Comparative Performance Analysis of TCP Congestion Control Algorithms over IPv4 and IPv6

Vitor Reiel M. Lima[1], **Arthur C. Callado**[1], **Michel S. Bonfim**[1], **Emanuel F. Coutinho**[1]

[1]Programa de Pós-Graduação em Computação (PCOMP)
Universidade Federal do Ceará (UFC) - Quixadá, CE - Brasil

vitorreiel@alu.ufc.br, {arthur, michelsb, emanuel.coutinho}@ufc.br

Abstract. *TCP, the main protocol for reliable data transport on the internet, has undergone numerous modifications over the years, resulting in various versions with specific characteristics for congestion control. Given the rapid growth of the internet and the gradual transition from IPv4 to IPv6, understanding the operation and performance of these versions in different network environments is essential. This study provides a comparative analysis of different TCP versions, which are embedded within the network stack and adapt to hardware buffer constraints, in scenarios operating with either IPv4 or IPv6. The experiments emulate adverse network conditions, including delay, packet loss, and congestion, to evaluate performance metrics during scenario execution. The results highlight the most efficient TCP versions under common network conditions and offer insights into their behavior when configured to operate with IPv4 or IPv6.*

1. Introduction

The Transmission Control Protocol (TCP) remains the main transport protocol responsible for reliable end-to-end data delivery across the Internet, and its operation depends directly on congestion control mechanisms that regulate the sending rate under adverse network conditions [Abadleh et al. 2022, Gomez et al. 2024]. Over time, several TCP congestion control algorithms have been proposed to improve bandwidth utilization and reduce the impact of congestion in different scenarios. Variants such as TCP Reno, CUBIC, BBR, VEGAS, VENO, and WESTWOOD adopt distinct strategies to detect congestion and adjust transmission behavior, which leads to different performance characteristics in the presence of latency, packet loss, and bandwidth constraints [Patel et al. 2020, Biswal et al. 2022, Gomez et al. 2024]. For this reason, evaluating how these algorithms behave under adverse conditions remains a relevant topic in transport-layer research [Patel et al. 2020, Biswal et al. 2022, Szilágyi and Bordán 2020].

Meanwhile, the Internet continues its transition from Internet Protocol version 4 (IPv4) to version 6 (IPv6), due to IPv4 address exhaustion and the increasing number of connected devices [Cañas et al. 2025, Miracle et al. 2025]. Although both protocols provide a best-effort service, IPv6 introduces architectural changes such as a simplified header structure, header checksum removal, and a much larger address space, which may affect packet processing, forwarding behavior, and routing efficiency [Cañas et al. 2025, Miracle et al. 2025, Harly 2025]. These differences do not directly modify the internal logic of TCP congestion control algorithms, but they may indirectly

affect metrics observed by TCP, such as throughput, RTT, packet loss, and retransmissions. Thus, comparing TCP variants over IPv4 and IPv6 under identical conditions is relevant to verify whether performance variations are mainly associated with the congestion control mechanism or with the underlying IP protocol version.

Previous studies have evaluated TCP variants in different scenarios, often using simulation or emulation to analyze metrics such as throughput, round-trip time (RTT), and packet loss [Patel et al. 2020, Biswal et al. 2022, Gomez et al. 2024]. However, much of this literature emphasizes the comparison between congestion control algorithms themselves, without explicitly isolating the impact of the IP protocol version under identical experimental settings [Patel et al. 2020, Biswal et al. 2022]. In addition, differences in topology design, traffic generation, and network parameters across studies make direct comparison difficult and reduce reproducibility [Biswal et al. 2022, Gomez et al. 2024]. Although some studies consider both IPv4 and IPv6, they usually focus on specific environments, such as multipath communication or protocol forwarding performance, rather than on a controlled comparison of multiple TCP congestion control algorithms under the same adverse conditions [Szilágyi and Bordán 2020, Meijers 2023]. Therefore, there is still room for controlled experimental analyses comparing TCP congestion control algorithms over both IPv4 and IPv6 using the same environment and evaluation criteria.

To address this gap, this work presents a controlled experimental evaluation of six widely used TCP congestion control algorithms (RENO, CUBIC, BBR, VENO, VEGAS, and WESTWOOD) over IPv4 and IPv6 using the Mininet network emulator in a reproducible environment [Gomez et al. 2024]. The motivation is not only to compare TCP variants, but also to analyze whether the IP protocol version influences transport-layer performance in scenarios with the same topology, traffic generation process, link configuration, and evaluation metrics. The experiments consider adverse network conditions through variations in latency, packet loss, and simultaneous connections, supporting a direct IPv4/IPv6 comparison in terms of throughput, RTT, packet loss, and retransmissions.

The remainder of this work is organized as follows. Section 2 presents related works. The theoretical background is described in Section 3. The methodology is detailed in Section 4. Section 5 presents the experimental evaluation and discusses the results. Finally, Section 6 concludes the paper and outlines future work.

2. Related Work

The performance of TCP congestion control algorithms has been widely studied under different network conditions, with most works focusing on comparing multiple variants using simulation or emulation environments. For instance, Patel et al. [Patel et al. 2020] evaluated several TCP algorithms using NS-3 and showed that CUBIC achieves higher throughput in high bandwidth-delay networks, while VENO performs better under random packet loss conditions. Similarly, Biswal et al. [Biswal et al. 2022] analyzed multiple TCP variants using NS-2 and showed that CUBIC tends to maximize throughput, whereas delay-based algorithms such as VEGAS reduce packet loss but achieve lower throughput.

However, these studies focus on comparing congestion control algorithms in isolation, without systematically controlling or evaluating the influence of the underlying IP version. As a result, the impact of IPv4 and IPv6 on TCP performance is not clearly isolated under comparable experimental conditions, which limits a more precise understand-

ing of whether the observed performance differences stem from the congestion control mechanisms themselves or from characteristics inherent to the network-layer protocol.

Some works consider both IPv4 and IPv6, such as Szilágyi and Bordán [Szilágyi and Bordán 2020], who evaluated TCP performance in multipath communication scenarios and showed that CUBIC consistently achieves high throughput across configurations. Additionally, architectural differences between IPv4 and IPv6, such as header structure and reduced reliance on NAT, may influence packet processing and network behavior [Beeharry and Nowbutsing 2016]. Nevertheless, these aspects are not evaluated together with multiple TCP congestion control algorithms in controlled environments, making it difficult to isolate the impact of the IP protocol version.

Therefore, there is still a lack of studies that systematically compare multiple TCP congestion control algorithms over IPv4 and IPv6 under identical and reproducible conditions. In this context, this work performs a controlled experimental evaluation of six TCP congestion control algorithms (RENO, CUBIC, BBR, VENO, VEGAS, and WESTWOOD) in the Mininet emulator. Unlike previous studies, which either focus solely on algorithm comparison or analyze IPv4/IPv6 in different settings, this work compares both IP versions under identical configurations, isolating the IP effect on TCP (see Table 1).

Table 1. Comparison with related studies

Study	Algorithms	Environment	Metrics	IPv4 vs IPv6
Patel et al. (2020)	NewReno, VENO, WESTWOOD, BIC, CUBIC	NS-3 simulation	Throughput, CWND	No
Biswal et al. (2022)	Reno, NewReno, BIC, CUBIC, VEGAS, VENO, CTCP	NS-2 simulation	Throughput, RTT, loss	No
Szilagyi and Bordán (2020)	Multiple TCP variants	Multipath network	Throughput, CPU usage	Yes
This work	Reno, CUBIC, BBR, VENO, VEGAS, WESTWOOD	Mininet emulation	Throughput, RTT, loss, retransmissions	Yes

3. Theoretical framework

This section presents the main concepts supporting the evaluation conducted in this work, focusing on the emulation environment and the TCP congestion control algorithms.

3.1. Mininet

According to its documentation [Mininet 2026], Mininet is a network emulation tool that enables the creation of virtual topologies composed of hosts, switches, controllers, and links within a single system. Its support for scripted and reproducible configurations makes it suitable for comparative studies under controlled network conditions.

3.2. TCP Congestion Control Algorithms

TCP congestion control regulates transmission rate according to network conditions, while IPv4 and IPv6 may indirectly influence observed metrics such as throughput, RTT, packet loss, and retransmissions due to differences in packet processing/forwarding. This

study evaluates six widely used TCP congestion control algorithms under identical network conditions: RENO, CUBIC, BBR, VEGAS, VENO, and WESTWOOD. RENO is a loss-based algorithm that relies on slow start, congestion avoidance, fast retransmit, and fast recovery [Niu and Liu 2012]. CUBIC, the Linux default, uses cubic window growth to improve performance in high bandwidth-delay networks [Miyazawa et al. 2020]. BBR adopts a model-based strategy by estimating bottleneck bandwidth and propagation time [Miyazawa et al. 2018]. VEGAS is delay-based and adjusts the congestion window according to RTT variation [Abed et al. 2011]. VENO extends RENO distinguishing congestion losses from random losses [Zhou et al. 2006]. WESTWOOD estimates available bandwidth from returning ACKs to improve recovery after loss [Chen et al. 2015].

4. Methodology

This section describes the methodology used to execute the experiments and collect metrics (Section 3.2). Network scenarios were automated using *scripts* and config files.

4.1. Experiment stages

A script[1] was developed to conduct experiments (see Figure 1). It installs all necessary dependencies for configuration, execution, and metric collection. Next, parameters such as transfer rate, latency, packet loss, number of repetitions, and simultaneous connections are defined. Then the network topology is created in Mininet, and data transfer is initiated between hosts with iPerf3[2]. A JSON file with captured information is generated after execution and converted into .CSV to generate graphs and tables of the collected metrics.

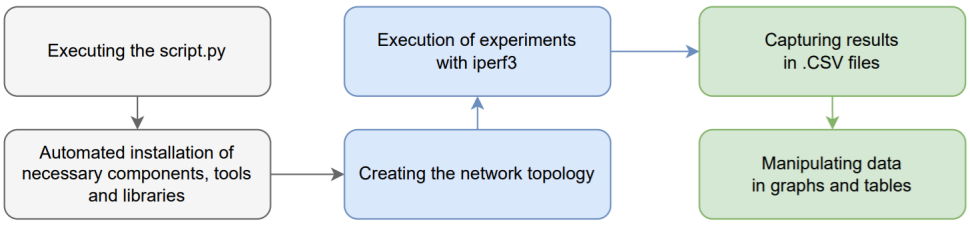

Figure 1. Workflow for experiment execution and data collection

Each execution is performed in a new network scenario. After .CSV generation, the Mininet environment is destroyed and recreated with the same parameters, ensuring independence between runs, and repeated according to the number of replications.

4.2. Network configuration

Figure 2 shows the network topology fixed and generated by executing the script described in Section 4.1. Devices are preconfigured with IPv4 and IPv6 addresses, and the necessary connections between them. Host 1 (h1) acts as traffic source, while Host 2 (h2) acts as destination. The routers (r1 and r2) forward packets between hosts.

The following commands enable IPv4 and IPv6 packet forwarding on the routers:

```
sysctl -w net.ipv4.ip_forward=1
sysctl -w net.ipv6.conf.all.forwarding=1
```

[1]https://github.com/vitorreiel/TCP-ExBench.git
[2]https://iperf.fr/

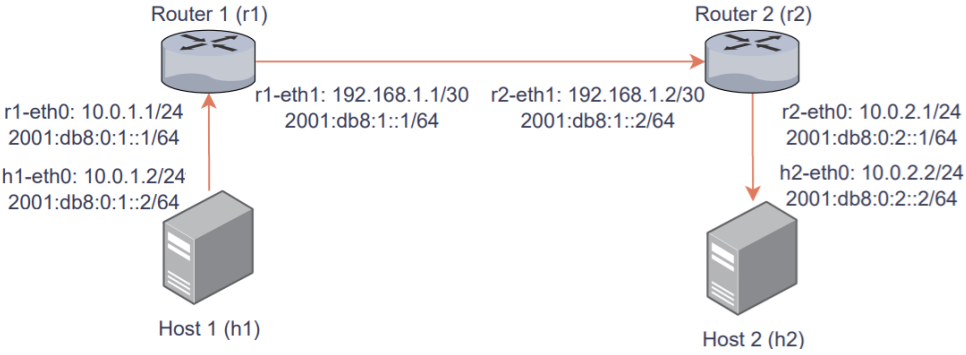

Figure 2. Emulated network topology used in the experiments

The TCP congestion control algorithm was configured at the host level (e.g., enabling TCP VEGAS in IPv4) using:

```
sysctl -w net.ipv4.tcp_congestion_control=vegas
```

The traffic generation used iPerf3. The server was background executed on h2:

```
h2 iperf3 -s -p 5201 &
```

The client was executed on h1, sending data for 30 seconds at a rate of 1 Gbps with 12 parallel streams and exporting results in JSON format:

```
h1 iperf3 -c 10.0.2.2 -p 5201 -t 30 -b 1G -P 12 -J
```

4.3. Performance Metrics

The metrics used in this study are essential for assessing TCP performance under experimental conditions. The following metrics are collected and analyzed:

- **Throughput**: Represents the data transfer rate over a given period, indicating how efficiently each TCP version utilizes the available bandwidth.
- **Latency (RTT)**: Measures the round-trip time between hosts, reflecting the responsiveness of the network.
- **Packet Loss**: Represents the percentage of packets lost during transmission, indicating reliability under adverse conditions.
- **Retransmission**: Is the protocol's ability to recover from loss and maintain flow.

4.4. Test scenarios

Experimental parameters were chosen to represent common network conditions. Latencies of 10 ms, 50 ms, and 100 ms were adopted to cover low, medium, and high-delay conditions, while a packet loss rate of 1% represents a moderately degraded network. Scenarios without packet loss were also included as baseline, and multiple simultaneous connections were considered to emulate flow competition for shared resources. TCP behavior is commonly evaluated under variations in delay, loss rate, and number of flows in controlled and reproducible environments [Gomez et al. 2024].

Scenarios were designed to evaluate TCP versions (RENO, CUBIC, BBR, VENO, VEGAS, and WESTWOOD) under IPv4 and IPv6 (see the topology description in Section 4.2. Latency and packet loss variations are only applied to the link between routers r1 and r2, while other links operate at 1 Gbps without additional delay or loss.

See Table 2 for the complete set of experimental scenarios. Scenarios 1–6 evaluate single-flow conditions, while Scenarios 7–12 introduce multiple simultaneous connections to generate congestion. Each execution lasts 30 seconds, and the experiment is repeated 100 times for each TCP version and IP protocol, resulting in a total of 14,400 executions (100 replications $\times$ 6 TCP versions $\times$ 2 IP protocols $\times$ 12 scenarios).

Table 2. Network scenarios used in the experiments

Scenario	Link	Delay	Loss	Simult. Conn.
1	1 Gbps	10 ms	1%	1
2	1 Gbps	50 ms	1%	1
3	1 Gbps	100 ms	1%	1
4	1 Gbps	10 ms	0%	1
5	1 Gbps	50 ms	0%	1
6	1 Gbps	100 ms	0%	1
7	1 Gbps	10 ms	1%	12
8	1 Gbps	50 ms	1%	12
9	1 Gbps	100 ms	1%	12
10	1 Gbps	10 ms	0%	12
11	1 Gbps	50 ms	0%	12
12	1 Gbps	100 ms	0%	12

5. Experiments and Results

5.1. Quantitative Analysis

The process of capturing the data for analysis was carried out through the execution of the previously configured script (Section 4.1), automating the creation of the emulated network topology and running experiments until a total of 100 samples for each TCP and IP version per scenario are collected, storing the metric data in datasets[3]. All experiments were orchestrated by local machine with the following specifications: Ubuntu 22.04.5 LTS; 12th Gen Intel(R) Core(TM) i9-12900F; NVIDIA GeForce RTX 3080 Ti; 128 GB DDR4 RAM; 4 TB NVMe SSD; 4 TB HDD. The results are presented below:

Figure 3 presents the analysis of the average throughput (Mbps) achieved by the TCP versions under IPv4 and IPv6 across 12 scenarios with variations in latency, packet loss, and simultaneous connections. The X-axis represents the scenarios, while the Y-axis shows the throughput, capped at 1000 Mbps (1 Gbps). The results show that TCP BBR, both with IPv4 and IPv6, demonstrates the best performance in most scenarios, standing out in conditions with packet loss, where it maintains significantly higher throughput. TCP VENO performs well in scenarios with a single connection and no losses, excelling in Scenario 5 under IPv4. TCP CUBIC shows positive results in scenarios with moderate latency and no losses, although it lags behind VENO and BBR. TCP WESTWOOD maintains average performance in most scenarios, while TCP RENO shows good values in high-latency, loss-free scenarios, with its best performance in Scenario 6 under IPv4. However, RENO struggles in scenarios with packet loss, ranking among the worst performers. TCP VEGAS, on the other hand, exhibits the poorest results, consistently

[3]https://github.com/vitorreiel/TCP-ExBench-dataset.git

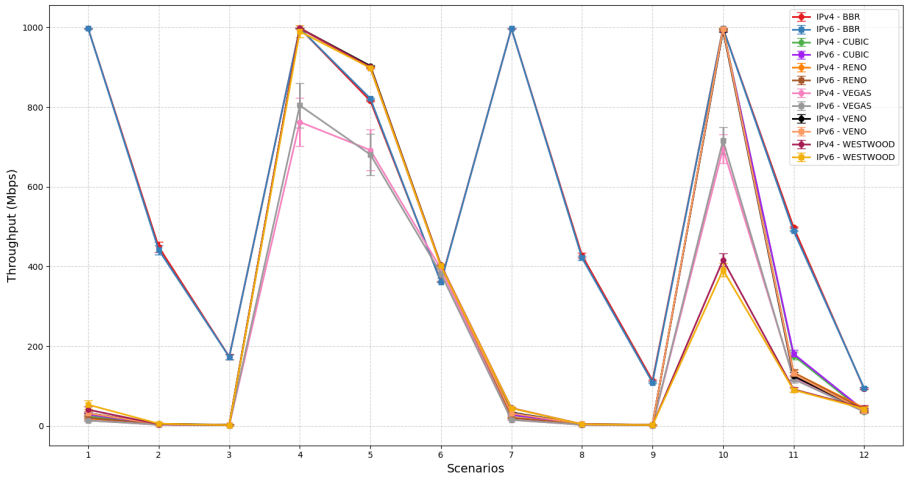

Figure 3. Average throughput of TCP variants across network scenarios

ranking at the bottom in all scenarios. Additionally, the protocols under IPv4 and IPv6 show very similar values across all TCP versions. The results highlight the performance of TCP BBR, which can be explained by its congestion window adjustment mechanism. Unlike other versions, BBR does not reduce the window even under network loss conditions, which justifies its superiority in the scenarios evaluated with the throughput metric.

Figure 4 illustrates the average RTT in microseconds. In scenarios with lower latency (scenarios 1, 4, 7, and 10, with 10 ms of delay), the average RTT values are significantly lower, directly reflecting the network's low latency, with all TCP versions showing similar results. As the delay increases (scenarios 2, 5, 8, and 11, with 50 ms, and scenarios 3, 6, 9, and 12, with 100 ms), RTT values grow proportionally, peaking in the scenarios with 100 ms of delay (scenarios 3, 6, 9, and 12). Although the differences are minimal, VEGAS shows the highest RTT values in most scenarios, while BBR shows the lowest, which may be related to their respective congestion control mechanisms. TCP versions under IPv4 and IPv6 exhibit almost identical behavior, indicating that the IP protocol has minimal impact on RTT compared to the variations introduced by delay and scenario configuration. Thus, the results show that RTT is influenced by the delay configured in the scenarios and that differences between TCP protocols are minimal.

Figure 5 shows the packet loss (%) for different versions of TCP across 12 scenarios. In scenarios 1 to 3, where there is only one connection and a configured loss of 1%, the average packet loss across all TCP versions remains close to 1% with minimal variation, as expected. In scenarios 4 to 6, where no packet loss is configured, the recorded loss is virtually zero, indicating the efficiency of the TCP versions in avoiding losses in stable networks. In scenarios 7 to 9, which introduce 12 simultaneous connections and a 1% loss, the impact of competition is evident in the BBR version, showing more than 10% loss, especially with IPv6, while other versions maintain packet loss close to 1%. The most divergent behavior occurs in scenarios 10 to 12. Despite the configured loss being zero, there is a significant increase, particularly in scenario 12, with loss values exceeding 20% in the BBR version, with IPv6 showing a slightly higher loss than IPv4. VEGAS and RENO perform better in these scenarios, recording lower losses, while BBR appears to be more sensitive, particularly with high latency and multiple connections. The comparison

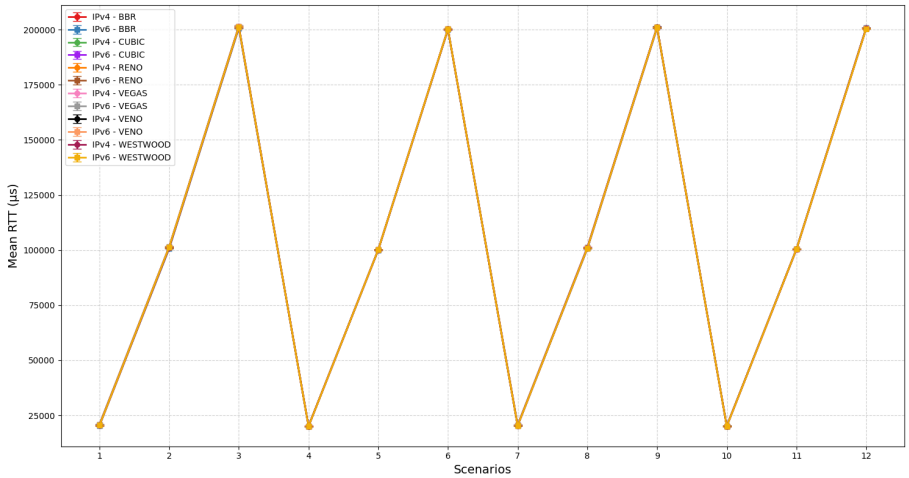

Figure 4. Average RTT of TCP variants across network scenarios

between IPv4 and IPv6 reveals minimal differences, with similar behaviors in almost all scenarios. These results highlight that packet loss is influenced not only by the configured loss and delay but also by each protocol's ability to handle high competition.

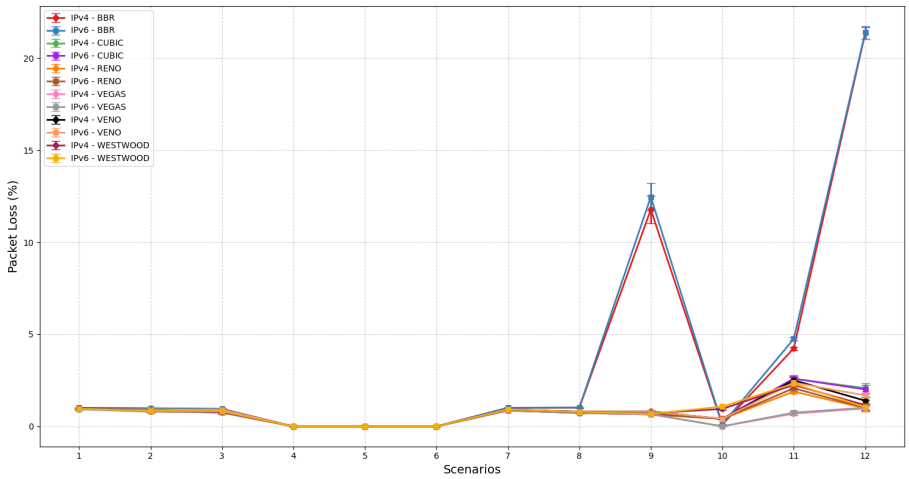

Figure 5. Packet loss of TCP variants across network scenarios

Figure 6 shows the average number of packet retransmissions during the experiments. In scenarios 1 to 3, the number of retransmissions is small for most versions, except for BBR, which shows higher values. In scenarios 4 to 6, where no packet loss is present, no retransmissions were observed. In scenarios 7 to 9, which introduce simultaneous connections, retransmissions increase, reflecting greater competition for bandwidth. BBR continues to have higher numbers, particularly with IPv6, while the RENO, CUBIC, and VEGAS protocols show lower values, suggesting more efficient retransmission strategies. The most pronounced behaviors occur in scenarios 10 to 12, where high latency (mainly in scenario 12) combined with simultaneous connections results in a considerable number of retransmissions, with emphasis on CUBIC and especially BBR, which leads to retransmissions due to its throughput-maximizing model, causing even more retransmissions. The differences between IPv4 and IPv6 are minimal, except for

BBR, where IPv6 showed more retransmissions than IPv4. These results highlight that retransmissions are primarily influenced by packet loss and resource competition, with each TCP version exhibiting distinct behaviors based on its congestion control strategy.

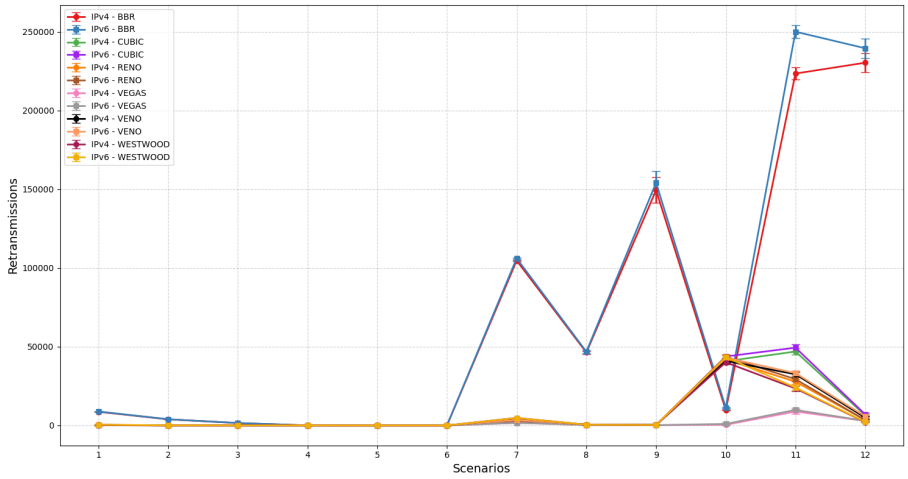

Figure 6. Average retransmissions of TCP variants across network scenarios

5.2. Descriptive Statistics

To determine whether there is a significant difference in the performance of TCP versions between IPv4 and IPv6 under adverse conditions, a hypothesis test was applied. In this study, the Shapiro-Wilk (SW) normality test was used to analyze the data distribution and guide the decision on the most appropriate statistical approach. The Shapiro-Wilk test was chosen due to its sensitivity in detecting deviations from normality in the samples.

5.2.1. Goodness of Fit Test

Table 3 presents the results of the SW normality test applied to the overall metrics averages by combining results obtained from all TCP versions across all scenarios, differing only by the IP version used. This approach facilitates hypothesis validation by providing descriptive statistics such as mean, median, mode, standard deviation, W statistic, and p-value. The results indicate p-values below 0,05, suggesting that the data population does not follow a normal distribution. Consequently, parametric statistical methods were not applicable, leading to the use of non-parametric approaches for analysis.

5.2.2. Hypothesis Testing

The non-parametric Mann-Whitney U (MW) test was selected for the analysis as it is suitable for comparing independent samples and because the collected data do not follow a normal distribution. The MW test compares the medians of the samples without assuming normality and can be applied to both large and small samples. To conduct the MW test, the following hypotheses were formulated, as shown in Table 4.

Table 3. Shapiro-Wilk (SW) normality test

IP	Metric	Mean	Median	Mode	Std. Dev.	*W* (SW)	*p-value* (SW)
IPv4	Throughput	319	80	997,8	390	0,74	7,75830e-75
IPv6	Throughput	319	82	997,8	390	0,738	6,55445e-75
IPv4	RTT	107168	100446	100037	73302,4	0,798	8,42381e-70
IPv6	RTT	107165	199482	20094	73324,3	0,798	8,75814e-70
IPv4	Packet Loss	1,2	0,74	0	2,9	0,343	3,47541e-96
IPv6	Packet Loss	1,2	0,74	0	2,9	0,348	4,93265e-96
IPv4	Retransmissions	15376	274	0	43242,8	0,403	7,66454e-94
IPv6	Retransmissions	16239	275	0	45994,1	0,4	5,56768e-94

Table 4. Performance hypotheses of IP protocols in TCP versions

Hypothesis	Description
H_0	TCP versions exhibit similar performance when using IPv4 or IPv6.
H_1	TCP versions exhibit better performance when using IPv4.
H_2	TCP versions exhibit better performance when using IPv6.

5.2.3. Validation of hypothesis tests

Table 5 presents the results of the MW hypothesis test. Results show that when evaluating the metrics Throughput, Mean RTT, Packet Loss, and Retransmissions, the Mann-Whitney statistical test did not identify significant differences between the performance of TCP versions when using IPv4 and IPv6 protocols in network scenarios with adverse conditions. The p-values above the threshold of 0,05 lead to the acceptance of the null hypothesis (H_0), indicating that there is no evidence to assert that the performance of TCP versions differ significantly between the two protocols. This result suggests that, in the controlled scenarios evaluated, the congestion control strategy had a stronger influence on the measured performance than the IP protocol version itself. Thus, the observed differences are mainly associated with how each TCP variant responds to delay, packet loss, and simultaneous connections, rather than with IPv4 or IPv6 alone.

Table 5. Hypothesis test on the performance of metrics between IP versions

Metric	Statistic	p-value	Conclusion
Throughput	26771453	0,619	H_0
RTT	27022599	0,138	H_0
Packet Loss	26470672,5	0,491	H_0
Retransmissions	26479749	0,514	H_0

5.3. Threats to Validity

Some limitations of this study should be acknowledged. First, the experiments were conducted using a network emulation environment rather than physical networking hardware. Although Mininet provides a realistic environment for evaluating network protocols, certain hardware-specific behaviors may not be fully reproduced.

Second, the experimental topology includes a single traffic source and destination pair. While this design allows a controlled evaluation of congestion control behavior, it limits the analysis of fairness among multiple competing flows in complex topologies.

Finally, the strict control of the experimental environment is both a methodological choice and a limitation. This approach enables systematic comparisons among TCP variants, but prevents the observation of common Internet factors, such as heterogeneous routing policies, cross-traffic patterns, dynamic link conditions, and route changes. Therefore, the results should be interpreted as evidence from a controlled emulation environment rather than as a direct generalization to large-scale operational networks.

6. Conclusion

This study presented an experimental evaluation of several TCP congestion control algorithms operating under IPv4 and IPv6 in controlled adverse network conditions. Using the Mininet network emulator, six TCP variants (RENO, CUBIC, BBR, VENO, VEGAS, and WESTWOOD) were analyzed across multiple scenarios involving variations in latency, packet loss, and simultaneous connections.

The experimental analysis showed that TCP BBR achieves the highest throughput in most scenarios, particularly in environments with packet loss. However, its more aggressive congestion control behavior also leads to increased packet loss and retransmissions under high contention conditions. In contrast, algorithms such as RENO and VEGAS exhibit more conservative behavior, although they may experience reduced performance in high-latency scenarios.

The statistical analysis based on the Shapiro–Wilk and Mann–Whitney U tests indicated no statistically significant differences between IPv4 and IPv6 in terms of throughput, RTT, packet loss, or retransmissions. This suggests that, in the controlled scenarios evaluated, the behavior of the TCP variants was influenced more strongly by their congestion control strategies than by the underlying IP protocol version. This result should not be interpreted as an argument against IPv6 adoption, since migration decisions involve factors such as addressing, scalability, management, and long-term Internet evolution.

Overall, this work contributes to the understanding of TCP congestion control behavior under adverse network conditions by providing a controlled comparison of multiple TCP variants over IPv4 and IPv6 under identical experimental settings. In this sense, the study helps clarify that the main performance differences observed in the evaluated scenarios are associated mainly with the congestion control algorithm itself rather than with the IP version adopted.

Future work include the evaluation of more congestion control algorithms, the use of more complex topologies with multiple traffic sources, competing flows, cross-traffic patterns, and dynamic links, as well as the investigation of TCP behavior in emerging environments such as multipath communication and software-defined networks (SDN).

References

Abadleh, A., Tareef, A., Btoush, A., Mahadeen, A., Al-Mjali, M. M., Alja'Afreh, S. S., and Alkasasbeh, A. A. (2022). Comparative analysis of tcp congestion control methods. In *2022 13th International Conference on Information and Communication Systems (ICICS)*, pages 474–478.

Abed, G. A., Ismail, M., and Jumari, K. (2011). Influence of parameters variation of tcp-vegas in performance of congestion window over large bandwidth-delay networks. In *The 17th Asia Pacific Conference on Communications*, pages 434–438.

Beeharry, J. and Nowbutsing, B. (2016). Forecasting ipv4 exhaustion and ipv6 migration. In *2016 IEEE International Conference on Emerging Technologies and Innovative Business Practices for the Transformation of Societies*, pages 336–340.

Biswal, S. P., Gupta, S., Kumar, A., and Patel, S. (2022). Comparative analysis of compound tcp with various end-to-end congestion control mechanisms. In *2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*, pages 1762–1767.

Cañas, R., Henríquez-Miranda, C., and Silva, J. (2025). Comparative analysis of ipv4 and ipv6 to improve quality of service on a university wireless network. *CESTA*, 6(1).

Chen, Z., Liu, Y., Duan, Y., Liu, H., Li, G., Chen, Y., Sun, J., and Zhang, X. (2015). A novel bandwidth estimation algorithm of tcp westwood in typical lte scenarios. In *2015 IEEE/CIC International Conference on Communications in China (ICCC)*, pages 1–5.

Gomez, J., Kfoury, E. F., Crichigno, J., and Srivastava, G. (2024). Evaluating tcp bbrv3 performance in wired broadband networks. *Computer Communications*, 222:198–208.

Harly, S. (2025). Performance analysis of ipv4 and ipv6 in network traffic management using various queuing mechanism algorithms. *RIGGS: Journal of Artificial Intelligence and Digital Business*, 4(2):1605–1609.

Meijers, I. (2023). Comparison of ipv4 and ipv6 forwarding performance in virtual and hardware routers. In *2023 IEEE 64th International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)*, pages 1–4.

Mininet (2026). *Mininet Overview. https://mininet.org/overview/. Accessed 11 Jan. 2025*.

Miracle, N. O., Peace, O. N., Michael, O., chinenye Love, E.-O., and Chinasa, I. S. (2025). Performance evaluation of ipv4 vs ipv6 in wireless networks.

Miyazawa, K., Sasaki, K., Oda, N., and Yamaguchi, S. (2018). Cycle and divergence of performance on tcp bbr. In *2018 IEEE 7th International Conference on Cloud Networking (CloudNet)*, pages 1–6.

Miyazawa, K., Yamaguchi, S., and Kobayashi, A. (2020). Drain asynchronization for cyclic throughput fluctuation of cubic tcp and tcp bbr. In *2020 IEEE 9th Global Conference on Consumer Electronics (GCCE)*, pages 588–589.

Niu, L. and Liu, D. (2012). Improvement of tcp reno protocol in wireless networks. In *International Conference on Automatic Control and Artificial Intelligence (ACAI 2012)*, pages 1210–1213.

Patel, S., Shukla, Y., Kumar, N., Sharma, T., and Singh, K. (2020). A comparative performance analysis of tcp congestion control algorithms: Newreno, westwood, veno, bic, and cubic. In *2020 6th International Conference on Signal Processing and Communication (ICSC)*, pages 23–28.

Szilágyi, S. and Bordán, I. (2020). The effects of different congestion control algorithms over multipath fast ethernet ipv4/ipv6 environments. In *International Conference on Applied Informatics*.

Zhou, B., Fu, C. P. n., Chiu, D.-m., Lau, C. T., and Ngoh, L. H. (2006). A simple throughput model for tcp veno. In *2006 IEEE International Conference on Communications*.