Implementation of an Admission Control in SDN Network

Adam Skorecki[1], Jakub Dutkiewicz[1], Bartłomiej Radziun[1], Wiktor Hoffmann[1], Grzegorz Kaczorek[1] and Magdalena Młynarczuk[1][0000-0002-8509-2914]

[1] Gdańsk University of Technology, Faculty of Electronics, Telecommunications and Informatics, Narutowicza 11/12, 80-233 Gdańsk, Poland
magmlyna@pg.edu.pl

Abstract. The rapid growth of network traffic and the increasing complexity of multi-vendor infrastructures pose a significant challenge to traditional network management. Software-Defined Networking (SDN) has emerged as the leading architectural solution to these problems by separating the control plane from the data plane and introducing centralized programmability. Although the SDN was originally designed to optimize packet-switched environments, inherent limitations of packet networks, such as unpredictable packet traffic spikes, require the implementation of robust oversight mechanisms. This paper presents the integration of Admission Control (AC) function with SDN architecture to guarantee the quality of service for different classes. For the first class, it was essential to provide both throughput and loss probability, whereas for the second class, only throughput needed to be guaranteed. These parameters were measured on the SDN network demonstrator with the AC function. The proposed demonstrator was implemented using the Mininet environment, the Ryu controller, and a Django-based application, which evaluates incoming service requests based on real-time network services. The results of the research show that the centralized AC mechanism effectively prevents resource exhaustion and enables guaranteed quality of service for packet flow, proving that Admission Control in the SDN network is essential for maintaining performance integrity in modern telecommunications networks.

Keywords: Software-Defined Networking, Admission Control, Quality of Service.

1 Introduction

Modern telecommunication networks are facing a dynamic surge in traffic volume and escalating architectural complexity, largely driven by the growing number of network devices from multiple vendors. These factors amplify the challenges associated with network orchestration and management, leading to significant operational inefficiencies. The continued reliance on manual configuration of network elements, resulting in increased service provisioning delays, directly contributes to diminished user satisfaction and degraded Quality of Experience (QoE).

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29924-6_4

Software-Defined Networking (SDN) [1] has emerged as the leading paradigm to address these systemic issues by fundamentally restructuring network architectures. The core objective of the SDN is the functional decoupling of the control plane from the data (forwarding) plane, which enables the direct programmability of the network resources. Resource orchestration is centralized and executed by a dedicated network element, the SDN controller.

Decisions regarding network reconfiguration in the SDN network can be driven by applications that monitor the overall network state in real-time. This approach enhances operational flexibility through automated control and mitigates hardware interoperability issues across multi-vendor environments.

The SDN network was initially designed for packet-switched environments, and current traffic dynamics and evolving user expectations necessitate the implementation of robust supervisory mechanisms, such as the Admission Control (AC) function. However, there has been little discussion on implementation of Admission Control function in the SDN network within testbed environments. The integration of AC is vital for effective resource management and ensuring Quality of Service (QoS). This research focuses on the realization of an AC application designed for the SDN-based architecture, supporting two classes of service with specific requirements.

The results of the realization of the SDN demonstrator with AC function are described in this paper, which is organized as follows. Section 2 contains information about admission control function and work related to its subject. Section 3 describes the concept of the SDN demonstrator with the AC function. Section 4 outlines the AC function test scenarios. Section 5 contains the tests results. Section 6 summarizes the paper and presents further work.

2 Admission control function and related work

Admission Control (AC) is a preventive mechanism designed to preserve Quality of Service (QoS) integrity within packet-switched environments. The mechanism evaluates whether a new service request aligns with the established traffic contract and ensures that the mandatory performance objectives can be sustained for the new connection. This decision-making process entails an assessment of the network's capacity to satisfy throughput and QoS metrics, such as packet loss probability, according to the Service Level Agreement (SLA). By validating the requested traffic profile against real-time resource availability, the AC function prevents resource exhaustion and ensures that admitting new flows does not compromise the performance bounds guaranteed to all concurrent active sessions. Such a proactive approach is critical for mitigating network congestion and maintaining quality for different classes of service.

The problem of Admission Control (AC) in Software-Defined Networks (SDN) has been extensively discussed in recent literature. The methods employed to study this issue can be broadly categorized into three main groups: methods based on advanced machine learning algorithms, mathematical traffic optimization models, and implementations within simulation environments.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29924-6_4

A significant portion of contemporary research focuses on the use of Artificial Intelligence for traffic prediction and AC decision-making. In [2], the authors proposed the use of Graph Neural Networks (GNN) in Beyond 5G networks, arguing that Deep Learning-based solutions handle topology complexity better than traditional methods. Similarly, in [3] the authors used a DeepAR probabilistic model to forecast traffic for managing network slices. These methods require substantial computational power and training data, which may hinder their deployment in smaller, dynamically managed environments.

The second group of studies concentrates on theoretical optimization algorithms. The paper [4] analyzes online algorithms for centralized admission control, evaluating their performance compared to offline solutions. Meanwhile, in [5] Partially Observable Markov Decision Processes (POMDPs) were applied, specifically for media delivery networks, where video quality is the critical parameter. Although mathematically robust, these approaches often focus on specific use cases or theoretical performance bounds, frequently overlooking the aspect of interaction with network administrators.

In the context of practical implementation and scalability, [6] presented an analysis of AC mechanism scalability using the Mininet environment and the OpenDaylight controller. However, most Mininet-based solutions focus primarily on the infrastructure layer, rarely integrating control logic with external management applications accessible to the end-user.

In contrast to the previously mentioned works, which rely on complex predictive models, purely theoretical analysis, or remain confined to the controller environment, this paper proposes a pragmatic approach to Admission Control. The novelty of the proposed solution is implementation of the AC as an application in the application plane of the SDN network. The application is based on a mathematical model to determine packet loss probability. Simultaneously, dedicated HTB (Hierarchical Token Bucket) mechanism is implemented in the data plane to guarantee the required throughput. The implemented system integrates the Ryu controller with a Django-based web application, providing QoS guarantees (packet loss probability) and throughput. Such a solution allows for the flexible definition of policies in real-time without the computational overhead characteristic of AI-based methods, while providing satisfactory results in order to guarantee quality for differentiated service classes. Our approach offers a lightweight, Django-integrated solution, making it suitable for real-time computing environments.

3 Network demonstrator

The experimental environment was designed as a comprehensive SDN-based demonstrator to validate the effectiveness of the Admission Control (AC) function. The architecture follows the standard of the SDN paradigm, logically decoupled into three functional planes: the data plane, the control plane, and the application plane. The architecture of the demonstrator is shown in Fig. 1.

The data plane was emulated using Mininet [7], a high-fidelity network emulation tool. The topology consists of five switches interconnected as shown in Fig. 2. To

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29924-6_4

support differentiated Quality of Service (QoS), the Hierarchical Token Bucket (HTB) queuing discipline was implemented at the egress ports of the switches. This allows for precise bandwidth management and traffic separation between two defined classes: a Class 1 with guaranteed throughput and packet loss probability, and a Class_2 with guaranteed throughput. The use of HTB ensures that flows of the Class 1 are protected from the interference caused by Class 2, while the OpenFlow protocol, on southbound interface, enables control of packet forwarding in the data plane.

The control plane is managed by the Ryu SDN controller [8], which operates in a proactive mode. The Ryu controller was selected for its modularity and robust support for the OpenFlow v1.3 protocol [9]. The controller serves as the intermediary between the network hardware and the management logic. It is responsible for maintaining an up-to-date view of the network topology and pushing flow entries (FLOW_MOD messages) to the switches. Furthermore, the controller exposes a northbound interface (NBI) based on the REST (Representational State Transfer) architecture, allowing external applications to retrieve real-time port statistics and flow data in JSON format.

The application plane consists of a dedicated network management system developed using the Django web framework [10]. This layer implements the core Admission Control logic and provides a user-friendly interface for service provisioning. Communication between the Django application and the Ryu controller is realized through asynchronous HTTP requests to the REST API. The application periodically polls the controller to monitor link utilization. When a new service request is submitted via the web interface, the AC module calculates the remaining capacity and evaluates the impact of the new flow on the current network state.

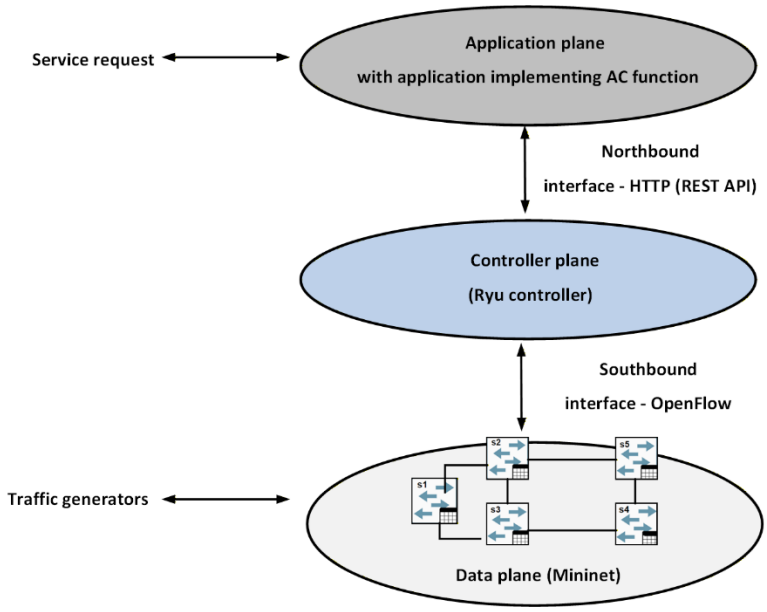

Fig. 1. SDN network demonstrator architecture.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29924-6_4

If admitted, the AC application triggers the controller to update the flow tables in the data plane, thereby establishing a connection with the requested QoS parameters.
To verify the performance and functional validity of the demonstrator, iperf traffic generators were used to simulate traffic load of Class 1 and Class 2, while Wireshark was used for deep packet inspection.

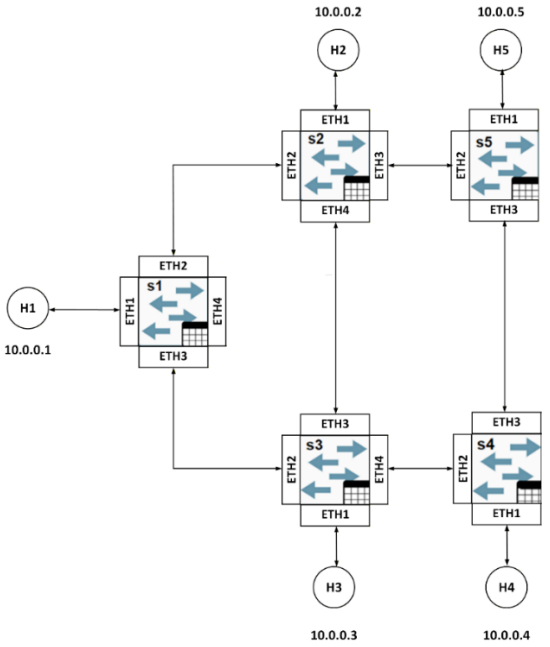

Fig. 2. Mininet network topology.

3.1 SDN AC Application

The core of the proposed system is a web-based application developed using the Django framework, which serves as the application plane in the SDN architecture. The selection of Django was motivated by its modularity, robust Object-Relational Mapping (ORM) system, and seamless integration with HTTP-based libraries, which are essential for communicating with the controller via the northbound interface.

The application's architecture, presented in Fig. 3, comprises several key functional modules:

- Statistics Module,
- Authentication and Authorization Module,
- Admission Control Logic,
- Flow Manager,
- User Interface and Administrative Dashboard.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29924-6_4

The Statistics Module component is responsible for the continuous monitoring of the network state. It utilizes REST API queries to the Ryu controller to retrieve port statistics and flow data in JSON format. Based on this information, the module calculates real-time link utilization.

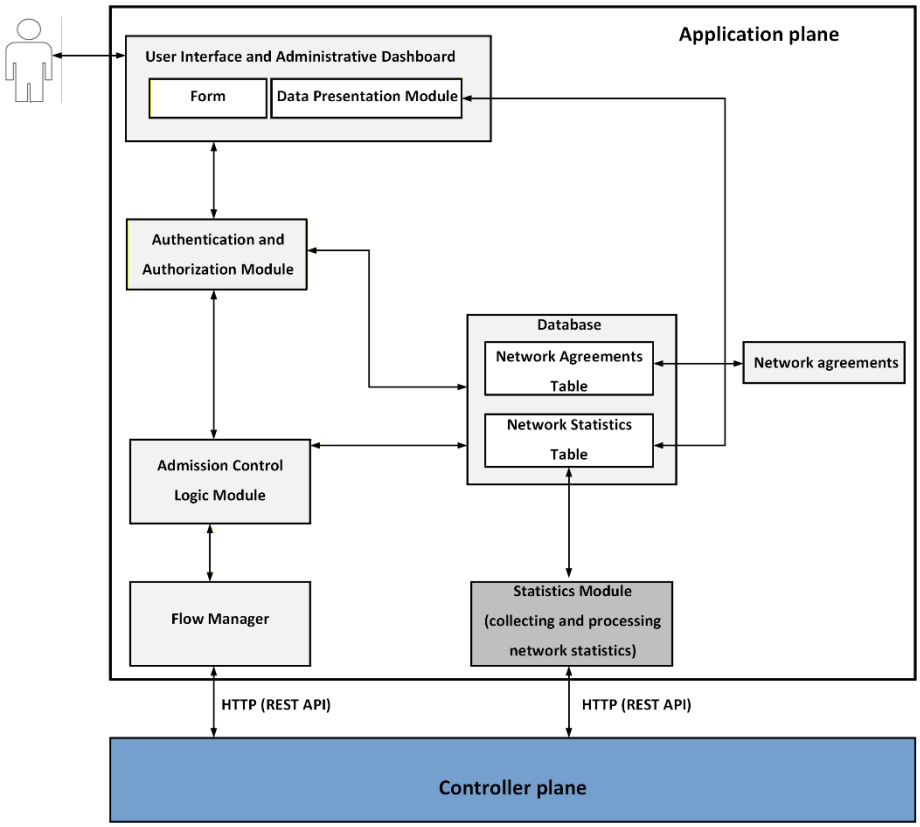

Fig. 3. Application architecture.

The Authentication and Authorization Module confirms the declared identity of the user, verifies the SLA related to the user and ensures SLA compliance, checking if the user has permission to request a specific capacity.

Admission Control Logic Module, acting as the decision-making engine, performs a verification for every incoming service request (req_bw) submitted via the web interface. The service request is rejected if occupied capacity is beyond the *UTILIZATION_TRESHOLD*, there is insufficient bandwidth or calculated loss probability for Class 1 (*p_loss_e2e)* exceeded *LOSS_TRESHOLD*. If degradation of the Class 1 is allowed in the service request, the bandwidth of the Class 2 is used if available.

The flows are removed from flow tables after the hard timeout defined in FLOW_MOD messages, corresponding to *req_duration* in the service request.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29924-6_4

The pseudocode of AC function in the Admission Control Logic Module is presented in Table 1.

Table 1. The pseudocode of AC function.

```
# --- AC Module Logic ---

Set ac_reject_reason = NONE

#--- CONDITIONS FOR AC ---
If capacity_occupied >= (UTILIZATION_THRESHOLD * C):
   ac_reject_reason = "Utilization threshold exceeded"
Else if req_bw > available_bw:
   ac_reject_reason = "Insufficient bandwidth"
Else if requested_class == CLASS_1 AND p_loss_e2e exists:
   If p_loss_e2e >= LOSS_THRESHOLD:
      ac_reject_reason = "Loss threshold exceeded"

#--- DECISION: ACCEPT ---
If ac_reject_reason == NONE:
   Reserve throughput for requested_class
   Schedule throughput release after req_duration
   Exit()

# --- REJECT OR DEGRADATION ---
If degrade_enabled AND requested_class == CLASS_1:
   Compute available bandwidth for CLASS_2
   If req_bw > available_bw_class2:
      Show ERROR: "Not enough bandwidth for degradation"
      Exit()

   # --- DEGRADE TO CLASS 2 ---
   Reserve throughput for CLASS_2
   Schedule throughput release after req_duration
   Exit()

#--- REJECT (no degradation possible) ---
Show ERROR message with ac_reject_reason
Exit ()
```

Flow Manager is responsible for sending HTTP POST request to the SDN controller after a positive admission decision. This module generates the necessary instructions in JSON format for the SDN controller. As a result, FLOW_MOD messages are sent

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29924-6_4

through the southbound interface in order to update the flow tables of the data plane switches, installing the required forwarding rules with the associated QoS parameters.

User Interface and Administrative Dashboard is a module in which the application provides a web-based form for users to define request service parameters. Simultaneously, it offers a monitoring dashboard for administrators, providing view of the flow table entries and network statistics.

4 Tests

The research environment was established using a topology consisting of five OpenFlow switches and five hosts serving as traffic generators and receivers. The experiments were conducted on a workstation equipped with an Intel Core i7-8700 (3.7 GHz) processor and 64 GB of RAM. The Mininet and Ryu controller virtual machines were each allocated 16 GB of RAM and 5 threads.

To implement differentiated traffic handling in service systems, the Hierarchical Token Bucket (HTB) [11], one of the most widely used mechanisms for rate limiting, was provided. The mechanism was configured on each switch output interface. The total capacity was limited to 20 Mbps, with a 5 Mbps guarantee for the Class 1 (VoIP, RTP, Stream) and 15 Mbps for the Class 2 (Data). Traffic was generated using the iperf tool with UDP streams. The experimental evaluation was divided into three scenarios.

Scenario 1: Operation for a Single Traffic Class 1

Test conditions. The experimental setup involves host H1 acting as the traffic generator and host H4 as the receiver. The application initiates a service request for Class 1 – VoIP (req_bw = 4.5 Mb/s, guaranteed throughput = 5 Mb/s, traffic load for Class 1 = 0.9). Traffic of Class 1 is generated using iPerf with source H1 to H4. In Admission Controle Logic Module the following parameters are given:

- UTILIZATION_TRESHOLD = 0.8,
- capacity = 20 Mbps
- LOSS_THRESHOLD = 0.001,
- req_duration = 120 seconds.

Scenario 1 objectives. The primary objective of the first scenario is to evaluate the efficiency of the Admission Control (AC) mechanism for the Class 1 under traffic load 0.9 for this class. The specific objectives include an investigation of the impact of HTB parameters (buffer length (k) and the burst size) on packet loss probability and determining the minimum acceptable buffer and burst sizes required to maintain *LOSS_TRESHOLD*. The burst is defined as the amount of bytes that can be burst at the maximum rate at which a class can send, in excess of the configured rate [12].

By systematically adjusting k and burst parameters within the Mininet environment, and measurement the resulting end-to-end packet loss probability, it is necessary to verify decision-making logic when a flow operates near its maximum guaranteed throughput limit.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29924-6_4

Scenario 2: Operation for Traffic Class 1 and Class 2

Test conditions. The experimental setup involves host H1 acting as the traffic generator and host H4 as the receiver. The application initiates a service request for:

- Class 1 – VoIP (req_bw = 4.5 Mbps, guaranteed throughput = 5 Mbps, traffic load for Class 1 = 0.9),
- Class 2 – Data (req_bw = 13,5 Mbps, guaranteed throughput = 15 Mbps, traffic load for Class 2 = 0.9).

Traffic sources of Class 1 and Class 2 are generated using iPerf with source H1 to H4. In Admission Controle Logic Module the parameters are given as in Scenario 1.

Scenario 2 objectives. The primary objective of the second scenario is to evaluate how the parameters of the Class 2 traffic source affect the quality of service experienced by the Class 1. Moreover, the research specifically examines the impact of the buffer length k and burst parameters on Class 1 while the network is under significant Class 2 load. By systematically adjusting these parameters within the Mininet environment, it is necessary to verify end-to-end packet loss probability for the Class 1. This scenario is designed to verify separation of traffic classes and to ensure that the quality for Class 1 remains uncompromised by Class 2.

Scenario 3: Multi-path and Variable Load Conditions

Test conditions. The experimental setup involves host H1, H3 and H4. In this configuration, hosts H1 and H3 act as simultaneous traffic generators sending data with high Class 1 (p1 and p2) to a common receiver, host H4, as presented in Fig. 4. The application initiates a service request for:

- Class 1 – p1 (traffic load 0.3, 0.5, 0.7, 0.9),
- Class 1 – p2 (traffic load 0.3, 0.5, 0.7, 0.9).

In Admission Controle Logic Module the parameters are given as in Scenario 1. In Mininet HTB parameters are as follow: k = 5, burst = 12000 B.

Scenario 3 objectives. The research focuses on the behavior of the AC mechanism when Class 1 requests are received for different network routes that may share common bottlenecks. The primary objective of the scenario is to determine the precise threshold at which the AC module begins rejecting new service requests to prevent violation of the maximum packet loss probability threshold. This scenario validates the system's capacity.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29924-6_4

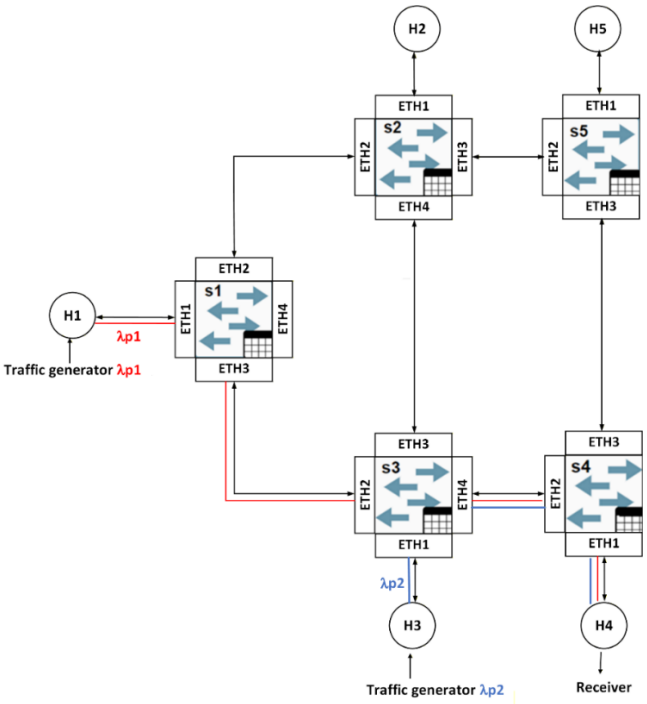

Fig. 4. Multi-path and Variable Load Conditions.

5 Results

The verification of the proposed system was conducted through a multi-stage process encompassing functional software evaluations and empirical research based on the previously defined scenarios. Initial functional tests confirmed the seamless operation of the application layer and its integration with the SDN infrastructure. The Authentication and Authorization module successfully filtered incoming requests by validating them against SLA contracts stored in the database. Deep packet inspection using Wireshark further verified the southbound communication, confirming that the Ryu controller correctly translated application-level decisions into FLOW_MOD messages for the Mininet switches. Additionally, the HTB queue structure and associated filters were validated using the *tc* command, ensuring that traffic was appropriately mapped to the correct service classes based on destination ports.

The research for the first scenario confirmed the correct operation of the AC function. The service request was accepted. The end-to-end packet loss probability in the AC module was computed considering buffer length (*k)* only, excluding the burst parameter,

As illustrated in Table 2, the burst parameter emerged as the primary factor influencing network stability. At a low value of 5000 B the system was unable to compensate for instantaneous traffic spikes, leading to significant packet loss even as buffer sizes

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29924-6_4

were increased. Conversely, increasing the burst to 12000 B successfully eliminated all observed packet loss probabilities proving that adequate traffic shaping is essential for meeting the QoS requirements of Class 1. The buffer length, k, only provided measurable improvements when the burst size was sufficient to mitigate the initial spikes in packet count.

Table 2. Results of the first scenario - packet loss probability for Class 1.

k/burst [B]	5000	9500	12000
1	0,01039604	0,002084962	0
5	0,0082074	0,001323802	0
10	0,00312662	0	0
20	0	0	0
50	0	0	0
100	0	0	0

The second scenario evaluated the quality of the Class 1 under Class 2 load, confirming the effective separation provided by the HTB mechanism. While for Class 2 90% of its 15 Mbps allocation was occupied, the Class 1 maintained a loss probability consistent with the trends observed in the single-flow scenario, as shown in Table 3. This results demonstrate that Class 1 flows remain independent of background data transmissions (Class 2), if appropriate burst size is provided (Table 3).

Table 3. Results of the second scenario - packet loss probability for Class 1.

k/burst [B]	5000	9500	12000
1	0,00990099	0,002084419	0
5	0,00672225	0,001302762	0
10	0,00366610	0,000781657	0
20	0,00216427	0	0
50	0,00136802	0	0
100	0	0	0

Finally, the third scenario tested the system's capacity to prevent network over-subscription across multiple traffic generators and paths. The experimental results, summarized in Table 4, indicate that the Admission Control module successfully maintains its functionality as the cumulative load approaches capacity limits. The system consistently blocked new call service requests sent via application plane once the predicted packet loss probability, calculated in AC Module Logic, exceeded the 0.001 threshold, particularly for flows requiring 70% or more of the guaranteed bandwidth on shared links. These findings confirm the preventive capability of the AC function.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29924-6_4

Table 4. Comparative analysis of the Admission Control function within the third scenario.

Priority class p1	0,3	0,5	0,7	0,9
Priority class p2	0,3	0,5	0,7	0,9
Dropped s1-eth3	0	0	Blocked by SDN AC application	Blocked by SDN AC application
Sent s1-eth3	7664	12766	Blocked by SDN AC application	Blocked by SDN AC application
Dropped s3-eth4	0	17	Blocked by SDN AC application	Blocked by SDN AC application
Sent s3-eth4	7664	24953	Blocked by SDN AC application	Blocked by SDN AC application
End-to-end packet loss probability	0	0,0006808	Blocked by SDN AC application	Blocked by SDN AC application

6 Conclusions

The primary outcome of this research project is the successful implementation of the SDN network demonstrator, featuring a functional Admission Control function that supports two service classes. The experimental environment was established using the Mininet emulator, where a specific network topology was defined alongside Hierarchical Token Bucket (HTB) queues to ensure guaranteed bandwidth for various class services.

In the control plane, the Ryu controller was integrated to facilitate the transmission of real-time network statistics to the management application and to enable the dynamic modification of flow tables within the emulated switches. The statistics were used in order to verify the load traffic in the network. The application plane, developed using the Django framework, effectively executes both Admission Control (AC) and Authentication and Authorization (AA) functions. Furthermore, the system provides a centralized interface for managing network agreements, submitting service requests, and monitoring critical network data, including port statistics and flow table contents. The conducted studies confirmed the correct operation of the implemented Admission Control function. The implemented SDN AC application successfully denied a service request that could not meet the required Quality of Service (QoS). As a result, the most important parameters were indicated. Utilizing Django framework and Ryu enhances the integration and deployment of the proposed solution within academic and experimental testbeds.

While the current Admission Control function is based on throughput and packet loss probability, future development should aim to incorporate additional Quality of Service parameters, such as end-to-end delay. Expanding the demonstrator's capabilities to include dynamic routing protocols is also recommended to enhance its performance and adaptability in more complex network scenarios.

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29924-6_4

Acknowledgments. This work was supported by the Gdańsk University of Technology under the grant DEC-25/1/2023/IDUP/I3b/Ag within the Argentum Triggering Research Grants – "Excellence Initiative - Research University" program.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Framework of Software-Defined Networking, ITU-T Recommendation Y.3300, June 2014.
2. Messaoudi, S., Ksentini, A., Messaoudi, F., Bonnet, C.: GNN-Based SDN Admission Control in Beyond 5G Networks, GLOBECOM 2023 - 2023 IEEE Global Communications Conference, pp. 6103-6108, Kuala Lumpur, Malaysia (2023)
3. Jiang, W., Zhan, Y., Zeng, G., Lu, J.: Probabilistic-Forecasting-Based Admission Control for Network Slicing in Software-Defined Networks, In IEEE Internet of Things Journal, vol. 9, no. 15, pp. 14030-14047, 1 Aug.1 (2022)
4. Leguay, J., Maggi, L., Draief, M., Paris, S., Chouvardas, S.: Admission control with online algorithms in SDN, NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium, pp. 718-721, Istanbul, Turkey (2016)
5. Cao, H., Yin, B., Cao, J., Shi, H., Lu, X.: Analysis of admission control in SDN-based media delivery network, 2016 35th Chinese Control Conference (CCC), pp. 2443-2448, Chengdu, China (2016)
6. Erel, M., Teoman, E., Özçevik, Y., Seçinti G., Canberk B.: Scalability analysis and flow admission control in mininet-based SDN environment, 2015 IEEE Conference on Network Function Virtualization and Software Defined Network (NFV-SDN), pp. 18-19, San Francisco, CA, USA (2015)
7. Mininet Project Contributors. Mininet, http://mininet.org, last accessed 2026/02/16
8. Ryu SDN Framework, https://ryu-sdn.org, last accessed 2026/02/16
9. Open Networking Foundation: ONF TR-502: SDN Architecture, Issue 1, June (2014)
10. The web framework Django, https://www.djangoproject.com, last accessed 2026/02/16
11. Yan, J. Jin, D.: VT-Mininet: Virtual-time-enabled Mininet for Scalable and Accurate Software-Define Network Emulation. In Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research (SOSR '15). Article 27, pp.1–7, Association for Computing Machinery, New York, NY, USA (2015)
12. Tc-htb(8) - Linux man page, https://man7.org/linux/man-pages/man8/tc-htb.8.html, last accessed 2026/03/28

ICCS Camera Ready Version 2026
To cite this paper please use the final published version:
DOI: 10.1007/978-3-032-29924-6_4