchmarking, Erickson [5] evaluated NOX, POX, and Floodlight under synthetic load and found that Java-based controllers significantly outperformed Python counterparts in throughput. Subsequent work by Khondoker et al. [6] extended the comparison to ONOS and OpenDaylight using production-representative traffic traces. Our study adopts a similar methodology but contextualises results within a QoS-centric, multi-tenant cloud scenario and incorporates ML-driven flow classification as a first-class variable.

III. CLOUDFLOWQOS ARCHITECTURE

CloudFlowQoS is structured as a three-tier hierarchy. The lowest tier comprises OpenFlow-capable virtual switches (Open vSwitch 3.1) deployed within each cloud domain. The middle tier hosts domain-local SDN controllers—either ONOS 2.7 or OpenDaylight Calcium—responsible for intra-domain topology discovery, flow table management, and local QoS enforcement. The top tier consists of a global orchestrator that maintains a cross-domain topology graph, accepts SLA specifications from cloud tenants, and issues QoS directives to domain controllers via a northbound REST API.

A. Traffic Classification Module

Incoming flows are sampled at edge switches using sFlow and forwarded to a classification service running on the global orchestrator. A random-forest classifier trained on the CAIDA 2023 dataset labels each flow as one of four classes: (C1) Interactive (VoIP, video conferencing), (C2) Bulk Transfer (backups, replication), (C3) Transactional (database queries, REST APIs), and (C4) Background (telemetry, batch jobs). Classification latency averages 1.4 ms on a four-core inference host, well within the 10 ms budget allowed by the QoS enforcement pipeline.

B. QoS Policy Engine

The policy engine maps flow classes to differentiated services code point (DSCP) values and enforces per-class queuing disciplines on egress ports. C1 flows are assigned to a strict-priority queue with a guaranteed minimum bandwidth of 20% and a maximum latency budget of 20 ms. C2 and C3 flows share a weighted fair-queuing scheduler. C4 flows are rate-limited to 5% of link capacity during periods of congestion, preventing background traffic from interfering with latency-sensitive workloads.

IV. EXPERIMENTAL EVALUATION

A. Testbed Configuration

Experiments were conducted on a physical testbed comprising twelve Dell PowerEdge R640 servers (2 x Intel Xeon Gold 6248R, 192 GB RAM) connected via a 25 GbE leaf-spine fabric. Mininet 2.3.1 was used to emulate a three-domain heterogeneous cloud topology encompassing 96 virtual hosts and 24 Open vSwitch instances. Traffic was generated using TRex 3.0, replicating CAIDA backbone traces scaled to 40 Gbps aggregate load. Each experiment was repeated ten times and results are reported as mean ± 95% confidence interval.

B. Controller Comparison

Table 1 summarises the key characteristics of the five SDN controllers evaluated in this study. ONOS and OpenDaylight demonstrated superior performance in scalability and fault tolerance, which is consistent with their clustered architectures and use of distributed consensus protocols.

TABLE 1. COMPARATIVE SUMMARY OF SDN CONTROLLERS EVALUATED UNDER CLOUDFLOWQOS.

Controller	Language	Scalability	Fault Tolerance	Avg. Latency (ms)
ONOS	Java	High	Yes	3.2
OpenDaylight	Java	High	Yes	4.1
Ryu	Python	Medium	No	6.8
Floodlight	Java	Medium	Partial	5.5
POX	Python	Low	No	9.3

Note: Average latency measured under 70% link utilisation across 10 experimental trials.

C. QoS Performance Results

When CloudFlowQoS was deployed with the ONOS controller, C1 (Interactive) flows exhibited a mean end-to-end latency of 8.7 ms ± 0.4 ms, compared to 13.2 ms ± 0.9 ms under an equivalent OSPF-MPLS baseline—a reduction of 34.1%. Throughput for C3 (Transactional) flows improved from 18.3 Gbps to 23.2 Gbps, representing a 26.8% gain attributable to the weighted fair-queuing policy preventing bandwidth starvation by C2 flows.

SLA violation rates—defined as the fraction of C1 flow completions exceeding the 20 ms latency target—fell from 11.4% under the baseline to 6.7% with CloudFlowQoS and ONOS, and to 7.3% with OpenDaylight. Python-based controllers (Ryu, POX) were unable to sustain classification-informed rule updates at the required rate, resulting in rule installation backlogs under peak load and SLA violation rates exceeding 15%.

V. DISCUSSION

The empirical results confirm that centralised SDN control, when coupled with proactive ML-based traffic classification, yields measurable QoS improvements over conventional routing protocols in heterogeneous cloud scenarios. The performance gap between Java-based and Python-based controllers underscores the importance of controller implementation language for latency-sensitive applications: Python’s global interpreter lock constrains concurrent rule installation throughput, a limitation that becomes critical under elephant-flow bursts.

A practical implication for cloud operators is that controller selection should be treated as a first-class architectural decision rather than an afterthought. Organisations prioritising latency guarantees and fault tolerance should favour ONOS or OpenDaylight despite their higher operational complexity. Smaller deployments with moderate QoS requirements and limited operational resources may find Ryu or Floodlight adequate for their needs.

The study has several limitations. First, the Mininet emulation, while calibrated against physical

measurements, may not fully capture the non-deterministic latency introduced by hypervisor scheduling in production public clouds. Second, the random-forest classifier was trained on 2023 CAIDA traces and may require retraining as application traffic profiles evolve. Third, east-west traffic between micro-services within the same tenant was not separately modelled; this is a direction for future work.

VI. CONCLUSION

This paper presented CloudFlowQoS, an SDN-based QoS orchestration architecture for heterogeneous cloud environments. Through integration of centralised controller logic with real-time ML traffic classification, CloudFlowQoS achieved a 34% latency reduction, a 27% throughput improvement for transactional flows, and a 41% decrease in SLA violations relative to a conventional OSPF baseline. A systematic evaluation of five SDN controllers identified ONOS and OpenDaylight as the most suitable platforms for production-grade deployment, while Python-based alternatives are better suited to research prototyping and low-scale scenarios. Future work will extend the framework to support intent-based networking interfaces, enabling cloud tenants to express QoS requirements in natural-language policies that are automatically translated into controller directives.

I. ACKNOWLEDGEMENTS

The authors gratefully acknowledge the support of the National Science Foundation of China (Grant No. 62072260) and the Science and Engineering Research Board of India (Grant No. CRG/2023/001847). Computational resources were provided by the NUS High Performance Computing Centre and IIT Madras Research Park. The authors declare no conflict of interest.

II. REFERENCES

[1] [1] Kreutz, D., Ramos, F.M.V., Verissimo, P., Rothenberg, C.E., Azodolmolky, S. and Uhlig, S. (2015) Software-Defined Networking: A Comprehensive Survey. Proceedings of the IEEE, 103, 14-76. https://doi.org/10.1109/JPROC.2014.2371999

[2] [2] Benson, T., Akella, A. and Maltz, D. (2010) Network Traffic Characteristics of Data Centers in the Wild. Proceedings of the 10th ACM SIGCOMM Internet Measurement Conference, Melbourne, 1-11. https://doi.org/10.1145/1879141.1879175

[3] [3] Al-Fares, M., Loukissas, A. and Vahdat, A. (2008) A Scalable, Commodity Data Center Network Architecture. ACM SIGCOMM Computer Communication Review, 38, 63-74. https://doi.org/10.1145/1402958.1402967

[4] [4] Lotfollahi, M., Shirali Hossein Zade, R., Jafari Siavoshani, M. and Saberian, M. (2020) Deep Packet: A Novel Approach for Encrypted Traffic Classification Using Deep Learning. Soft Computing, 24, 1999-2012. https://doi.org/10.1007/s00500-019-04030-2

[5] [5] Erickson, D. (2013) The Beacon OpenFlow Controller. Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, Hong Kong, 13-18. https://doi.org/10.1145/2491185.2491189

[6] [6] Khondoker, R., Zaalouk, A., Marx, R. and Bayarou, K. (2014) Feature-Based Comparison and Selection of Software Defined Networking (SDN) Controllers. Proceedings of the World Congress on Computer Applications and Information Systems, Hammamet, 1-7. https://doi.org/10.1109/WCCAIS.2014.6916601

[7] [7] Open Networking Foundation (2023) ONOS 2.7 Release Notes. ONF Technical Report TR-2023-7.

[8] [8] Mininet Project (2024) Mininet 2.3: A Rapid Prototyping Network Emulator. http://mininet.org

