Hindawi
Security and Communication Networks
Volume 2021, Article ID 9963829, 11 pages
https://doi.org/10.1155/2021/9963829

Research Article

Can Multipath TCP Be Robust to Cyber Attacks? A Measuring Study of MPTCP with Active Queue Management Algorithms

Yuanlong Cao, Ruiwen Ji, Lejun Ji, Mengshuang Bao, Lei Tao, and Wei Yang

School of Software, Jiangxi Normal University, Nanchang 330022, China

Correspondence should be addressed to Yuanlong Cao; ylcao@jxnu.edu.cn

Received 11 March 2021; Accepted 19 May 2021; Published 28 May 2021

Academic Editor: Zhe-Li Liu

With the development of social networks, more and more mobile social network devices have multiple interfaces. Multipath TCP (MPTCP), as an emerging transmission protocol, can fit multiple link bandwidths to improve data transmission performance and improve user experience quality. At the same time, due to the large-scale deployment and application of emerging technologies such as the Internet of Things and cloud computing, cyber attacks against MPTCP have gradually increased. More and more network security research studies point out that low-rate distributed denial of service (LDDoS) attacks are relatively popular and difficult to detect and are recognized as one of the most severe threats to network services. This article introduces six classic queue management algorithms: DropTail, RED, FRED, REM, BLUE, and FQ. In a multihomed network environment, we perform the performance evaluation of MPTCP under LDDoS attacks in terms of throughput, delay, and packet loss rate when using the six algorithms, respectively, by simulations. The results show that in an MPTCP-enabled multihomed network, different queue management algorithms have different throughput, delay, and packet loss rate performance when subjected to LDDoS attacks. Considering these three performance indicators comprehensively, the FRED algorithm has better performance. By adopting an effective active queue management (AQM) algorithm, the MPTCP transmission system can enhance its robustness capability, thus improving transmission performance. We suggest that when designing and improving the queue management algorithm, the antiattack performance of the algorithm should be considered: (1) it can adjust the traffic speed by optimizing the congestion control mechanism; (2) the fairness of different types of data streams sharing bandwidth is taken into consideration; and (3) it has the ability to adjust the parameters of the queue management algorithm in a timely and accurate manner.

1. Introduction

With the development of social networks and the large-scale application of multiple wireless access technologies (i.e., Bluetooth, Wi-Fi, GPRS, and 4G), more and more mobile social network terminal devices are equipped with multiple network interfaces of different standards. Multihomed terminals can access multiple networks at the same time and achieve multipath data transmission by fitting the bandwidth of multiple links so as to improve data transmission performance, maximize network resource utilization, and improve user experience quality [1] while the multipath TCP (MPTCP) protocol [2] can distribute the data across multiple end-to-end transmission paths, by enabling the use of several network interfaces of devices simultaneously, as shown in Figure 1. As a variant of the TCP technology, MPTCP preserves the standard socket application programming interfaces (APIs) that are used by most Internet applications. Hosts can establish an MPTCP connection by using the existing socket APIs, without requiring any modification or addition to the applications and still being compatible on today's Internet. Therefore, MPTCP protocol is considered to play a huge role in the application of future Internet data transmission services [3, 4]. In spite of MPTCP has some advantages when applied to concurrent transmission in the network, in the real environment, the commonly happening network attacks have led to frequent changes in network quality which makes the network connection has a negative impact on the performance of MPTCP.

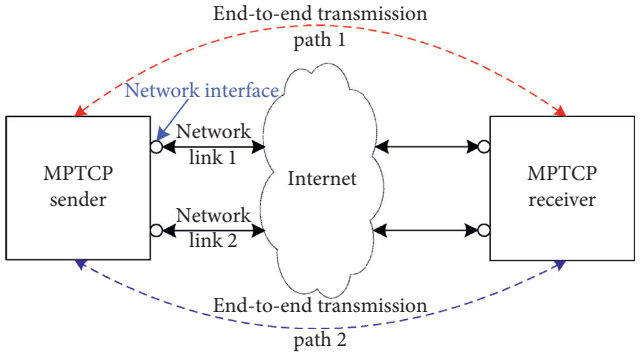

Figure 1: A basic MPTCP usage.

The MPTCP-based multipath data transmission is a process with complex network behavior. Although each transmission path can independently perform data transmission tasks according to its own network conditions, yet when a certain transmission path in the MPTCP multipath transmission system is attacked by a network, the reduction of the path transmission performance or the path failure will affect the transmission performance of other paths so as to produce adverse effects on the overall performance of multipath transmission such as robustness degradation [5]. Due to the large-scale deployment and application of emerging information technologies such as the Internet of Things (IoT) and cloud computing, a growing trend of various network attacks is presented, especially low-rate distributed denial of service (LDDoS) attacks. It has been pointed out by more and more network security research studies that LDDoS attacks are more prevalent and difficult to detect in the network, and it is recognized as one of the most major threats for network services [6, 7].

Taking advantage of the loopholes in the TCP protocol's retransmission timeout (RTO) mechanism, LDDoS sends periodically small pulse traffic separately by controlling multiple puppet machines, so it can reach the victim at the same time and converge into a huge impact traffic, causing the target host resources (such as bandwidth, memory, and CPU) to be exhausted and making the victim lose its ability to respond to the user's reasonable service request for a long-time [8]. Because of the characteristics of small traffic and concealment of LDDoS, when the MPTCP multipath transmission system suffers an LDDoS attack, the attack flow is difficult to be detected so that the defense capabilities of the transmission system will be affected and the robustness of the multi-path transmission system greatly reduced. Therefore, when the MPTCP transmission system is attacked by an LDDoS network, it is extremely important to improve the attack and defense capability of the system network.

The current academic research is mainly on improving the ability of detecting and defending LDDoS attacks from the perspective of extracting the characteristics of the attack flow. In recent years, it has been found by some scholars [9] that networks with different queue management algorithms have different defense capabilities when they are attacked by LDDoS. Queue management algorithms belong to link-based congestion control algorithms, which can be divided into passive queue management (PQM) and active queue management (AQM). By actively discarding or marking some data packets on the router side, network congestion can be effectively avoided and the performance of the TCP protocol can be improved by the queue management algorithm [10].

By tracking the research trends of MPTCP in academic circles, we find that there are insufficient research studies on the survivability analysis and robustness optimization of the MPTCP multipath transmission system. This paper introduces six network queue management algorithms: DropTail, random early detection (RED), fair random early detection (FRED), random exponential marking (REM), BLUE, and fair queuing (FQ). We compare and analyze their performance through simulation experiments. The conclusion of this paper provides a method for enhancing the robustness of the MPTCP network and provides effective suggestions for improving the queue management algorithm. Our research work in this paper is the first time to focus on the robustness of MPTCP from the perspective of AQM, hoping to attract the attention of relevant scholars through our research and promote the research on the adaptability and robustness of MPTCP congestion control.

The rest of this paper is organized as follows. The second part introduces the related research of LDDoS attack detection and defense and queue management mechanism. The third part will briefly introduce the basic ideas, advantages, and disadvantages of the six queue management algorithms. In the fourth part, the simulation experiment design and the evaluation of three performance indexes will be carried on. In the last part, we will give a summary.

2. Related Work

In recent years, various network attacks have shown a significant growth trend due to the large-scale deployment and application of emerging information technologies such as the IoT and cloud computing, especially LDDoS attacks [11]. It uses the weakness of the TCP protocol congestion control mechanism to periodically launch malicious attack traffic at a low rate, thereby reducing network throughput. Compared with traditional DDoS, LDDoS has a low attack rate and strong concealment. It is difficult to be discovered by traditional DDoS attack defense systems. LDDoS attacks can last longer, resulting in a rapid decline of network quality [12]. LDDoS is showing more and more serious harm to the network environment and has become a hot spot in the field of network security research at home and abroad. At present, many research institutions have carried out research on LDDoS attacks. In the network based on TCP protocol, the detection and defense of LDDoS have achieved a series of results.

For the research on LDDoS attacks, Kuzmanovic and Knightly [13] first proposed the concept of "Shrew" attacks. They collected relevant data about LDDoS attacks on the backbone network and conducted related research. After that, many researchers have proposed solutions for the detection and defense of LDDoS attacks by extracting LDDoS traffic characteristics or combining machine

learning methods. Sahoo et al. [14] proposed a measurement method based on generalized entropy. According to the characteristics of Software-Defined Network (SDN) data flow, they used information distance to quantify the deviation of traffic under different probability distributions as a metric to detect attack behavior. Ren et al. [15] proposed a smart NCAP that supports LDDoS detection and proposed a method to detect LDDoS attack traffic using a linear multiple regression model with Simple Network Management Protocol (SNMP) content. Agrawal and Tapaswi [16] proposed a power spectral density (PSD)-based method to detect and mitigate LDDoS attacks in the frequency domain. This method can monitor and analyze real-time aggregated traffic for attack detection. Gu et al. [17] proposed a semisupervised clustering detection method with multiple characteristics, which can effectively detect DDoS attacks from massive data streams. Li et al. [18] proposed a DDoS detection model and defense system in a Software-Defined Network environment based on deep learning. The model can learn patterns from network traffic sequences, track network attack activities in a historical manner, and effectively clear DDoS attack traffic. de Lima Filho et al. [19] proposed a DoS detection system based on machine learning and inferred from signatures extracted from network traffic samples. Han et al. [20] proposed a cross-plane DDoS attack defense framework and detection mechanism in a Software Defined Network. The mechanism consists of a coarse-grained flow monitoring algorithm on the data plane and an attack classification algorithm based on fine-grained machine learning on the control plane. Kavitha and Padmavathi [21] developed an effective defense technology against LDoS attacks and proposed an advanced random time queue blocking (ARTQB) scheme. Lin et al. [22] proposed a "double check priority queue" structure, which can effectively reduce the impact of DDoS attacks so that ordinary users can still access the service. Yue et al. [23] found that the random early detection (RED) algorithm is vulnerable to LDoS attacks, which limits the sending rate of TCP senders. Wei et al. [9] analyzed and compared the defense capabilities of three classic queue management algorithms in ad hoc networks attacked by DDoS.

For the detection and defense of network attacks such as LDDoS, different scholars have conducted research from different perspectives. We find that there are few research results that combine queue management algorithms with LDDoS defense. Queue management algorithm is one of the research hotspots in the field of network congestion control. By actively discarding or marking some data packets on the router side, the queue management algorithm can effectively avoid network congestion and improve the performance of the TCP protocol. Researchers have proposed many improved techniques for queue management algorithms. Karmeshu and Bhatnagar [24] proposed an adaptive queue management mechanism with a random drop algorithm. Compared with the existing active queue management algorithm, it significantly improves system performance in terms of throughput, average queue size, utilization, and queue delay. Bisoy and Pattnaik [25] proposed a rate and queue-based active queue management (RQ-AQM) algorithm to improve the stability of network systems supporting TCP streams. Although these algorithms can be applied to various network conditions, most of the existing queue management algorithms are designed without considering their antiattack performance.

Based on the previous research on LDDoS defense methods and queue management algorithms, we introduced six classic queue management algorithms—DropTail, RED, Fred, REM, BLUE, and FQ. And we compare the performance of throughput, delay, and packet loss rate when MPTCP networks are under LDDoS attacks. This paper provides a solution for enhancing the defense capability of the MPTCP transmission system against LDDoS and other network attacks and enriches the theoretical results of the antiattack research of queue management algorithms.

3. Queue Management Algorithm

Currently, the phenomenon of network congestion can be seen everywhere, and the most effective way to solve network congestion is to manage the queues in the network. The queue management mechanism is mainly to control the network transmission node to buffer the transmission of information in the form of a queue. When the queue length reaches a critical value, the corresponding service message is discarded to achieve the purpose of controlling the queue length. Therefore, routers should manage the queues and maintain a small queue length, resulting in a series of queue management algorithms [26]. Current queue management algorithms can be divided into passive queue management algorithms and active queue management algorithms. The idea of PQM is that the queue management module only takes corresponding measures when the queue buffer overflows. The most commonly used in routers is the passive queue management algorithm DropTail. The idea of AQM is to make early judgments and take a series of measures before the queue buffer overflows so as to avoid the occurrence of congestion as soon as possible [27–29]. According to the design principle, the existing AQM algorithms can be divided into three types: queue length-based, network load-based, and both queue length and network load-based. This paper selects five AQM algorithms (RED, FRED, BLUE, REM, and FQ) belonging to different design principles. In addition, these AQM algorithms have been extensively studied by the academic community, and the experimental results have certain representativeness and reference value.

3.1. DropTail Algorithm. The DropTail algorithm is a typical passive queue management algorithm and the most widely used queue management algorithm in the network. The basic idea of the DropTail queue management algorithm is when a data message arrives at a network node, it needs to be queued in different output port buffers. Regardless of the length of its own queue, it will put the data message in the queue and wait to be sent. However, when the data flow is large, the queue length has exceeded the set buffer capacity value, and the network node has no space to temporarily store these new data messages [30]. Therefore, when the network is

congested, all newly arrived data packets that are too late to be processed will be stored in the buffer, and these saved data packets will be processed when the system is idle. When the network continues to be congested, the buffer will be filled, and all newly arrived end packets will be discarded. When the sender detects that a data packet is discarded, it will reduce the data transmission rate until the congestion is eliminated.

The advantage of DropTail lies in its simple algorithm. Due to the simple way of processing data messages, it is supported by almost all network node platforms. However, the DropTail algorithm cannot avoid the occurrence of network congestion in advance, so there may be problems such as "global synchronization of TCP flows," continuous full queue status, and deadlock of service flows to the buffer, which affects the overall transmission speed and reduces network efficiency.

3.2. RED Algorithm. The RED algorithm [31] is a typical active queue management algorithm. The basic idea of the RED queue management algorithm is to judge congestion by monitoring the average length of the output port queue of the router. When the average length of the queue reaches a certain threshold, the router will randomly select some newly arrived packets to discard or mark and send a congestion notification. This algorithm can ensure that the sending window is reduced before the queue overflow causes packet loss, thereby reducing the sending rate and alleviating network congestion.

The RED queue management algorithm has two important computer mechanisms [32]. One is to calculate the average queue length and to predict congestion in advance by monitoring the average length of the buffer queue. The other is to calculate the drop probability P of data packets. If the average queue length is within the set threshold range, the arriving packets are discarded according to probability P.

In order to avoid unnecessary congestion control caused by sudden data, the RED queue management algorithm calculates the average queue length using an exponential weighted moving average algorithm; the formula is

$$L_{\text{now}} = (1 - w) \times L_{\text{first}} + w \times L, \tag{1}$$

where L is the current queue length, L_{first} is the previous estimated value of the average queue length, L_{now} is the current average queue length, w is the weighted coefficient of the current queue length, and its value range is between [0, 1].

At the same time, RED queue management mechanism needs to set two thresholds for the average queue length, which are the minimum threshold $L_{\min}$ and the maximum threshold $L_{\max}$. Comparing L_{now} with the threshold $L_{\min}$ and $L_{\max}$, the following operation is performed:

(1) If $L_{\text{now}} < L_{\min}$, all data packets are allowed to enter the queue

(2) If $L_{\text{now}} > L_{\max}$, all data packets are discarded

(3) If $L_{\min} \le L_{\text{now}} \le L_{\max}$, then calculate the transition packet loss probability p_a and discard the arriving data packets according to the probability P

In the RED queue management algorithm, the probability P of packet dropping by grouping is calculated according to the following formula:

$$P_a = P_{\max} \times \frac{(L_{\text{now}} - L_{\min})}{(L_{\max} - L_{\min})}. \tag{2}$$

$P_{\max}$ is the maximum packet loss rate. Obviously, the relationship between P_a, $P_{\max}$, $L_{\min}$, $L_{\max}$, and L_{now} is shown in Figure 2.

Then, the final packet loss probability is $P = (Pa/(1 - \text{count} \times Pa))$, where count is the number of packets accepted since the last packet was dropped.

Compared with the DropTail algorithm, the RED algorithm controls the flow rate through its own congestion control, thereby avoiding the problems of long delay time and low link utilization caused by the long-time full queue state of the data receiving node. However, the RED algorithm has parameter sensitivity problems and cannot effectively control the cache size. In addition, the RED algorithm fails to consider different types of data streams on the network when calculating the packet loss probability, so it cannot effectively handle the congestion notification connections of different data streams, resulting in various connections sharing bandwidth unfairly and affecting network performance [33].

3.3. FRED Algorithm. The FRED algorithm belongs to the active queue management algorithm and is a new algorithm based on the improvement of the fairness of the RED queue management algorithm [34]. The FRED queue management algorithm uses accounting for each active flow to make different marking packet decisions for flows that use different bandwidths, thereby improving the fairness of different flows sharing bandwidth. The FRED algorithm is mainly based on the basic framework of the RED algorithm, so the algorithm also has two main parts: calculating the average queue length and calculating the probability of dropped packets, and the calculation formula is consistent with the RED algorithm. However, there is a big difference from the RED algorithm. It needs to recalculate the average queue length in the buffer when the packet arrives and leaves.

Compared with the RED algorithm, the FRED algorithm has more advantages in terms of fairness, and it effectively discriminates and restricts the nonadaptive data flow. However, because the FRED algorithm needs to record the active flows in the entire cache queue and maintain its corresponding flow state, when the number of flows is large, the router will be overloaded and computational overhead will increase.

3.4. REM Algorithm. The REM algorithm is one of the active queue management algorithms, which uses link prices to represent the network congestion metric. The basic idea of the REM queue management algorithm is to use the price concept to detect and control the congestion state of the network. The REM algorithm uses the cumulative sum of the

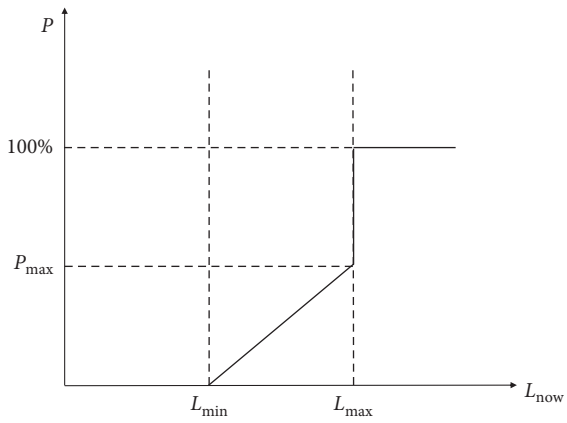

Figure 2: Average queue length and data packet drop probability of the RED algorithm.

price values of all connections on a channel as the congestion measure of this channel. And it embeds the metric value into the end-to-end packet marking probability that can be detected by the source so that the packet arrival rate matches the link bandwidth. Since calculating the data packet arrival rate needs to save certain state information, in order to avoid calculating the data packet arrival rate, the rate difference is approximated by the queue difference [35].

Price $P_l(t)$ of link l is calculated as follows:

$$P_l(t+1) = \left[P_l(t) + \gamma\left(b_l(t+1) - (1-\alpha_l)b_l(t) - \alpha_l b^*\right)\right]^+. \tag{3}$$

Among them, $\gamma > 0$, $\alpha_1 > 0$, $[z]^+ = \max\{0, z\}$, $b_l(t)$ is the instantaneous queue length of the queue of link l at time t, and b^* is the target queue length.

The marking probability of the queue of link l at time t is $m_l(t) = 1 - \phi^{-P_l(t)}$. ϕ is constant, and $\phi > 1$, and the end-to-end marking probability of the message is $1 - \phi^{-\Sigma_l P_l(t)}$. In practice, $\gamma = 0.001$, $\phi = 1.001$, $\alpha_1 = 0.1$, and $b^* = 20$.

The REM algorithm has opened up a new field for flow control, which can achieve the technical goal of AQM, but its performance is not ideal at present.

3.5. BLUE Algorithm. The BLUE algorithm is an active queue management algorithm based on network load. It uses packet loss events and links idle events to manage and notify congestion. The basic idea of the BLUE queue management algorithm is when the queue in the router overflows, the data packets will be continuously discarded. At this time, the BLUE algorithm will increase the probability of discarding the data packets and adjust the sending speed of the data packets. On the contrary, if the link is relatively idle at this time and the queue is empty, then the drop probability is reduced to increase the speed of sending data packets, thereby effectively controlling the speed of sending congestion notification information to improve the performance of the network [36].

The biggest advantage of the BLUE queue management algorithm is that a relatively small buffer can be used to complete congestion control, which improves the throughput of TCP streams and allows routers to have more free space. However, the BLUE algorithm also has a parameter setting problem. When the RTT of a data packet changes greatly or the number of connections suddenly changes, the set parameters will be invalid and the queue will fluctuate between packet loss and low usage.

3.6. FQ Algorithm. The FQ algorithm is an active management algorithm based on fair queues. The FQ algorithm establishes an independent output queue for each connection in the router. The router processes each queue in a round-robin manner to ensure fairness between each data flow. When a line is idle, the router scans all queues in turn and sends out the first packet of the queue each time. When a flow's data packets arrive too fast, its queue will quickly fill up, and new data packets belonging to this flow will be discarded [37].

With the FQ algorithm, it is impossible for each data stream to sacrifice other data streams and occupy more resources. In addition, it separates data streams so that data streams that do not comply with the congestion control mechanism will not affect other streams. So, it provides fairness without sacrificing statistical reuse.

4. Performance Evaluation

4.1. Experimental Design. In order to study the performance of six queue management algorithms when an MPTCP-enabled multihomed network suffers from LDDoS attacks, we develop a basic double dumbbell simulation topology with reasonable LDDoS attack traffic in NS-2 [38], as shown in Figure 3.

The router $R_{1,1}$ on path A is connected to five edge nodes that send UDP attack flows, and router $R_{1,2}$ is connected to five edge nodes that receive attack flows. We set the bandwidth between the node sending and receiving the attack stream and its connected router to 50 Mb and the propagation delay to 25 ms. The bandwidth between $R_{1,1}$ and $R_{1,2}$ and between $R_{2,1}$ and $R_{2,2}$ is set to 5 Mb, the propagation delay is 25 ms, and the queue management algorithm uses the DropTail algorithm. The total simulation time is 60 seconds. LDDoS attacks usually use the UDP protocol with constant bit rate (CBR) traffic to take a lot of bandwidth, so all attackers will generate UDP/CBR packets and start the attack after 2 s. The following three parameter values are used to describe the characteristics of LDDoS attacks [39]:

$$\text{LDDoS}(T, L, R) = \text{LDDoS}(100\text{ms}, 100\text{ms}, 1\text{Mbps}). \tag{4}$$

Among them, T is the attack period, L is the duration of the attack (the width of the attack pulse), and R is the strength of the attack pulse (the attack rate). If the parameter values of T, L, and R are set reasonably, the LDDoS traffic can reject the bandwidth of the regular TCP stream and avoid being detected by the DoS defense system. When the congestion control mechanism is triggered, the data packet will enter the timeout retransmission state. When an LDDoS attack occurs, the higher the R, the greater the bandwidth loss caused.

We choose the most commonly used DropTail algorithm in the simulation experiment to set the parameters of the best

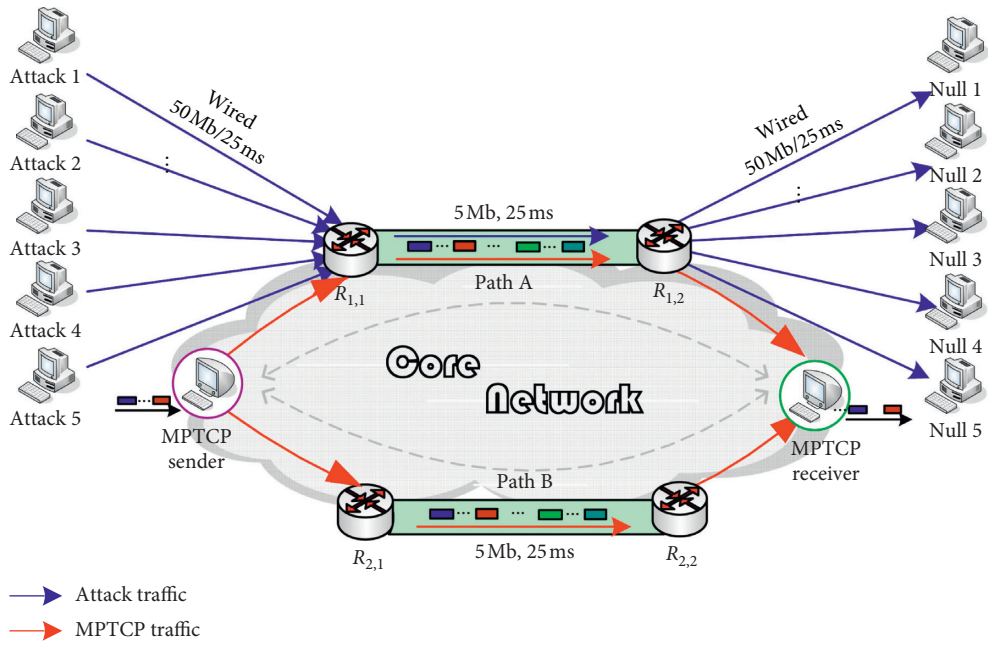

Figure 3: A basic dual-dumbbell simulation topology with LDDoS attacks.

attack flow and use the parameters of LDDoS as the fixed values in the comparison experiment. Figure 4 shows the congestion window (cwnd) size of path *A* when LDDoS is attacked and when it is not attacked. When an LDDoS attack is launched (after 2 s), the cwnd size of path *A* drops sharply. This is because the LDDoS attack can use MPTCP's RTO mechanism to make the MPTCP sender stay in the timeout retransmission state on path *A* and cannot exit. The congestion control mechanism of MPTCP is similar to that of TCP. In order to test the congestion of the network, the cwnd size is set to 1 at the initial slow start. As long as the sender judges that the network is congested, it is necessary to set the slow start threshold to half of the sender window value when congestion occurs (but not less than 2), then reset cwnd size to 1, and return to slow start stage. After the 20 s of simulation time in this experiment, the size of cwnd is maintained at 1. The network attack keeps the TCP data packet on path A in the timeout retransmission state, which shows that the LDDoS attack has achieved the best attack effect. This confirms that the DropTail algorithm, which is analyzed later, has a throughput of 0 for the normal TCP data flow on path *A* after the 20 s.

4.2. Simulation Analysis. Based on the MPTCP network, this paper analyzes and compares the defense capabilities of six classic network queue management algorithms such as DropTail, RED, FRED, REM, BLUE, and FQ when they are attacked by LDDoS. During the simulation, we test and analyze the performance of the throughput, end-to-end delay, and packet loss rate.

4.2.1. Comparison of the Throughput Performance. Throughput is the amount of successfully transmitted data per unit time. We measure the throughput between the sender and receiver when the MPTCP-enabled multihomed

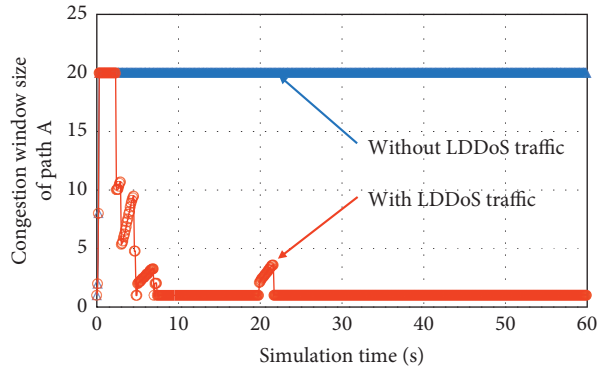

Figure 4: The congestion window size of path *A* with or without an LDDoS attack, respectively.

network is attacked by LDDoS. Figure 5 shows the comparison of the throughput performance of the MPTCP network transmission system using six queue management algorithms in a 60 s simulation time. For the convenience of observation, we plot the comparison of throughput performance when the LDDoS attack reaches stability, that is, after 20 s. Figure 6 tests the throughput of the MPTCP network transmission system (including path *A* and path *B*). Figure 7 tests the throughput of path *A* attacked by LDDoS.

We can see that when path *A* is attacked by LDDoS, using the DropTail algorithm and the BLUE algorithm will lose the data transmission capability of the normal TCP stream after the 20 s. In addition, regardless of the single path or the entire transmission system, the FRED algorithm has the best throughput performance. This is because RED predicts congestion in advance by monitoring changes in the average length of the buffer queue so that data transmission nodes can control traffic speed through their own congestion control, thus avoiding low link utilization due to long periods of full queues.

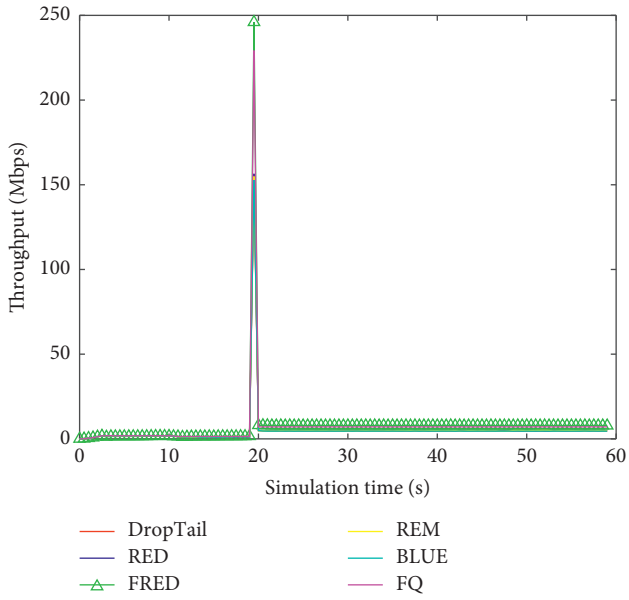

Figure 5: Comparison of the throughput performance of different queue management algorithms.

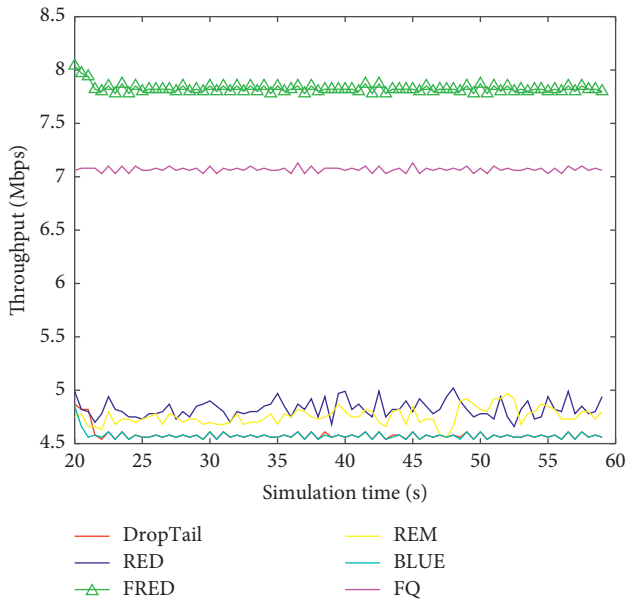

Figure 6: Comparison of the throughput performance of different queue management algorithms after 20 s.

In the MPTCP-enable multihomed networks, different queue management algorithms have different throughput performances when subjected to LDDoS attacks. We find that if the queue management algorithm cannot distinguish between different types of data streams, it is very likely that bad-behaving data streams will occupy a large number of data streams. From this, it can be seen that the queue management algorithm can improve the defense ability in the case of network attacks when considering the fairness of different types of data streams sharing bandwidth, which shows higher throughput performance. However, the RED

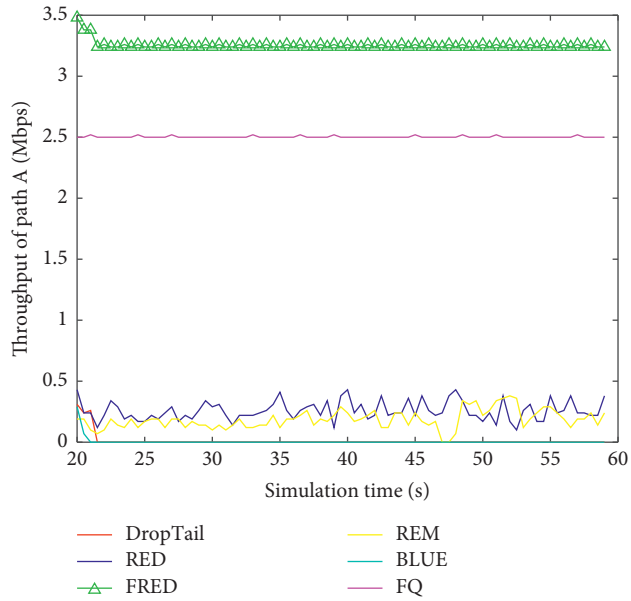

Figure 7: Comparison of the throughput performance of different queue management algorithms on path A after 20 s.

algorithm fails to consider that the data streams on the network are of different types when calculating the packet loss probability, which leads to unfair sharing of bandwidth among various connections and affects network performance. The FRED algorithm improves the fairness of the RED algorithm. It makes different marking packet decisions by accounting for each active stream, thereby improving the fairness of different streams sharing bandwidth. When data streams with different competition capabilities compete for limited bandwidth, fairness ensures that the throughput performance of less competitive data streams will not suffer great damage, but a part of the transmission capacity is maintained. In addition, the FQ algorithm is slightly worse than FRED in throughput performance, but compared with the other four algorithms, because the fairness between different flows is also considered, when the network is under LDDoS attacks, it also retains a part of the throughput performance. The DropTail algorithm, as a typical passive queue management algorithm, cannot avoid network congestion in advance. When a sudden attack flow is encountered, the queue of the router will always be in a full state, a large number of TCP data flows will jointly slow down the sending rate to reduce congestion, and the utilization rate of the network will decrease accordingly. The REM algorithm and the BLUE algorithm also do not consider the fairness of different data streams sharing bandwidth, so when the number of router connections suddenly changes drastically, it will lead to poor throughput.

In the MPTCP-enabled multihomed networks, different queue management algorithms have different throughput performance when subjected to LDDoS attacks. If the queue management algorithm cannot distinguish between different types of data streams, it is very likely that bad-behaving data streams will occupy a lot of bandwidth. It can be seen that the queue management algorithm, when considering the

fairness of different types of data streams sharing bandwidth, can improve the defense capability in the event of network attacks, which is manifested in higher throughput performance.

4.2.2. Comparison of the End-to-End Delay Performance. The end-to-end delay is the time required for a message or packet to be transmitted from one end of a network to another. It includes transmission delay, propagation delay, processing delay, and queuing delay. We test the delay of TCP data flow on path *A*. Figure 8 shows the comparison of the delay performance of the six queue management algorithms when the network is under LDDoS attacks. We mainly observe and analyze the comparison of delay performance when the network attack reaches a stable state (20 s).

From Figure 8, the delay performance of the RED algorithm is the best and the FRED performance is second. The RED algorithm can control the flow rate through its own congestion control, thereby avoiding the long delay time caused by the data receiving node due to the full queue state for a long time. The FRED algorithm considers the fairness issue more than the RED algorithm. It needs to record the active flow in the entire cache queue and maintain its corresponding flow state, which causes the router to be overloaded and increases the computational overhead. Therefore, in terms of delay performance, the FRED algorithm is slightly worse than the RED algorithm. The DropTail algorithm only sends a congestion signal to the router or the sender when the queue is full, resulting in prolonged queuing time of data packets in the queue and increased end-to-end delay. In addition, from Figure 8, we find that the time delay data of the DropTail algorithm disappear after 25 s. This is because the TCP stream on path *A* has been severely attacked by LDDoS and has been in a timeout retransmission state, so the delay of this type of data stream cannot be calculated. Although the BLUE algorithm adjusts the sending speed of data packets when the queue overflows in the router, when the router is attacked by a network such as LDDoS, the set parameters will become invalid. The delay of the REM algorithm shows obvious fluctuations because the algorithm's operating mechanism is to match the data packet arrival rate with the link bandwidth. When subjected to periodic LDDoS attacks, the data packet transmission rate will also change accordingly. We find that the delay of the FQ algorithm is in the middle of the delay of these six algorithms and presents a horizontal straight line. This is consistent with its design philosophy of ensuring fairness between each flow and allowing routers to process each queue in a polling manner.

In the MPTCP-enabled multihomed network, different queue management algorithms have different delay performances when subjected to LDDoS attacks. The queue management algorithm can adjust the flow rate by optimizing the congestion control mechanism and avoiding the long delay time caused by the long-time full queue state of the data receiving node, thereby improving the defense ability in the case of network attacks.

4.2.3. Comparison of the Packet Loss Rate Performance. The packet loss rate refers to the ratio of the number of data packets lost in the test to the data group sent. We test the packet loss of TCP data flow on path *A*. Table 1 shows the detailed data of the total number of packets, the number of lost packets, and the packet loss rate of the six queue management algorithms when the network is under LDDoS attacks. It can be seen from Figure 9 that the FQ algorithm has no packet loss during the entire data transmission process. The packet loss rates of the FRED, DropTail, BLUE, and REM algorithms are 0.02%, 2.68%, 3.15%, and 4.22%, respectively. The packet loss rate of RED is the highest, with a packet loss rate of 5.23%.

In a network environment, it is entirely possible that an application does not use the TCP protocol. The LDDoS attack flow can bypass the end-to-end congestion control mechanism and send its own data packets to the router arbitrarily, causing normal application data packets to be discarded. The FQ algorithm solves this problem. In the FQ algorithm, the router has a queue for each output line. The router processes packets in a "polling" manner to ensure fairness between each flow. Therefore, the packet loss rate using the FQ algorithm is low. However, when data packets of a flow arrive too fast, its queue will quickly fill up, and new data packets belonging to this flow will also be discarded. Although the RED algorithm proposes a method to deal with sudden data flow, it uses an exponentially weighted moving average algorithm to make the average queue length change relatively slowly, but because the algorithm has the disadvantage of parameter sensitivity, the parameters (such as the maximum threshold $L_{\max}$) cannot be modified in time, resulting in a large number of packets being discarded. Compared with the RED algorithm, the FRED algorithm recalculates the average queue length in the buffer when the packet arrives and leaves. It can more timely and accurately reflect the queue changes and modify the parameters, so the packet loss rate is very low. We find that when DropTail, BLUE, and REM algorithms are attacked by LDDoS and other network attacks, more data packets will be discarded by the queue, which reduces the efficiency of the network.

It can be seen that when it is subjected to network attacks such as LDDoS, the queue management algorithm should have the ability to adjust parameters in a timely and accurate manner so as to effectively ensure the transmission of normal TCP data streams. In addition, improving the fairness of the algorithm can also reduce the packet loss rate and show better transmission performance.

The results show that in the MPTCP-enabled multihomed networks, different queue management algorithms have different throughput, delay, and packet loss rate performance when subjected to LDDoS attacks. In terms of throughput performance, considering fairness, the FRED algorithm has the best performance and the FQ algorithm has the second-highest performance. In view of delay performance, the RED algorithm is the best, and the performance of FRED is slightly worse than that of RED. However, with the development of technology, the small delay gap can

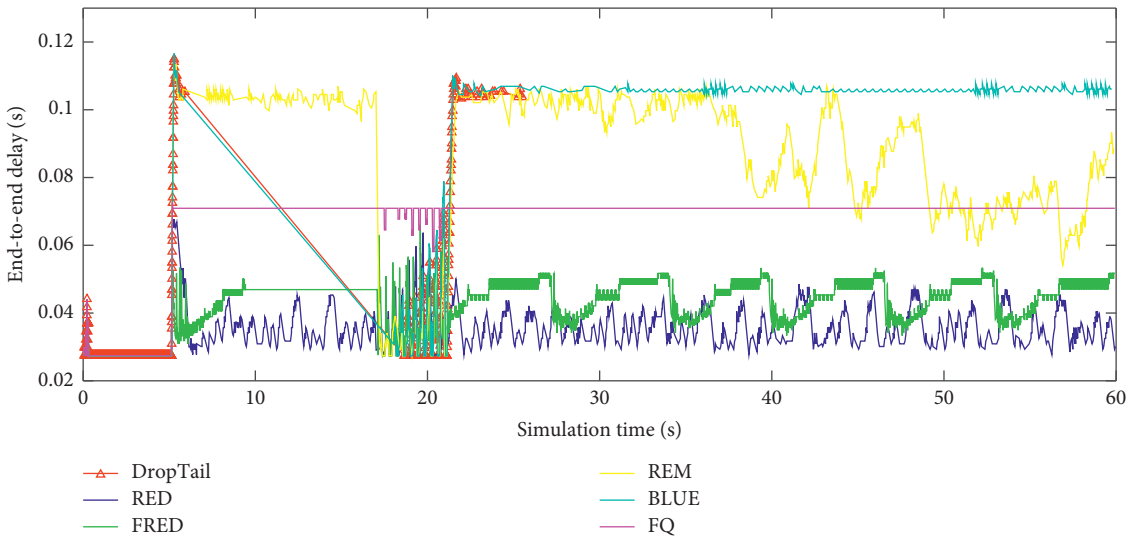

Figure 8: Comparison of the end-to-end delay performance of different queue management algorithms.

Table 1: Packet loss data of different queue management algorithms.

Queue management algorithm	Total number of packages	Number of lost packets	Packet loss rate (%)
DropTail	1046	28	2.68
RED	2014	110	5.23
FRED	16012	4	0.02
BLUE	1112	35	3.15
REM	1894	80	4.22
FQ	12798	0	0.00

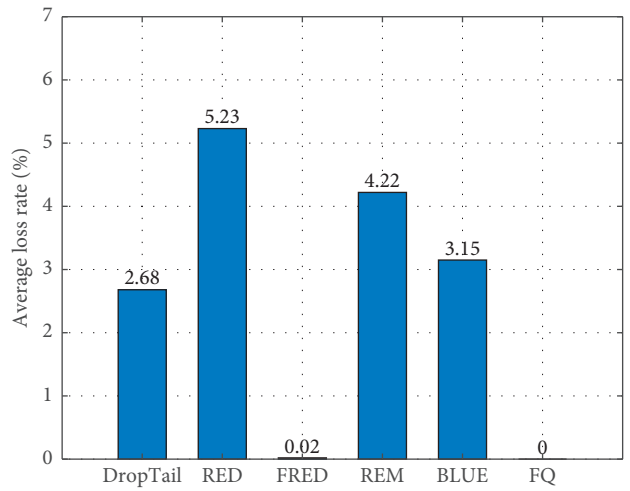

Figure 9: Comparison of the packet loss rate performance of different queue management algorithms.

be made up by increasing the operating speed of hardware devices. In consideration of packet loss rate performance, the FQ and FRED algorithms can maintain a lower packet loss rate when subjected to network attacks of LDDoS compared with other algorithms. Through an overall consideration of the three performance indicators of throughput, delay, and packet loss rate, it is evident that the FRED algorithm has better performance.

5. Conclusion

This paper introduces six queue management algorithms: DropTail, RED, FRED, REM, BLUE, and FQ. Through simulation experiments, we compare the performance of different queue management algorithms in the MPTCP network under LDDoS attack. The results show that in the multihost network using MPTCP, when one of the paths is attacked by LDDoS, the other paths can still transmit normally and the whole system will not collapse. Different queue management algorithms have different throughput, latency, and packet loss rates. Through an overall consideration of the three performance indicators of throughput, delay, and packet loss rate, it is evident that the FRED algorithm has better performance. By adopting an effective queue management algorithm, the MPTCP transmission system can enhance its robustness and defense capability, thus improving transmission performance. In addition, our research conclusions provide effective suggestions for the technical improvement of the queue management algorithm. In the future, the antiattack performance of the algorithm should be taken into consideration when designing and improving the queue management algorithm. An effective queue management algorithm should achieve three aspects: (i) it can adjust the traffic speed by optimizing the congestion control mechanism; (ii) the fairness of different types of data streams sharing bandwidth is taken into consideration; and (iii) it has the ability to adjust the

parameters of the queue management algorithm in a timely and accurate manner, thereby effectively ensuring the transmission performance of normal TCP data streams so as to improve the defense capability against network attacks.

Data Availability

No data were used to support this article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) under grant no. 61962026 and the Natural Science Foundation of Jiangxi Province under grant no. 20192ACBL21031.

References

[1] M. R. Palash and K. Chen, "MPWiFi: synergizing MPTCP based simultaneous multipath access and WiFi network performance," *IEEE Transactions on Mobile Computing*, vol. 19, no. 1, pp. 142–158, 2020.

[2] A. Ford, C. Raiciu, M. Handley, O. Bonaventure, and C. Paasch, "TCP extensions for multipath operation with multiple addresses," *Internet Engineering Task Force (IETF) RFC 8684*, 2020.

[3] Y. Cao, M. Collotta, S. Xu et al., "Towards adaptive multipath managing: a lightweight path management mechanism to aid multihomed mobile computing devices," *Applied Sciences-Basel*, vol. 10, no. 1, pp. 1–18, 2020.

[4] F. Song, Z. Ai, Y. Zhou, I. You, K.-K. R. Choo, and H. Zhang, "Smart collaborative automation for receive buffer control in multipath industrial networks," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 2, pp. 1385–1394, 2020.

[5] Y. Cao, J. Chen, Q. Liu, G. Lei, H. Wang, and I. You, "Can multipath TCP be robust to cyber attacks with incomplete information?" *IEEE Access*, vol. 8, pp. 165872–165883, 2020.

[6] X. Zhao, H. Peng, X. Li et al., "Defending application layer DDoS attacks via multidimensional parallelotope," *Security and Communication Networks*, vol. 2020, Article ID 6679304, 11 pages, 2020.

[7] L. Zhou, M. Liao, C. Yuan, and H. Zhang, "Low-rate DDoS attack detection using expectation of packet size," *Security and Communication Networks*, vol. 2017, Article ID 3691629, 14 pages, 2017.

[8] Z. Wu, Q. Xu, J. Wang et al., "Low-rate DDoS attack detection based on factorization machine in software defined network," *IEEE Access*, vol. 8, pp. 17404–17418, 2020.

[9] W. Wei, H. Song, H. Wang, and X. Fan, "Research and simulation of queue management algorithms in Ad Hoc networks under DDoS attack," *IEEE Access*, vol. 5, pp. 27810–27817, 2017.

[10] C. A. Gomez, X. Wang, and A. Shami, "Federated intelligence for active queue management in inter-domain congestion," *IEEE Access*, vol. 9, pp. 10674–10685, 2021.

[11] M. V. Kieu, D. T. Nguyen, and T. T. Nguyen, "A way to estimate TCP throughput under low-rate DDoS attacks: one TCP flow," in *Proceedings of the 2020 RIVF International Conference on Computing and Communication Technologies (RIVF)*, pp. 1–8, Ho Chi Minh, Vietnam, October 2020.

[12] Z. Li, H. Jin, D. Zou, and B. Yuan, "Exploring new opportunities to defeat low-rate DDoS attack in container-based cloud environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 3, pp. 695–706, 2020.

[13] A. Kuzmanovic and E. Knightly, "Low-rate TCP-targeted denial of service attacks (the shrew vs. the mice and elephant)," in *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM '03)*, vol. 33, no. 4, pp. 75–86, Karlsruhe, Germany, August 2003.

[14] K. S. Sahoo, D. Puthal, M. Tiwary, J. J. P. C. Rodrigues, B. Sahoo, and R. Dash, "An early detection of low rate DDoS attack to SDN based data center networks using information distance metrics," *Future Generation Computer Systems*, vol. 89, pp. 685–697, 2018.

[15] J. Ren, Y. Liu, J. Wu, J. Li, and K. Wang, "Smart NCAP supporting low-rate DDoS detection for IEEE 21451-1-5 internet of things," in *Proceedings of the 2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS)*, pp. 532–535, Taipei, Taiwan, May 2019.

[16] N. Agrawal and S. Tapaswi, "Low rate cloud DDoS attack defense method based on power spectral density analysis," *Information Processing Letters*, vol. 138, pp. 44–50, 2018.

[17] Y. Gu, Y. Wang, Z. Yang et al., "Multiple-features-based semi-supervised clustering DDoS detection method," *Mathematical Problems in Engineering*, vol. 2017, Article ID 5202836, 10 pages, 2017.

[18] C. Li, Y. Wu, X. Yuan et al., "Detection and defense of DDoS attack based on deep learning in OpenFlow-based SDN," *International Journal of Communication Systems*, vol. 31, no. 5, Article ID e3497, 2018.

[19] F. S. de Lima Filho, F. A. F. Silveira, A. D. M. Brito Jr. et al., "Smart detection: an online approach for DoS/DDoS attack detection using machine learning," *Security and Communication Networks*, vol. 2019, Article ID 1574749, 15 pages, 2019.

[20] B. Han, X. Yang, Z. Sun et al., "OverWatch: a cross-plane DDoS attack defense framework with collaborative intelligence in SDN," *Security and Communication Networks*, vol. 2018, Article ID 9649643, 15 pages, 2018.

[21] R. Kavitha and G. Padmavathi, "Advanced random time queue blocking for effective protection of application servers against low-rate DoS attacks," *International Journal of Network Security*, vol. 19, no. 6, pp. 1024–1035, 2017.

[22] C. Lin, H. Lin, T. Wu et al., "Preserving quality of service for normal users against DDoS attacks by using double check priority queues," *Journal of Ambient Intelligence and Humanized Computing*, vol. 4, no. 2, pp. 275–282, 2013.

[23] M. Yue, Z. Wu, and J. Wang, "Detecting LDoS attack bursts based on queue distribution," *IET Information Security*, vol. 13, no. 3, pp. 285–292, 2019.

[24] S. P. Karmeshu and S. Bhatnagar, "Adaptive mean queue size and its rate of change: queue management with random dropping," *Telecommunication Systems*, vol. 65, no. 2, pp. 281–295, 2017.

[25] S. K. Bisoy and P. K. Pattnaik, "RQ-AQM: a rate and queue-based active queue management using feedback control theory," *International Journal of Communication Networks and Distributed Systems (IJCNDS)*, vol. 21, no. 2, pp. 266–295, 2018.

[26] S. K. Mohapatra, S. K. Bisoy, and P. K. Dash, "Stability analysis of active queue management techniques," in *Proceedings of the*

2015 International Conference on Man and Machine Interfacing (MAMI), pp. 1–6, Bhubaneswar, India, December 2015.

[27] R. Al-Saadi, G. Armitage, J. But, and P. Branch, "A survey of delay-based and hybrid TCP congestion control algorithms," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3609–3638, 2019.

[28] Y. Liu, X. Liu, Y. Jing, Z. Zhang, and X. Chen, "Congestion tracking control for uncertain TCP/AQM network based on integral backstepping," *ISA Transactions*, vol. 89, pp. 131–138, 2019.

[29] M. Khatari, A. A. Zaidan, B. B. Zaidan, O. S. Albahri, and M. A. Alsalem, "Multi-criteria evaluation and benchmarking for active queue management methods: open issues, challenges and recommended pathway solutions," *International Journal of Information Technology & Decision Making*, vol. 18, no. 4, pp. 1187–1242, 2019.

[30] P. K. Sahu and B. M. Acharya, "Performance analysis of unicasting routing protocols for mobile ad-Hoc network," in *Proceedings of the 2019 International Conference on Applied Machine Learning (ICAML)*, pp. 286–290, Bhubaneswar, India, May 2019.

[31] A. Pandey, T. Anand, M. Shah, and M. P. Tahiliani, "Adaptive RED for FreeBSD: design, implementation and challenges," in *Proceedings of the 2019 IEEE Region 10 Conference (TENCON 2019)*, pp. 2340–2344, Kochi, India, Octorber 2019.

[32] S. Patel, "Performance analysis of RED for stabilized queue," in *Proceedings of the 2014 Seventh International Conference on Contemporary Computing (IC3)*, pp. 306–311, Noida, India, August 2014.

[33] S. Simaiya, A. Shrivastava, and N. P. Keer, "IRED algorithm for improvement in performance of mobile Ad Hoc networks," in *Proceedings of the 2014 Fourth International Conference on Communication Systems and Network Technologies*, pp. 283–287, Bhopal, India, April 2014.

[34] H. Wu, F. Ren, D. Mu, and W. Pan, "Utilizing TTL to enhance TCP fairness," in *Proceedings of the 2007 Second International Conference on Communications and Networking in China*, pp. 208–212, Shanghai, China, August 2007.

[35] S. Patel, "Performance analysis and modeling of congestion control algorithms based on active queue management," in *Proceedings of the 2013 International Conference on Signal Processing and Communication (ICSC)*, pp. 449–454, Noida, India, December 2013.

[36] Y. Irawan and N. Surantha, "Performance evaluation of queue algorithms for video-on-demand application," in *Proceedings of the 2020 International Conference on Information Management and Technology (ICIMTech)*, pp. 966–971, Bandung, Indonesia, August 2020.

[37] H. Attar, M. R. Khosravi, S. S. Igorovich et al., "Review and performance evaluation of FIFO, PQ, CQ, FQ, and WFQ algorithms in multimedia wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 16, no. 6, Article ID 15501477209, 2020.

[38] Y. Cao, F. Song, Q. Liu, M. Huang, H. Wang, and I. You, "A LDDoS-aware energy-efficient multipathing scheme for mobile cloud computing systems," *IEEE Access*, vol. 5, pp. 21862–21872, 2017.

[39] M. Baskar, J. Ramkumar, C. Karthikeyan et al., "Low rate DDoS mitigation using real-time multi threshold traffic monitoring system," *Journal of Ambient Intelligence and Humanized Computing*, vol. 2021, 2021.