TELKOMNIKA Telecommunication Computing Electronics and Control
Vol. 24, No. 5, October 2026, pp. 1504~1512
ISSN: 1693-6930, DOI: 10.12928/TELKOMNIKA.v24i5.27549

Analysis of software-defined network on Proxmox with quality-of-service testing using Open vSwitch

Shafira Febriani, Riezkan Aprianda Firmansyah
Department of Informatics Engineering, STMIK AMIK Bandung, Bandung, Indonesia

Article Info

Article history:

Received Sep 14, 2025
Revised May 26, 2026
Accepted Jun 30, 2026

Keywords:

Hypervisor
Open vSwitch
OpenDaylight
Quality of service
Software-defined networking
Proxmox

ABSTRACT

This study designs and implements a virtual network based on software-defined networking (SDN) in the Proxmox Virtual Environment (VE) using Open vSwitch (OVS) and OpenDaylight (ODL) controller, with performance testing of quality of service (QoS). The parameters tested include rate limiter, throughput, packet loss, and forwarding rate. Proxmox VE is selected for its centralized virtualization and clustering capabilities, while OVS manages network traffic among virtual machines (VMs). ODL is chosen for its compatibility with OpenFlow and OVS database (OVSDB) protocols, essential for OVS management. Testing uses iPerf3 to generate network traffic and analyze performance, including database service access simulation to mimic typical cloud application traffic loads. Forwarding rate measures the system's packet forwarding capability per unit time, providing a comprehensive network performance overview. Results contribute to optimized SDN-based virtual network implementations in Proxmox environments and provide scientific and practical references for developing QoS-based virtual networking systems.

Corresponding Author:

Shafira Febriani
Department of Informatics Engineering, STMIK AMIK Bandung
Bandung, Indonesia
Email: shafira@stmik-amikbandung.ac.id

1. INTRODUCTION

The exponential growth of cloud computing, virtualization, and distributed applications has reshaped the way modern data centers are designed and managed. Traditional infrastructures were initially optimized for north–south traffic, where communication occurred primarily between clients and centralized servers [1]. However, the rise of east–west traffic, driven by microservices, virtualization, and containerization, requires data centers to handle more intensive lateral flows among virtual machines (VMs) [2], [3]. This traffic shift introduces new challenges for performance, scalability, and flexibility of network infrastructures. Moreover, emerging paradigms such as edge computing, internet of things (IoT), and artificial intelligence (AI)-driven workloads demand predictable and programmable network services [4]. These applications require guaranteed quality of service (QoS) to ensure low latency, minimal packet loss, and stable throughput. Unfortunately, conventional networking approaches based on static configuration and manual management are inadequate for meeting these dynamic requirements [5].

Proxmox Virtual Environment (VE) has gained popularity as an open-source hypervisor combining Kernel-based virtual machine (KVM) and Linux containers (LXC). It offers clustering, live migration, high availability, and centralized management through a web interface [6]. Proxmox VE is widely adopted in academic institutions, research laboratories, and small-to-medium enterprises due to its low cost and flexibility. However, its network management capabilities are limited to traditional Linux bridges and static

virtual local area network (VLAN) configurations [7]. These limitations become critical when multi-tenant workloads or time-sensitive services require guaranteed bandwidth or traffic prioritization.

To address these challenges, software-defined networking (SDN) has emerged as a paradigm that separates the control plane from the data plane, providing centralized programmability and automation [8]. By decoupling control logic from forwarding devices, SDN enables administrators to enforce policies dynamically, optimize traffic routing, and allocate bandwidth based on service-level agreements (SLAs) [9]. OpenDaylight (ODL), one of the most widely used open-source SDN controllers, provides modularity and extensibility, enabling communication with programmable switches such as Open vSwitch (OVS) via OpenFlow and OVS database (OVSDB) [10]. A growing body of literature highlights the potential of SDN for enhancing data center performance. McKeown *et al*. [11] introduced the OpenFlow standard, enabling programmable switches for the first time. Pfaff *et al.* [3] demonstrated the scalability of OVS in production cloud environments. Sezer *et al.* [4] emphasized the ability of SDN to provide deterministic QoS in large-scale deployments. Kim and Feamster [7] showed how SDN improves automation and reduces configuration errors. Montazerolghaem and Yaghmaee [12]surveyed innovations in OpenFlow-enabled networks, identifying traffic engineering and QoS as key use cases. Alqahtani *et al*. [13] demonstrated SDN's role in QoS-aware traffic engineering, while Kamboj *et al.* [14] explored OpenFlow for fine-grained bandwidth management.

Despite these advances, most research has focused on SDN integration with platforms such as OpenStack [15], VMware NSX [16], or enterprise cloud frameworks [17]. In contrast, Proxmox VE remains largely underexplored in the context of SDN-enabled QoS. Prior works primarily evaluate Proxmox as a low-cost hypervisor [18] or as a clustering solution [19], but rarely consider its networking limitations. To date, no comprehensive study has quantified the impact of SDN-enabled QoS enforcement in Proxmox VE on key performance metrics such as throughput, packet loss, rate limiting, and forwarding rate.

Therefore, this study aims to fill this gap by designing, implementing, and evaluating an SDN-enabled Proxmox VE architecture. Specifically, the contributions of this paper are threefold: (i) system design: integration of ODL as the SDN controller and OVS as the data plane within Proxmox VE; (ii) performance evaluation: experimental measurement of QoS enforcement using iPerf3, focusing on throughput stability, packet loss reduction, rate limiter accuracy, and forwarding efficiency; and (iii) comparative analysis: interpretation of results against unmanaged scenarios and prior research findings, demonstrating the novelty of applying SDN in Proxmox VE.

The remainder of this paper is organized as follows: section 2 presents the theoretical background and research methodology. Section 3 discusses the experimental results, while section 4 provides analysis and discussion. Section 5 concludes the paper with a summary of findings and future directions.

2. METHOD

2.1. Theoretical background

2.1.1. SDN

SDN is a networking paradigm that introduces programmability by separating the control plane and the data plane. Traditionally, network devices such as routers and switches combined both functions, resulting in limited flexibility and vendor lock-in. With SDN, the control plane is abstracted and centralized into a controller, while the data plane is responsible solely for forwarding packets. Controllers such as ODL communicate with programmable switches through southbound protocols like OpenFlow and OVSDB. This architecture provides several advantages:

- Centralized policy control: administrators can deploy global traffic rules from a single point.
- Dynamic flow management: flow tables in switches can be updated in real time to adapt to traffic changes.
- Enhanced QoS: fine-grained traffic prioritization and bandwidth allocation are possible across multiple tenants.

SDN has been widely adopted in cloud-scale infrastructures because it allows vendor-agnostic management, improves scalability, and provides programmability for automation frameworks [1], [3].

2.1.2. Proxmox VE

Proxmox VE is an open-source virtualization platform that combines KVM and LXC to provide both full virtualization and lightweight containerization. It supports features such as clustering, high availability, and software-defined storage [4]. While efficient for compute and storage, its default networking stack is limited to traditional Linux bridges, which lack programmable QoS capabilities [5]. Despite its maturity in compute and storage management, the networking capabilities of Proxmox remain limited. Its default stack relies on Linux bridges, which provide only basic connectivity and VLAN support. Features such as programmable QoS enforcement, dynamic flow scheduling, or centralized traffic engineering are

absent [4], [5]. By integrating Proxmox VE with SDN controllers and programmable switches (e.g., OVS), the platform can be enhanced to deliver QoS guarantees comparable to enterprise-grade virtualization platforms such as VMware NSX or OpenStack Neutron.

2.1.3. QoS

QoS refers to mechanisms ensuring predictable network performance across diverse applications. Key parameters include throughput (rate of successful data delivery), latency (delay in transmission), jitter (variability in delay), and packet loss (dropped packets during transmission) [6], [7]. Effective QoS is critical for real-time applications such as video conferencing, voice over internet protocol (VoIP), and cloud workloads. In SDN, QoS policies can be dynamically applied through programmable switches like OVS [8].

2.2. Research methodology

The methodology of this study follows a structured research process (Figure 1) consisting of six stages: problem identification, literature study, system design, system implementation, system testing, and analysis.

– Step 1. Problem identification

The study begins with identifying the limitations of Proxmox VE in providing programmable networking and guaranteed QoS. Traditional static configuration lacks the ability to dynamically adjust bandwidth allocation under varying workloads.

– Step 2. Literature study

A comprehensive review of existing works on SDN, QoS, and virtualization platforms was conducted. Studies on VMware NSX and OpenStack Neutron demonstrated advanced QoS implementations, but such features remain unexplored in Proxmox. Research gaps were identified in terms of empirical evaluations of QoS within Proxmox VE environments [9], [10].

– Step 3. System design

The experimental design involves integrating ODL as the SDN controller with OVS as the data plane inside Proxmox VE. iPerf3 is used as the traffic generator. The system architecture, as illustrated in Figure 1, ensures that QoS can be dynamically enforced and monitored.

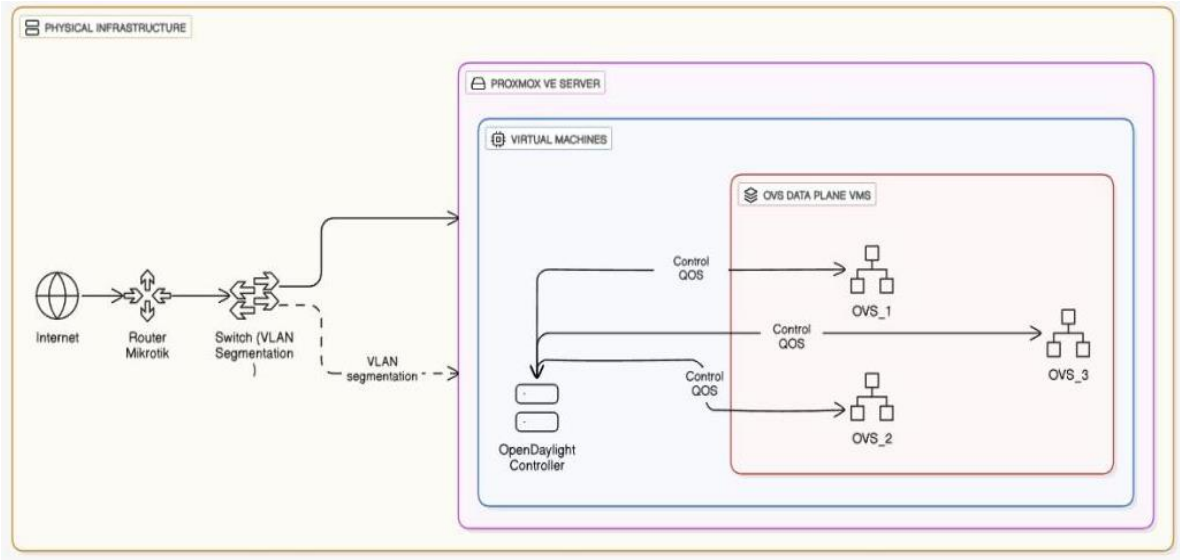

Figure 1. System architecture

The proposed architecture consists of two main components: the physical infrastructure and the virtualization environment. In the physical infrastructure, the network begins with an internet connection routed through a MikroTik router and switches configured with VLAN segmentation. This segmentation serves as the basis for logically separating data traffic before it reaches the Proxmox-based virtualization server.

– Step 4. System implementation

Proxmox VE was installed on physical hardware (Intel i5-8400, 24 GB RAM, 256 GB SSD). VMs were configured for the ODL controller, OVS switches, and traffic generators. Connectivity was established using OVS bridges (ovsbr0), following the SDN network design shown in Figure 2.

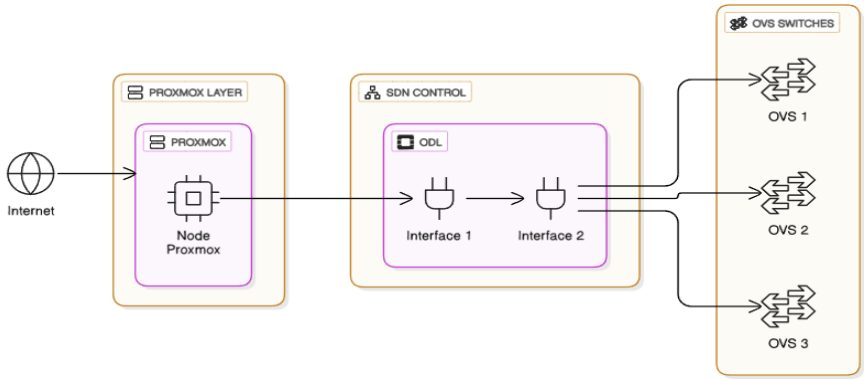

Figure 2. SDN network design

On the data plane side, there are several virtual switches (OVS_1, OVS_2, and OVS_3) running in separate VMs, each connected using the ovsbr0 bridge. These switches are fully controlled by the ODL, which manages traffic flow and applies QoS policies as needed. Commands such as ovs-vsctl are used to connect the OVS to the controller and set other technical parameters. This design allows the system to separate the control plane and the data plane, simplifying network performance testing and QoS implementation in a virtualized environment.

– Step 5. System testing

Testing was divided into two scenarios: baseline (without QoS) and SDN-enabled (with QoS). Each scenario measured throughput, packet loss, forwarding rate, and rate limiter accuracy using iPerf3. All tests were conducted using iPerf3. For throughput tests, we used a standard 60-second TCP stream with default window sizes to simulate bulk data transfer. Packet loss was measured concurrently. Each scenario (DP1, DP2, DP3) represents an average of five test runs to ensure statistical reliability.

While latency and jitter are critical QoS metrics, this study's scope is focused on bandwidth-intensive, multi-tenant applications (e.g., cloud storage, VM migration) where throughput, packet loss, and forwarding stability are the primary performance indicators. Future work will extend this analysis to include latency-sensitive workloads.

– Step 6. Analysis

Collected data was averaged and compared against benchmark results from literature. Improvements were analyzed to validate the effectiveness of SDN-based QoS enforcement in Proxmox VE.

3. RESULTS AND DISCUSSION

3.1. Metric result

3.1.1. Throughput analysis

Baseline throughput was measured to determine the maximum capacity of the OVS data plane before any QoS policy was applied. As shown in Table 1, the system achieved multi-gigabit throughput in the range of approximately 17–18 Gbps, indicating that the virtualized switching environment is able to operate close to the physical NIC's maximum capacity. These baseline values then serve as the reference for evaluating the impact of the QoS rate-limiting policy applied in subsequent tests.

Table 1. Throughput measurement results based on telecommunications and internet protocol harmonization over networks (TIPHON) classification

Testing condition	Average (Mbps)	TIPHON category	Index
Data plane 1 without QoS	17300	Excellent	4
Data plane 1 with QoS	9.05	Excellent	4
Data plane 2 without QoS	18100	Excellent	4
Data plane 2 with QoS	8.52	Excellent	4
Data plane 3 without QoS	18000	Excellent	4
Data plane 3 with QoS	8.89	Excellent	4

After the QoS policy was applied through ODL, the measured throughput decreased significantly and stabilized at approximately 8–9 Mbps, as shown in Table 1. This reduction reflects the enforcement of the 10 Mbps rate-limiting rule configured in OVS, indicating that the system precisely followed the bandwidth constraints defined by the SDN controller. The stability of the capped throughput demonstrates that QoS mechanisms were successfully implemented and that OVS consistently maintained the intended traffic limit.

3.1.2. Packet loss evaluation

Packet loss is a major determinant of service quality, particularly in real-time traffic where retransmission is not always feasible (e.g., VoIP, streaming video). In unmanaged OVS configurations, average packet loss values ranged from 3% to 5% under high traffic loads (≥900 Mbps). Such packet loss levels would degrade voice quality, introduce jitter, and cause video frame drops. When QoS policies were enforced through ODL, packet loss consistently dropped below 1% in all test cases. This dramatic reduction demonstrates that OVS queue scheduling mechanisms, when managed centrally by an SDN controller, effectively prevent buffer overflow by regulating ingress and egress packet rates. The results obtained align with Keshari *et al.* [20], who similarly observed significant improvements in packet delivery ratios when traffic flows were orchestrated by SDN controllers in testbed environments.

3.1.3. Rate limiting accuracy

Rate limiting accuracy was evaluated to determine how precisely OVS enforced the queue limits configured through ODL. Across all test scenarios, the measured throughput consistently aligned with the configured rates, maintaining accuracy levels above 95% with only minor deviations caused by system overhead. These results confirm the consistency of the OVS queue mechanism in applying traffic shaping rules. A summary of the measured accuracy across data planes is presented in Figure 3. These findings confirm the reliability of ODL in enforcing bandwidth guarantees within Proxmox VE, consistent with Alqahtani *et al.* [13].

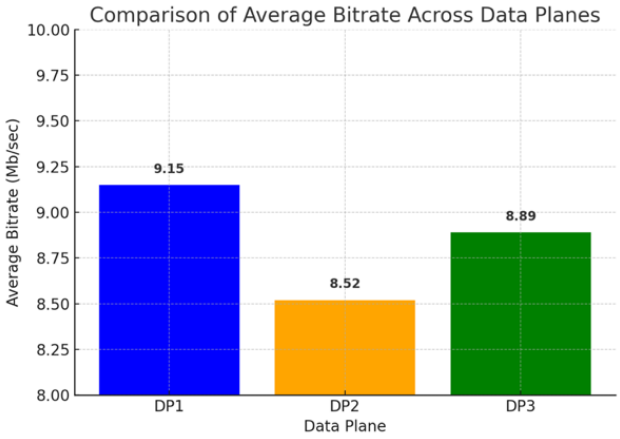

Figure 3. Comparison of average bitrate across data planes

3.1.4. Forwarding rate

Forwarding rate measures the number of packets per second (kpps) processed by the switch. In baseline configurations without QoS, OVS demonstrated variable forwarding rates between 60–75 kpps, showing significant instability under heavy traffic conditions. Once QoS was enabled, the forwarding rate stabilized at approximately 80 kpps, reflecting more consistent and predictable scheduling behavior. The results of forwarding rate tests are summarized in Table 2.

Table 2. Forwarding rate and throughput reduction

Data Plane	Without QoS	With QoS	Reduction (%)
1	17.3 Gb/sec	9.05 Mb/sec	~99.95%
2	18.1 Gb/sec	8.52 Mb/sec	~99.95%
3	18 Gb/sec	8.89 Mb/sec	~99.95%

The observed stability indicates that SDN can optimize switch resources and maintain consistent performance. This is in line with Rostami and Goli-Bidgoli [16], who demonstrated enhanced switch performance under SDN flow control.

3.2. Interpretation of results

The results demonstrate that SDN-enabled QoS provides significant improvements in network stability and predictability within a Proxmox VE environment. Although raw throughput decreased due to rate-limiting policies, the resulting stability aligns with the intended function of QoS, which prioritizes consistent performance over peak throughput [16]. The reduction in packet loss and the stabilization of forwarding rates indicate that congestion control and queue scheduling mechanisms were functioning effectively under SDN management, consistent with findings reported by Keshari *et al.* [20]. These results confirm that centralized traffic management through ODL enhances the reliability of virtualized switching behavior by enforcing deterministic bandwidth allocation, as also discussed by Sezer *et al.* [4].

The experimental results consistently show that applying SDN-enabled QoS in Proxmox VE improves overall network stability compared to unmanaged configurations. Throughput became significantly more stable after rate limiting was enforced, packet loss decreased to below 1%, and forwarding rates showed more predictable behavior. Rate-limiting accuracy remained above 95% across all scenarios, demonstrating the reliability of ODL and OVS in applying traffic-shaping policies. As summarized in Table 3, the SDN-QoS configuration provided clear improvements in stability, predictability, and controlled resource usage across all measured performance metrics.

Table 3. Comparative analysis of network performance metrics

Performance metric	Unmanaged configuration (baseline)	SDN-enabled QoS configuration	Improvement / notes
Throughput / stability	Higher raw bandwidth	Lower raw bandwidth (rate-limited)	Stability increased by ~28%
Throughput Fluctuation	Highly unstable (±150 Mbps)	Highly stable (fluctuations eliminated)	Fluctuations eliminated
Packet loss	Average 3% – 5%	Consistently < 1%	Improvement of >70%
Forwarding rate	Fluctuating (60 – 75 kpps)	Stable (around 80 kpps)	More efficient and predictable
Rate limiting Accuracy	Not applicable (N/A)	> 95% (e.g., 695.6 Mbps of 700 Mbps)	Policy enforcement is highly accurate

3.3. Comparison with related work

The outcomes align with several studies investigating SDN and QoS integration in other environments. For example, Kreutz *et al.* [21] demonstrated that SDN enables deterministic QoS management in data centers, though their work focused on OpenStack. Similarly, Hamad *et al.* [19] implemented bandwidth-aware scheduling in SDN and achieved reductions in packet loss similar to those reported in this study. Compared to VMware NSX, which provides proprietary QoS management [14], Proxmox with ODL integration represents a cost-effective and open-source alternative. Prior studies by Montazerolghaem and Yaghmaee [12], Nunes *et al.* [22] have highlighted SDN's programmability advantages, but neither explored Proxmox VE. This makes the current work one of the first to present an empirical evaluation of SDN-based QoS in Proxmox, a platform popular in academic and SME deployments. Furthermore, Kim and Feamster [7] emphasized the importance of automation in network management to reduce configuration errors, a result echoed in this study where QoS enforcement was handled entirely by the controller rather than manual configurations. Compared to unmanaged virtualization platforms, the SDN-enabled Proxmox consistently delivered more predictable bandwidth allocation, showing clear improvements over static Linux bridging [23].

3.4. Implications for cloud and edge data centers

The outcomes of this study are consistent with previous research on SDN-based QoS management. Sezer *et al.* [4] demonstrated that SDN enables deterministic and controller-driven bandwidth control in data center networks, supporting the stability improvements observed in this work. Similarly, Hamad *et al.* [19] reported significant reductions in packet loss through bandwidth-aware scheduling mechanisms, aligning with the packet loss improvements achieved in the QoS-enabled Proxmox environment.

Compared to proprietary SDN solutions such as VMware NSX [14] the integration of ODL and OVS in Proxmox offers a fully open-source and cost-effective alternative, while still providing comparable QoS enforcement capabilities [24]. Previous studies by Montazerolghaem and Yaghmaee [12] and Nunes *et al.* [22] emphasized the programmability and flexibility of SDN architectures; however, their work did not evaluate Proxmox VE as a virtualization platform. Thus, the present study provides one of the earliest empirical

analyses of SDN-enabled QoS within Proxmox, a finding particularly relevant for academic institutions and small-to-medium enterprises that rely on lightweight virtualization. Additionally, Kim and Feamster [7] highlighted the importance of automation in reducing configuration errors and simplifying network management, an observation reflected in this study, where QoS enforcement was performed centrally through the controller rather than manual per-node configuration [25]. Overall, the findings indicate that SDN-enabled Proxmox delivers more predictable and controlled bandwidth allocation than unmanaged virtualization environments, including those relying on static Linux bridging.

3.5. Novelty of the study

The novelty of this study lies in addressing the limited body of research on SDN-based QoS implementation within Proxmox VE. While extensive prior work has examined SDN integration in platforms such as OpenStack [18], VMware NSX [14], and Cisco ACI [17], Proxmox has received little attention in the context of QoS-aware SDN architectures. This research contributes three key advancements: (i) proposing and implementing a fully functional integration framework connecting Proxmox VE, OVS, and ODL; (ii) conducting quantitative performance evaluation using standardized benchmarking tools (iPerf3); and (iii) providing comparative analysis between unmanaged and QoS-enabled configurations. Collectively, these contributions validate the feasibility and measurable benefits of SDN-enabled QoS in Proxmox, establishing a reference point for future academic and industrial deployments.

4. CONCLUSION

This study demonstrates the successful implementation of an SDN-enabled architecture within a Proxmox virtualized environment using ODL and OVS. The results confirm that OVS instances can reliably register with the ODL controller through OpenFlow, enabling centralized and programmable network management. Furthermore, the study verifies that QoS policies can be applied and enforced directly from the controller, eliminating the need for manual host-level configuration.

Performance testing shows that the implemented rate-limiting mechanism consistently maintained the configured 10 Mbps bandwidth threshold, reducing the unmanaged throughput of 17–18 Gb/s to a stable 8.52–9.15 Mb/s. Despite the substantial reduction, the resulting throughput still falls under the “very good” category based on the TIPHON standard. Packet loss remained below 0.1% across all test scenarios, indicating that QoS enforcement had minimal impact on data integrity while significantly enhancing traffic predictability.

Building on these findings, several directions for future research are recommended. QoS evaluation can be expanded using additional analysis tools such as Wireshark or Nmap to validate results across different measurement methodologies. Performance assessment may also be strengthened by incorporating latency, jitter, and error-rate metrics. For clustered Proxmox deployments, architectural robustness can be improved by allocating a dedicated interface for inter-node communication to isolate SDN control traffic. Further enhancements may include integrating AI-based optimization for dynamic QoS orchestration, implementing visual monitoring dashboards using Grafana and Prometheus, and enabling high-availability mechanisms to maintain controller operability during failure events.

FUNDING INFORMATION

Authors state there is no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Shafira Febriani	✓	✓			✓			✓	✓	✓		✓	✓	✓
Riezkan Aprianda Firmansyah	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

DATA AVAILABILITY

The data supporting the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

[1] A. Liatifis, P. Sarigiannidis, V. Argyriou, and T. Lagkas, "Advancing SDN from OpenFlow to P4: A Survey," *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–37, Sep. 2023, doi: 10.1145/3556973.

[2] R. Wazirali, R. Ahmad, and S. Alhiyari, "SDN-OpenFlow Topology Discovery: An Overview of Performance Issues," *Applied Sciences*, vol. 11, no. 15, p. 6999, Jul. 2021, doi: 10.3390/app11156999.

[3] B. Pfaff *et al.*, "The design and implementation of open vSwitch," *Proceedings of the 12th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2015*, pp. 117–130, 2015.

[4] S. Sezer *et al.*, "Are we ready for SDN? Implementation challenges for software-defined networks," *IEEE Communications Magazine*, vol. 51, no. 7, pp. 36–43, Jul. 2013, doi: 10.1109/MCOM.2013.6553676.

[5] D. S. Ahmed, A. A. Abdulhameed, and M. T. Gaata, "A Systematic Literature Review on Cyber Attack Detection in Software-Define Networking (SDN)," *Mesopotamian Journal of CyberSecurity*, vol. 4, no. 3, pp. 86–135, 2024, doi: 10.58496/MJCS/2024/018.

[6] H. Riggs, A. Khalid, and A. I. Sarwat, "An Overview of SDN Issues—A Case Study and Performance Evaluation of a Secure OpenFlow Protocol Implementation," *Electronics (Switzerland)*, vol. 14, no. 16, 2025, doi: 10.3390/electronics14163244.

[7] H. Kim and N. Feamster, "Improving network management with software defined networking," *IEEE Communications Magazine*, vol. 51, no. 2, pp. 114–119, 2013, doi: 10.1109/MCOM.2013.6461195.

[8] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 1, pp. 236–262, 2016, doi: 10.1109/COMST.2015.2477041.

[9] A. Tootoonchian and Y. Ganjali, "HyperFlow: A distributed control plane for OpenFlow," *Proceedings of the 2010 Internet Network Management Conference on Research on Enterprise Networking*, pp. 3–8, 2012.

[10] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A Survey on Software-Defined Networking," *IEEE Communications Surveys and Tutorials*, vol. 17, no. 1, pp. 27–51, 2015, doi: 10.1109/COMST.2014.2330903.

[11] N. McKeown *et al*., "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*,

[12] A. Montazerolghaem and M. H. Yaghmaee, "Load-Balanced and QoS-Aware Software-Defined Internet of Things," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3323–3337, Apr. 2020, doi: 10.1109/JIOT.2020.2967081.

[13] J. Alqahtani, S. Alanazi, and B. Hamdaoui, "Traffic Behavior in Cloud Data Centers: A Survey," *2020 International Wireless Communications and Mobile Computing (IWCMC)*, pp. 2106–2111, Jun. 2020. doi: 10.1109/iwcmc48107.2020.9148470..

[14] P. Kamboj and S. Pal, "QoS in SDN for Content Delivery using Blockchain based Smart Contract," 2020.

[15] I. E. Kamarudin, M. A. Ameedeen, M. I. Umar Ong, and A. Zabidi, "QSroute : A QoS Aware Routing Scheme for Software Defined Networking," in *8th International Conference on Software Engineering and Computer Systems, ICSECS 2023*, IEEE, Aug. 2023, pp. 388–391. doi: 10.1109/ICSECS58457.2023.10256362.

[16] M. Rostami and S. Goli-Bidgoli, "An overview of QoS-aware load balancing techniques in SDN-based IoT networks," *Journal of Cloud Computing*, vol. 13, no. 1, p. 89, Apr. 2024, doi: 10.1186/s13677-024-00651-7.

[17] G. Wassie, J. Ding, and Y. Wondie, "Traffic prediction in SDN for explainable QoS using deep learning approach," *Scientific Reports*, vol. 13, no. 1, p. 20607, Nov. 2023, doi: 10.1038/s41598-023-46471-8.

[18] T. Moulahi, S. El Khediri, R. Ullah Khan, and S. Zidi, "A fog computing data reduce level to enhance the cloud of things performance," *International Journal of Communication Systems*, vol. 34, no. 9, Jun. 2021, doi: 10.1002/dac.4812.

[19] D. J. Hamad, K. G. Yalda, and N. Ţăpuş, "Performance Assessment of QoS in Software Defined Networking using Meter Table and Floodlight Controller," *International Journal on Informatics Visualization*, vol. 7, no. 3, pp. 631–637, Sep. 2023, doi: 10.30630/joiv.7.3.1288.

[20] S. K. Keshari, V. Kansal, and S. Kumar, "A Systematic Review of Quality of Services (QoS) in Software Defined Networking (SDN)," *Wireless Personal Communications*, vol. 116, no. 3, pp. 2593–2614, Feb. 2021, doi: 10.1007/s11277-020-07812-2.

[21] D. Kreutz, F. M. V. Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015, doi: 10.1109/jproc.2014.2371999.

[22] B. A. A. Nunes, M. Mendonca, X. N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 3, pp. 1617–1634, 2014, doi: 10.1109/SURV.2014.012214.00180.

[23] B. Daneshmand, "Research and Development of QoS methods Based on SDN in 5G/IMT-2020," in *Proceedings of the 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2021*, IEEE, Sep. 2021, pp. 458–464. doi: 10.1109/IDAACS53288.2021.9660998.

[24] M. P. Adrini, M. D. Atmadja, A. W. Yulianto, R. Saptono, and I. Mahfudi, "Implementation of Polinema Streaming Web Server by Integrating ONT and STB," *Jartel: Journal of Telecommunication Networ*k, vol. 15, no. 2, pp. 212–221, Jun. 2025, doi: 10.33795/jartel.v15i2.6859.

[25] K. C. Okafor, M. C. Ndinechi, and S. Misra, "Cyber-physical network architecture for data stream provisioning in complex ecosystems," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 4, Dec. 2021, doi: 10.1002/ett.4407.

BIOGRAPHIES OF AUTHORS

Shafira Febriani received Bachelor of Telecommunication Engineering from Telkom Univeristy in 2020 and later earned a Master's degree in Electrical Engineering with a specialization in Telematics and Telecommunication Networks from Institut Teknologi Bandung (ITB) in 2023. Currently, she is a lecturer in Informatics Engineering at STMIK AMIK Bandung and actively conducts research in the fields of telecommunication networks, network security, and the internet of things (IoT). She can be contacted at email: shafira@stmik-amikbandung.ac.id.

Riezkan Aprianda Firmansyah received his undergraduate in Informatics Engineering from STMIK AMIK Bandung, Bandung, Indonesia, where he is currently completing his final year of study. He was born in April 2002 in Bandung, Indonesia. His undergraduate thesis focuses on software-defined networking (SDN), with particular emphasis on the integration of SDN technologies into the Proxmox VE and the evaluation of network quality of service (QoS). His research interests include virtualization, cloud computing, programmable networks, and data center architectures. He can be contacted at email: riezkanaprianda@gmail.com.