COMPUTER NETWORKING

Software-Defined Networking in Modern Networks: A Systematic Literature Review of Architectures, Benefits, and Deployment Challenges (2020–2026)

Ndi Simon Awan, Akwuma Nathaniel Eru, Solomon Ahiaba, Temitope Olufunmi Atoyebi, A. Prema Kirubakaran

Department of Information Technology and Systems, Faculty of Computing, Nile University of Nigeria, Abuja, Nigeria

Published November 2026

Abstract

Software-Defined Networking (SDN) separates network control from packet forwarding and exposes programmable interfaces for policy, automation, and traffic engineering. This article systematically synthesizes research on SDN in modern networks published from 2020 to 2026. The review followed PRISMA 2020 reporting principles and searched SciSpace, Google Scholar, and PubMed. From 1,417 records, 492 duplicates were removed, 925 unique records were ranked, 135 full texts were assessed, and 71 studies were included in the qualitative synthesis. Distributed or hierarchical control was the most frequently reported architecture (48%), followed by centralized (34%) and hybrid arrangements (18%). The most commonly reported benefits were scalability and flexibility (68%), programmability and simplified management (59%), and enhanced security capabilities through centralized visibility (41%). The principal constraints were control-plane scalability (54%), security vulnerabilities (34%), performance limitations under demanding conditions (31%), deployment complexity (27%), and inter-operability or standardization deficits (23%). The evidence indicates that SDN is no longer only an experimental networking model; it is an enabling platform for cloud, data-centre, IoT, 5G, wide-area, and vehicular systems. However, dependable adoption requires resilient multi-controller designs, secure control channels, interoperable interfaces, realistic performance validation, and phased migration from legacy infrastructure. The review concludes that hybrid and hierarchical deployments currently offer the most practical balance between centralized programmability and operational resilience.

Keywords: Control plane; network programmability; OpenFlow; scalability; security; software-defined networking; systematic literature review

1 Introduction

Cloud computing, data-centre consolidation, the Internet of Things (IoT), 5G, edge platforms, and data-intensive services have changed the scale and volatility of network demand. Traditional networks embed control decisions and packet forwarding in individual routers and switches, usually through vendor-specific operating systems and device-level command interfaces. This design remains dependable for many stable environments, but it becomes difficult to automate, reconfigure, and govern consistently when services must be provisioned rapidly across heterogeneous infrastructure [3, 4].

Software-Defined Networking (SDN) addresses this limitation by separating the control plane from the data plane. Forwarding devices retain packet-handling functions, while a software controller maintains a logically centralized view of topology, state, policy, and traffic. Applications communicate with the controller through northbound interfaces, and the controller programs forwarding devices

through southbound interfaces such as OpenFlow, NETCONF, or related mechanisms. This separation makes network behaviour programmable and permits policies to be tested, automated, and deployed at network scale rather than configured independently on each device [3, 5].

The same architectural features that make SDN attractive also create new engineering risks. A controller can become a performance bottleneck, a high-value attack target, or an enlarged failure domain. Distributed controllers improve resilience and capacity but introduce synchronization cost, consistency problems, and placement decisions. Hybrid deployments can preserve legacy investments, yet they must reconcile two different control philosophies and often expose limitations in interoperability and northbound-interface portability [6]–[10].

Recent SDN research is therefore broad but fragmented. Individual studies commonly emphasize a single controller, protocol, security mechanism, application domain, or simulation environment. This makes it difficult for researchers and practitioners to judge which architectures dominate, which benefits are supported most consistently, and which obstacles remain unresolved. The present review consolidates this evidence for the 2020–2026 period and translates it into deployment implications for modern enterprise and carrier networks.

The review addresses four questions: RQ1, which control-plane architectures dominate contemporary SDN research and deployment; RQ2, which functional advantages are reported relative to traditional distributed networking; RQ3, which technical and operational constraints most often impede deployment; and RQ4, how SDN and traditional networking compare in control, performance, security, scalability, and operational efficiency. The contribution is a transparent study-selection account, an architecture taxonomy, a frequency-based synthesis of benefits and challenges, and practical recommendations for migration and future research.

2 Conceptual and empirical background

2.1 SDN architecture and control models

A conventional SDN stack contains three functional planes. The application plane expresses service requirements and policy; the control plane translates those requirements into network-wide decisions; and the data plane forwards packets according to installed rules. Northbound interfaces connect applications to controllers, while southbound interfaces connect controllers to forwarding devices. Telemetry and management processes cut across the stack and provide the state required for optimization, assurance, and security (Figure 1).

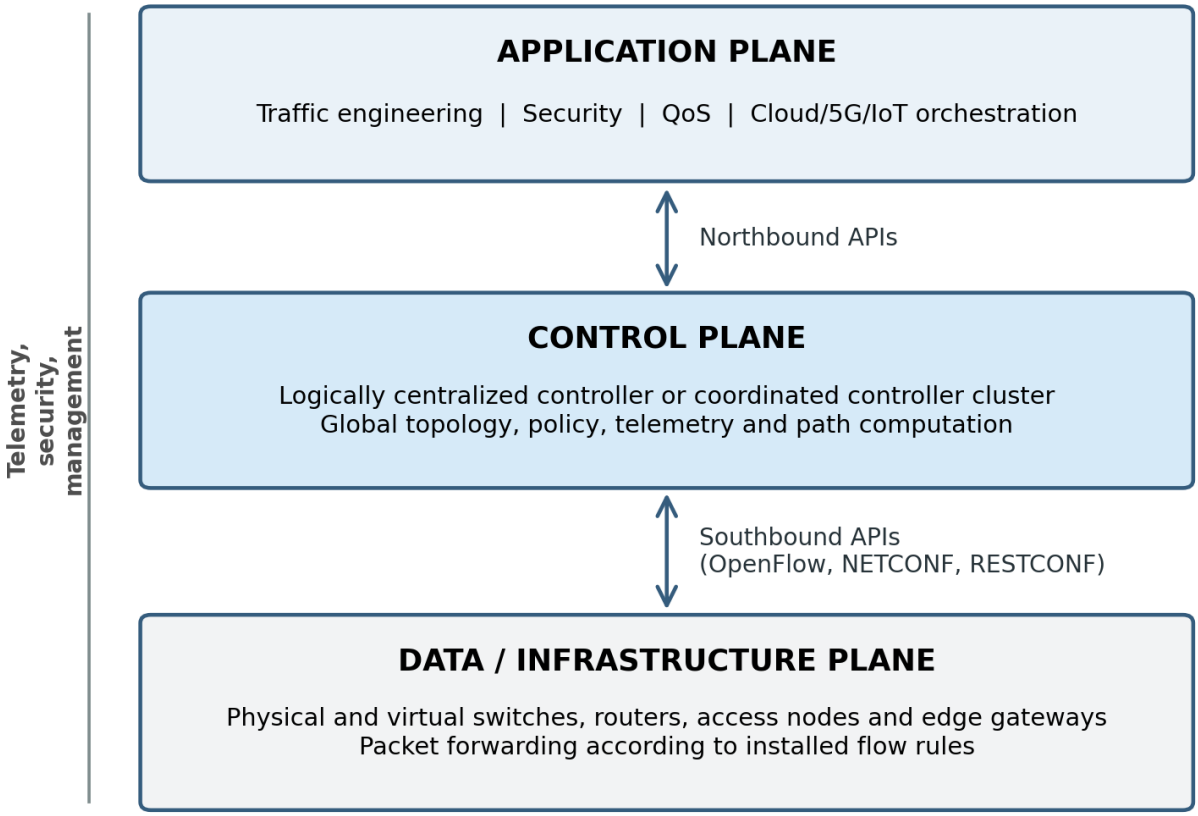


Figure 1. Functional architecture of a modern software-defined network.

Three controller arrangements recur in the reviewed literature. A centralized architecture uses one logical controller and offers a simple global view, but its capacity and availability must be engineered carefully. Distributed or hierarchical designs divide control responsibilities among coordinated controllers and are better suited to large or geographically dispersed networks. Hybrid designs combine SDN-enabled and conventional devices, allowing incremental deployment but making policy consistency and troubleshooting more difficult. Production systems such as Google Orion demonstrate that logical centralization can coexist with distributed microservices, controller domains, and hierarchy [5].

2.2 Benefits relative to traditional networking

The principal value of SDN is operational programmability. High-level policies can be applied across many devices, enabling automated provisioning, dynamic path selection, centralized telemetry, traffic engineering, and more consistent security enforcement. The global network view also supports coordinated load balancing and resource allocation in data centres, mobile networks, IoT systems, and smart-grid environments [4, 9, 11]. SDN can reduce vendor dependence where open interfaces are implemented consistently, although controller and application ecosystems can introduce new forms of platform dependence.

Performance improvements are context dependent. A controller can calculate paths using broader information than any single device, but reactive flow setup adds control latency and large installations can overload flow tables or controller queues. Orion reported up to forty-fold faster convergence after successive production improvements, together with substantial gains in update throughput and availability [5]. Such results demonstrate SDN potential, but they should not be

generalized to small testbeds or unrelated traffic profiles without comparable measurement conditions.

2.3 Security, resilience, and migration constraints

Centralized visibility can improve detection and coordinated response, yet the controller, controller applications, and southbound channel enlarge the attack surface. Threats include denial-of-service against the control plane, malicious or faulty applications, unauthorized flow-rule modification, and exploitation of insecure controller-device communication [6]–[8, 13, 16]. Resilience requires controller redundancy, authenticated and encrypted channels, role separation, validated applications, rate limiting, and monitoring of both data-plane and control-plane behaviour.

Migration is equally important. Organizations rarely replace a functioning network in one step. They operate mixed environments in which legacy routing protocols and SDN policies must coexist. The transition requires capital expenditure, controller and automation skills, compatible hardware, test environments, and rollback plans. Hybrid architectures are therefore not merely temporary compromises; for many enterprises, they are the practical mechanism for controlling migration risk [10, 20].

3 Methodology

3.1 Review design and search strategy

The study adopted a systematic literature review design. PRISMA 2020 guided transparent reporting of identification, screening, eligibility assessment, and inclusion [1]. Protocol design, extraction, and synthesis were also informed by established software-engineering review guidance [2]. Searches were conducted in SciSpace, Google Scholar, and PubMed using combinations of "Software-Defined Networking" or "SDN" with architecture, implementation, framework, benefits, challenges, security, performance, scalability, and traditional networking. The search period was restricted to 2020–2026 and to English-language publications.

3.2 Eligibility, screening, and selection

Studies were eligible when they explicitly addressed an SDN architecture, compared SDN with traditional networking, examined scalability or flexibility, or analysed deployment, performance, interoperability, or security constraints. Papers focused exclusively on traditional protocols, general networking theory without control-data-plane separation, non-English material, and publications outside the review period were excluded. Records were deduplicated, ranked for relevance, screened by title and abstract, assessed in full text, and subjected to a final relevance and redundancy check. Table 1 and Figure 2 summarize the selection process.

Table 1. Review protocol and study-selection outcomes.

Stage	Outcome
Records identified	1,417
Duplicates removed	492
Unique records	925
Advanced after ranking	500

Stage	Outcome
Full texts assessed	135
Eligible after appraisal	121
Final included studies	71

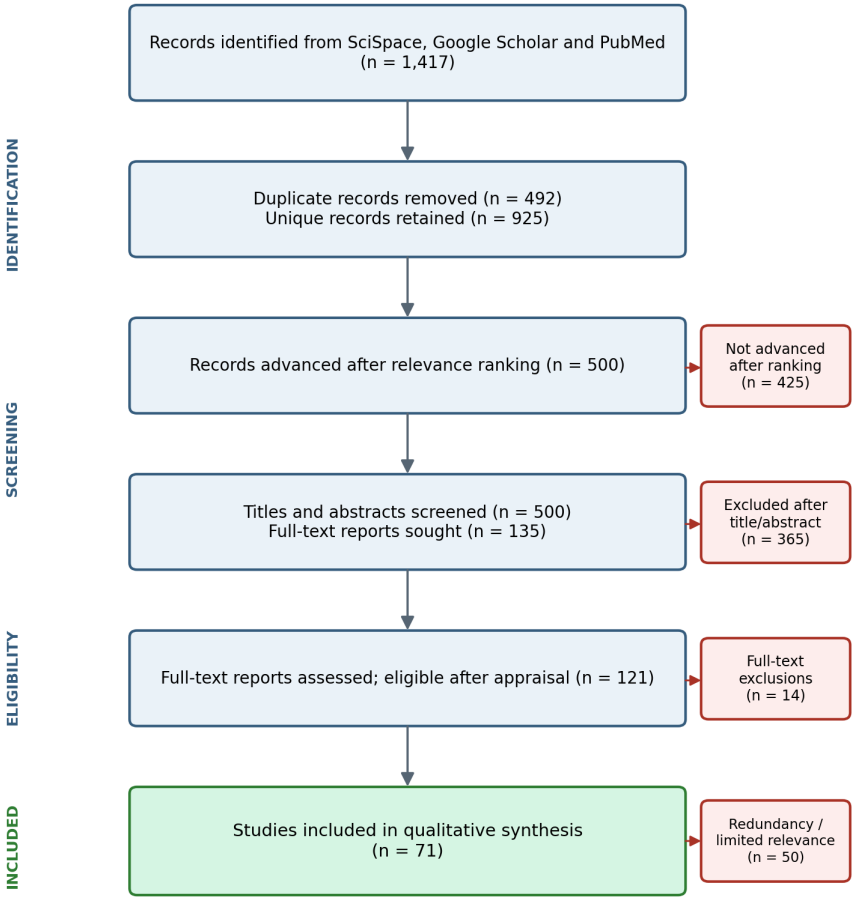


Figure 2. PRISMA-guided study-selection workflow reconstructed from the project records.

3.3 Data extraction and synthesis

Six fields were extracted from each included study: architecture type, reported benefits, challenges or limitations, comparison with traditional networking, study design, and key findings. Descriptive synthesis summarized publication year, study type, application domain, architecture, and controller platform. Thematic synthesis grouped evidence under architecture, programmability and management, scalability, performance, security, automation, cost, interoperability, and migration. Benefit and challenge categories were non-mutually exclusive; therefore, percentages represent the share of studies reporting a theme rather than an effect size or pooled performance estimate.

4 Results and discussion

4.1 Study profile

The final corpus comprised 71 studies. Surveys and reviews were the largest group (n = 29; 41%), followed by architecture proposals (n = 20; 28%), experimental or implementation studies (n = 12; 17%), and simulation-based evaluations (n = 10; 14%). The dated literature peaked in 2023 (n = 16)

and 2025 (n = 15); two extracted records lacked explicit publication dates. The strongest application concentration was IoT (n = 18; 25%), followed by data-centre networks (n = 14; 20%), 5G and mobile networks (n = 12; 17%), cloud computing (n = 11; 15%), wide-area networks (n = 10; 14%), and vehicular networks (n = 6; 8%).

Controller reporting was incomplete. Forty studies (56%) either used multiple platforms or did not specify a controller. Among named platforms, ONOS appeared in seven studies, OpenDaylight and Ryu in six each, custom or proprietary controllers in five, Floodlight in four, and POX or NOX in three. This reporting gap limits direct comparison because controller implementation, clustering, hardware resources, and flow-programming strategy materially affect performance.

4.2 Dominant architectures (RQ1)

Table 2 shows how the included studies are distributed across the three control architectures.

Table 2. SDN architecture distribution in the included studies.

Architecture	*n* (%)	Interpretation
Distributed/ hierarchical	34 (48%)	Multi-controller or tiered control; stronger scale and resilience, with synchronization overhead.
Centralized	24 (34%)	Simple global state and policy control; possible bottleneck and enlarged failure domain.
Hybrid	13 (18%)	SDN plus legacy infrastructure; supports phased adoption but complicates policy consistency.

Distributed and hierarchical architectures were therefore dominant, contradicting the common view of SDN as necessarily dependent on a single controller. The result reflects an evolution from early single-controller prototypes toward clustered, partitioned, and domain-based designs. The central research problem is no longer whether control should be logically centralized, but how global intent can be maintained while computation, state, and failures are distributed. Orion illustrates this balance through a logically centralized model implemented as distributed microservices and control domains [5].

4.3 Reported benefits (RQ2)

Figure 3 shows how often each benefit and challenge theme was reported.

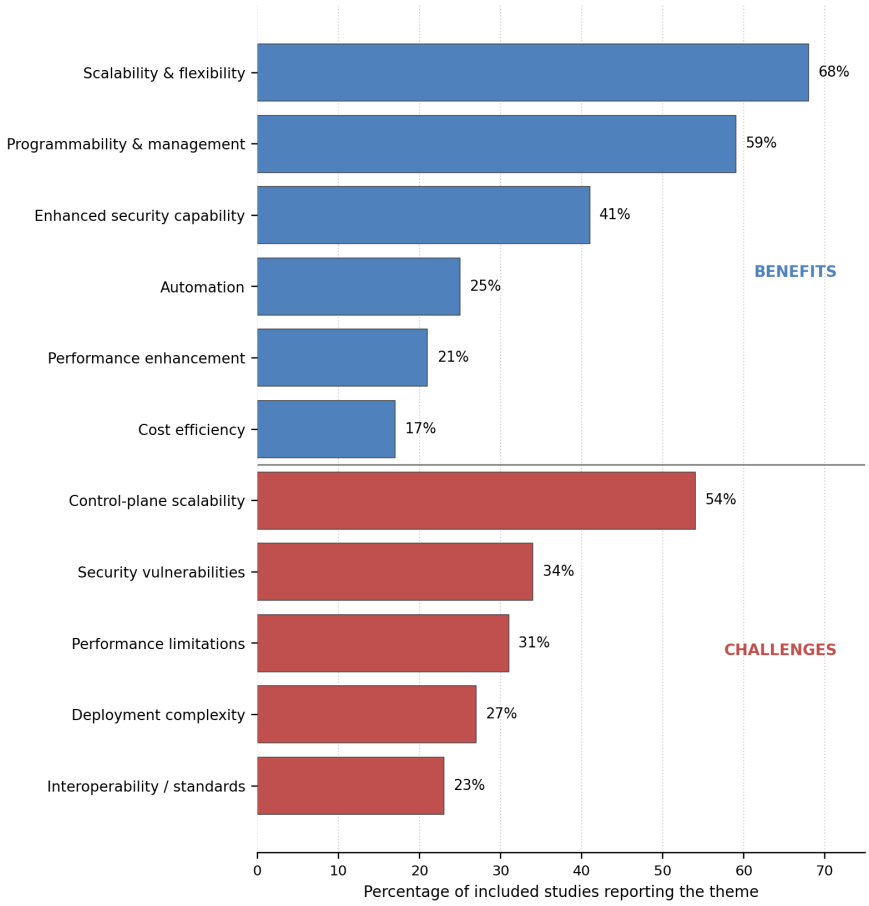


Figure 3. Frequency of major benefit and challenge themes among the 71 included studies.

Scalability and flexibility were the most frequently reported benefits (48 studies; 68%). They included elastic resource allocation, load balancing, rapid topology adaptation, and support for heterogeneous devices. Programmability and management simplification appeared in 42 studies (59%), reflecting policy-based control, reduced device-by-device configuration, and faster service deployment. Twenty-nine studies (41%) described security benefits, especially centralized visibility, network-wide policy enforcement, and coordinated anomaly response.

Automation was reported in 18 studies (25%), performance enhancement in 15 (21%), and cost efficiency in 12 (17%). Several implementation and simulation studies reported throughput or bitrate gains between 13% and 40% under their tested conditions, while the Orion production study documented up to forty-fold faster convergence [5]. These outcomes support SDN's technical value, but the wide variation in testbeds, workloads, controller resources, and metrics prevents a pooled quantitative estimate.

4.4 Persistent challenges (RQ3)

Control-plane scalability was the most common challenge (38 studies; 54%). Single controllers can saturate under large numbers of flow requests, while multi-controller systems exchange state through East-West channels that consume bandwidth and introduce consistency delay. Performance limitations were reported in 22 studies (31%), particularly reactive first-packet latency, flow-table capacity, controller queue saturation, and the trade-off between fine-grained control and signalling overhead.

Security vulnerabilities appeared in 24 studies (34%). The same centralized view that supports comprehensive monitoring makes the controller an attractive target, and insecure applications or interfaces can propagate incorrect policy across the network. Deployment complexity was reported in 19 studies (27%), and interoperability or standardization deficits in 16 (23%). Northbound APIs remain fragmented, controller applications are not always portable, and hybrid networks must reconcile SDN intent with legacy routing behaviour. Security research has proposed machine-learning detection, blockchain-backed rule validation, and controller hardening, but practical deployment must account for computational cost, false positives, and new trust dependencies [7, 8, 16, 23].

4.5 SDN compared with traditional networking (RQ4)

Table 3 compares the two paradigms across seven dimensions.

Table 3. Comparative synthesis of SDN and traditional networking.

Dimension	Traditional networking	Software-defined networking
Control	Distributed per-device logic	Logically centralized policy and global state
Configuration	Manual, device-specific CLI	Programmatic, policy-based automation
Scalability	Expansion through device-by-device design	Elastic control, clustering, and abstraction
Performance	Local forwarding decisions; mature convergence	Global optimization; possible control latency
Security	Distributed monitoring and smaller failure domains	Coordinated visibility, but controller is a high-value target
Interoperability	Often vendor dependent	Open-interface potential; northbound fragmentation persists
Migration	Mature tools and skills	New skills, testing, and phased integration required

The comparison is not a universal verdict in favour of one paradigm. SDN is stronger where networks require frequent policy change, centralized assurance, dynamic resource allocation, or rapid service deployment. Traditional architectures remain attractive where scale is modest, operational patterns are stable, staff and tooling are mature, and the cost of migration exceeds the value of programmability. The most defensible conclusion from the evidence is that SDN expands the design space, but realizes its benefits only when controller capacity, resilience, security, and operational processes are engineered as first-class requirements.

5 Practical implications and research priorities

5.1 Deployment guidance

Architecture should be matched to scale and failure tolerance. Small or laboratory networks may use a centralized controller, but enterprise, carrier, and geographically dispersed deployments should use clustered, distributed, or hierarchical control with tested failover and bounded synchronization overhead. Controller capacity planning should include flow-arrival rate, topology size, telemetry volume, application workload, state-replication cost, and control-channel latency rather than relying only on packet throughput.

Security must be designed into the control plane. Recommended measures include mutual authentication and encryption for controller-device communication, least-privilege access for applications, controller redundancy, rate limiting, application validation, protected management networks, continuous telemetry, and incident procedures that preserve forwarding during controller impairment. Artificial intelligence and machine learning can support anomaly detection, but they should complement deterministic policy, threat intelligence, and human oversight rather than replace them.

Migration should be incremental. A defensible sequence is to inventory applications and dependencies, establish a representative testbed, begin with a bounded domain, define coexistence rules with legacy routing, validate rollback and failover, measure operational outcomes, and expand only after policy and telemetry are stable. Open-source controllers and standard southbound protocols can reduce entry cost, but organizations should still test application portability and avoid assuming that every northbound interface is interoperable.

5.2 Implications for Nigeria and similar environments

For Nigerian universities, enterprises, and telecommunications operators, the evidence supports controlled pilot deployments rather than immediate replacement of established infrastructure. Open-source laboratory environments using Mininet, Ryu, ONOS, or OpenDaylight can build practical skills before production adoption. Curriculum modernization, professional training, and joint industry-academic testbeds are important because the skills gap identified in the literature is as consequential as the technical architecture. National and organizational guidance should emphasize secure interfaces, procurement interoperability, data protection, resilience, and measurable service-level outcomes.

5.3 Limitations and future research

The review is limited to English-language publications retrieved primarily from three platforms. Search-engine ranking influenced which records advanced to screening, and two included records lacked explicit dates. Heterogeneous simulations, controllers, metrics, and application domains prevented meta-analysis. The evidence may also be affected by publication bias, because successful SDN demonstrations are more likely to be reported than failed migrations or neutral performance results.

Future work should prioritize reproducible, multi-vendor and real-network evaluations; standardized northbound application interfaces; low-overhead state synchronization; controller placement under dynamic traffic; security models that remain effective during control-plane overload; and economic evidence on phased migration. Studies should report controller version, hardware resources, topology, traffic model, flow-arrival rate, failure assumptions, and operational metrics such as change lead time, mean time to recovery, policy-error rate, and analyst workload. These details are required to move the field from promising demonstrations toward comparable production evidence.

6 Conclusion

This systematic review synthesised 71 studies of Software-Defined Networking published or retained for the 2020–2026 evidence period. Distributed and hierarchical control was the dominant architectural direction, while scalability, flexibility, programmability, and simplified management were the most frequently reported benefits. The major unresolved issues were control-plane scalability, security exposure, performance under demanding workloads, deployment complexity, and interoper-

ability. SDN should therefore be understood neither as a universal replacement for traditional networking nor as an experimental niche. It is a mature architectural option whose value depends on resilient controller design, secure and open interfaces, realistic performance engineering, and disciplined migration. For most organisations, hybrid and hierarchical deployment provides the most credible path to programmable, scalable, and operationally sustainable networks.

References

[1] Page MJ, McKenzie JE, Bossuyt PM, Boutron I, Hoffmann TC, Mulrow CD, et al. The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *BMJ*. 2021;372:n71. https://doi.org/10.1136/bmj.n71

[2] Kitchenham B, Charters S. *Guidelines for performing systematic literature reviews in software engineering*. EBSE Technical Report EBSE-2007-01. Keele University and Durham University; 2007.

[3] Hu F, Hao Q, Bao K. A survey on software-defined network and OpenFlow: From concept to implementation. *IEEE Communications Surveys & Tutorials*. 2014;16(4):2181–2206. https://doi.org/10.1109/COMST.2014.2326417

[4] Barakabitze AA, Ahmad A, Mijumbi R, Hines A. 5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges. *Computer Networks*. 2020;167:106984. https://doi.org/10.1016/j.comnet.2019.106984

[5] Ferguson AD, Gribble S, Hong CY, Killian C, Mohsin W, Muehe H, et al. Orion: Google's software-defined networking control plane. In: *Proceedings of the 18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*; 2021. p. 83–98.

[6] Hussein A, Chahad L, Adalian N, Chehab A, Elhajj IH, Kayssi A. Software-defined networking (SDN): The security review. *Journal of Cyber Security Technology*. 2020;4(1):1–66.

[7] Ahuja N, Singal G, Mukhopadhyay D, Kumar N. Automated DDoS attack detection in software defined networking. *Journal of Network and Computer Applications*. 2021;187:103108. https://doi.org/10.1016/j.jnca.2021.103108

[8] Shinan K, Alsubhi K, Alzahrani A, Ashraf MU. Machine learning-based botnet detection in software-defined network: A systematic review. *Symmetry*. 2021;13(5):866. https://doi.org/10.3390/sym13050866

[9] Bera S, Misra S, Vasilakos AV. Software-defined networking for Internet of Things: A survey. *IEEE Internet of Things Journal*. 2017;4(6):1994–2008. https://doi.org/10.1109/JIOT.2017.2746186

[10] Ordonez-Lucena J, et al. HybCon: Hybrid SDN-based distributed cloud architecture for 5G. *IEEE Communications Magazine*. 2020;58(7):58–64.

[11] Naseer QK, Raza H, Jeon G. A distributed software defined networking model to improve the scalability and quality of services for flexible green energy internet for smart grid systems. *Computers & Electrical Engineering*. 2020;87:106634. https://doi.org/10.1016/j.compeleceng.2020.106634

[12] Bholebawa IZ, Dalal UD. Performance analysis of SDN/OpenFlow controllers: POX versus Floodlight. *Wireless Personal Communications*. 2018;98(2):1679–1699. https://doi.org/10.1007/s11277-017-4939-1

[13] Ohri P, Neogi SG. Software-defined networking security challenges and solutions: A comprehensive survey. *International Journal of Computing and Digital Systems*. 2022;12(1):383–400.

[14] Sheikh MNAA, Hwang I, Raza P. Software-defined networking (SDN): A review of architecture, security, and challenges. *Applied Sciences*. 2023;13(18):10427.

[15] Shafiq S, Rahman MS, Shaon SA, Mahmud I, Sanwar AS. A review of software-defined networking for Internet of Things inclusive of distributed computing, blockchain and mobile network technology: Basics, trends, challenges and applications. *International Journal of Distributed Sensor Networks*. 2024;2024:9006405.

[16] Fartitchou A, et al. IOTASDN: IOTA 2.0 smart contracts for securing software-defined networking ecosystem. *Sensors*. 2024;24(17):5716. https://doi.org/10.3390/s24175716

[17] Hussain M, Amin R, Gantassi R, Alshehri AH, Frnda J, Raza SM. Efficient handling of ACL policy change in SDN using reactive and proactive flow rule installation. *Scientific Reports*. 2024;14:14976. https://doi.org/10.1038/s41598-024-65721-x

[18] Hodaei A, Babaie S. A survey on traffic management in software-defined networks: Challenges, effective approaches, and potential measures. *Wireless Personal Communications*. 2021;118(2):1507–1534. https://doi.org/10.1007/s11277-021-08100-3

[19] Liu YF, Lin KCJ, Tseng CC. Dynamic cluster-based flow management for software defined networks. *IEEE Transactions on Services Computing*. 2022;15(1):361–371. https://doi.org/10.1109/TSC.2019.2943852

[20] Wijesekara N, Gunawardena S, Dias D. Hybrid SDVN: Strategies for incremental SDN adoption. *IEEE Access*. 2023;11:34512–34526. https://doi.org/10.1109/ACCESS.2023.3264112

[21] Zoradia B, Indumati G. Comparison of software defined networking with traditional networking using NS2 simulator. *International Journal of Information Technology and Security*. 2023;15(3):3–14.

[22] Elumalai J, R S. Review of the SDN architecture with various API controllers. In: *2023 International Conference on Research Methodologies in Knowledge Management, Artificial Intelligence and Telecommunication Engineering (RMKMATE)*; 2023. https://doi.org/10.1109/RMKMATE59243.2023.10369709

[23] Amin R, Costanzo A, Alzabin LR, Aqdus A, Kamarulzaman SF, Alshehri AH, et al. An efficient federated learning based defense mechanism for software defined network cyber threats through machine learning models. *Scientific Reports*. 2025;15:41390. https://doi.org/10.1038/s41598-025-25345-1