

---

This item was submitted to [Loughborough's Research Repository](#) by the author.  
Items in Figshare are protected by copyright, with all rights reserved, unless otherwise indicated.

## Priority checking RED for improving QoS in IPv6

PLEASE CITE THE PUBLISHED VERSION

PUBLISHER

© IEEE

VERSION

VoR (Version of Record)

LICENCE

CC BY-NC-ND 4.0

REPOSITORY RECORD

Li, Yunqiu, and Shuang-Hua Yang. 2019. "Priority Checking RED for Improving Qos in Ipv6". figshare.  
<https://hdl.handle.net/2134/4146>.

This item was submitted to Loughborough's Institutional Repository (<https://dspace.lboro.ac.uk/>) by the author and is made available under the following Creative Commons Licence conditions.



For the full text of this licence, please go to:  
<http://creativecommons.org/licenses/by-nc-nd/2.5/>

# Priority Checking RED for Improving QoS in IPv6

Yunqiu LI, Shuang-Hua YANG

Computer Science Department, Loughborough University

**Abstract**— This paper presents a Priority Checking Random Early Detection (PC-RED) gateway for ensuring the Quality of Service (QoS) of high priority dataflow in IPv6 networks. A bit in the IP header is used in PC-RED to label the packet with the current status of the QoS that the dataflow is being treated in. The status of the QoS is determined by the difference between the packet average-dropping rate and the fixed desired limit dropping rate of the dataflow. PC-RED would perform dissimilarly to every dataflow corresponding to the different QoS status throughout congestions. PC-RED has been modeled and the parameter setting has been studied. Simulations of a TCP/IP network are used to illustrate how PC-RED affects the transfer of dataflow. The result shows remarkable contrast between the High-Priority and Non-Priority dataflow throughput under PC-RED mechanism.

**Index Terms**— Congestion Control, IPv6, RED, AQM, Priority

## 1. INTRODUCTION

As a result of the explosive growth of Internet data transmission, the Internet congestion becomes an obstacle to providing high-levelled Quality of Service (QoS) to certain end users. Research on Internet congestion control comprises two main parts: TCP (Transmission control protocol) and AQM (Active Queue Management) [1]. Many AQM algorithms have been proposed over the last decade, such as RED (Random Early Detection) [2], GREEN (Generalized Random Early Evasion Network) [3], and REM (Random Exponential Marking) [4]. Only RED has been widely implemented in major commercial routers, even though there is some doubt of RED implementation [5]. The implementation of RED could avoid global synchronization and provide better QoS to the end users. For many years, RED has been investigated [6] [7], RED with different algorithms has been designed, such as Adaptive RED [8], and there are also some methods to tune RED parameters [9] [10], to improve RED performance. However, the Internet nowadays is expected to provide distinct service to different users based on value, and to deliver distinct service based on content accessed, etc [11].

The growth of Internet also urges the Internet Engineering Task Force (IETF) to produce the upgraded Internet Protocol IPv6 to satisfy the large demands of IP addresses. IPv6 completely revamps IPv4, addressing performance, scalability, security, etc. The address space jumps from 32-bit format to 128-bit is just one of the aspects. There are several changes to the header format with IPv6 as well [12] [13]. The flow label field, which is a new field added in IPv6 header, is used to facilitate identification of data requiring special handling, such as those involved in real-time applications, etc.

Priority checking RED is developed to ensure the end user with high priority to get the level of QoS they expected by implementing a Priority Checking function in RED, in which

the extra flow label field is used to label the packets of different dataflow with their priority and the current level of services they have experienced in order to request from the network router with PC-RED distinct QoS. So the real-time system users or people who would like to pay more for better services can get the guaranteed high level of QoS according to their demands.

The rest of the paper is organized as follows. In Section 2, the algorithm of PC-RED is introduced. Section 3 focuses on the PC-RED parameter setting. Simulation results obtained by using Network Simulator 2 (NS2) that verify the PC-RED contribution to guaranteeing some form of QoS to particular end user with high priority is presented in Section 4. Finally, the conclusions are given in Section 5.

## 2. PRIORITY CHECKING RANDOM EARLY DETECTION

The IPv6 packets can be labelled with different priority, so dataflows with different QoS requirements could be classified and probably stored in separate buffers before they are transmitted into the further hop [14]. But such mechanism has disadvantages such as it complicates the implementation using separate buffers for different QoS level, and there is a high probability that the buffer utilization is inferior throughout the whole transmission. Thus, PC-RED overcomes the under-utilization of the queue by using one buffer for multi-dataflows, and a different mechanism to pick which packet to drop when there is congestion.

One bit in the IPv6 header of the packet, which is referred to as QOSA (Quality Of Service Alert), is used to indicate whether the dataflow has been treated worse than it was expected. During the transmission, all of the dataflows with priority have certain levels of packet average-dropping rate  $a_d$ .

If  $a_d$  exceeds the desired limit  $L$ , QOSA would be set to one to notify PC-RED to implement some form of protection.

In the PC-RED router, an extra database is set up referred to as Priority List. The priority list is a list containing the priority limit and the current status of every data flow passing through the router. The elements of the list are: Data flow ID, Priority limit, Last dropping time,  $a_d$  and QoS Status.

The working process of PC-RED is shown in Figure 1. Blocks in grey illustrate the differences compared with the traditional RED. The priority list maintenance function is triggered each time when the router receives a packet. When PC-RED adds a new entry to the priority list with the dataflow id as the index element,  $a_d$  would be set to half of  $L$  as default, and the status would be set to zero indicating the average dropping rate is currently lower than the desired limit. In case the dataflow is switched to one route due to the transmission failure in the other routes, a higher start point of the average drop rate could help to notify the router with the QoS level

more rapidly. If the flow id is in the list, PC-RED would check the status element in the entry. If the status is one, the QOSA bit would be set to one, indicating this packet is un-droppable, before adding it into the queue.

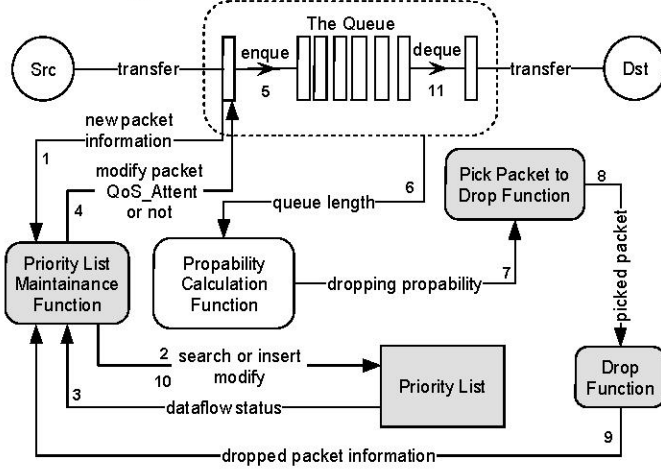


Figure 1 PC-RED Working Processes

All of the other function would be the same as the traditional RED except when picking a packet to drop while the average queue length has exceeded the threshold. Instead of randomly picking the victim, PC-RED only picks those packets with zero as the QOSA bit. And after dropping a packet, the priority list maintenance function is triggered again to update the priority list with the latest average dropping rate.

When the average queue length exceeds the threshold, PC-RED will first randomly pick a packet, and access the header, to check the QOSA bit. If the QOSA is not zero and it is not the last packet in the queue, it would check to the next packet. Otherwise, PC-RED will get the packet size, and calculate the latest average dropping rate  $a_d$ , then update  $a_d$  element with the new value, and compare  $a_d$  with  $L$ . If  $a_d$  has exceeded  $L$ , set the status element to one, otherwise, set the status to zero. Finally it will drop the packet and update the last dropping time element in the corresponding entry.

In RED, the dropping probability  $p(t)$  calculated from the average queue length  $q(t)$  is regarded as an overall dropping probability. However, when breaking down the dataflows into high-priority ones and non-priority ones,  $p(t)$  can be divided into  $p_p(t)$ , the dropping probability of high-priority dataflows, and  $p_n(t)$ , the dropping probability of the non-priority ones. Thus PC-RED can provide different levels of QoS to dataflows with different priorities by manipulating their dropping probabilities respectively.

### 3. PC-RED MODELLING

In [15], a dynamic model of TCP behaviour was developed using fluid-flow and stochastic differential equation analysis. Based on that, [16] has presented a simplified TCP-queue dynamic model.

For a network with TCP dataflows with different priorities, assume the total number of TCP sessions is  $N$ , including

$N_p$  TCP sessions with high-priority and  $N_n$  TCP sessions with non-priority. In  $N_p$  sessions,  $N_{p+}$  of them have the average-dropping rate  $a_d$  exceeding the desired limits  $L$  and  $N_{p-}$  of them have the average-dropping rate  $a_d$  under the desired limits  $L$ . Thus, we have

$$\begin{cases} N = N_p + N_n \\ N_p = N_{p+} + N_{p-} \end{cases} \quad (1)$$

Separating the perturbation of the packet dropping probability  $\delta p$  into three parts in terms of the dataflow priorities:  $\delta p_{p+}$ , the perturbation of the packet-dropping probability of high-priority TCP sessions whose average-dropping rates  $a_d$  have exceeded the desired limits  $L$ ,  $\delta p_{p-}$ , the perturbation of the packet-dropping probability of high-priority TCP sessions whose average-dropping rates  $a_d$  are under the desired limits  $L$  and  $\delta p_n$ , the perturbation of the packet-dropping probability of non-priority TCP sessions, the TCP PC-RED Dynamics has three feed-forward channels as shown in Figure 2.

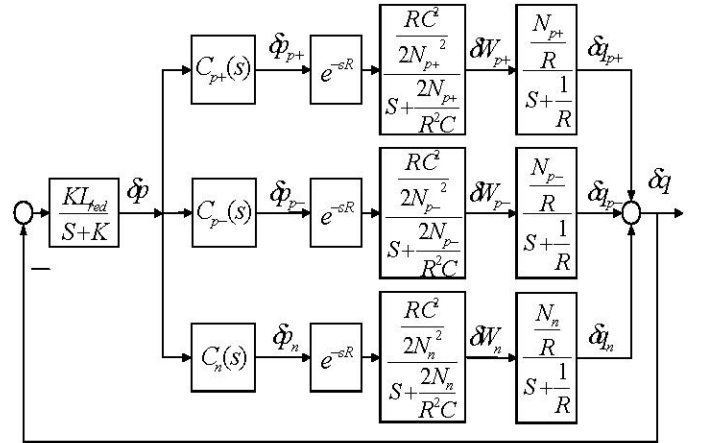


Figure 2 TCP-PCRED Dynamics

Where

$C_{p+}(s)$ : The transfer function between  $\delta p$  and  $\delta p_{p+}$ ;  $C_p(s)$ :

The transfer function between  $\delta p$  and  $\delta p_{p-}$ ;  $C_n(s)$ : The

transfer function between  $\delta p$  and  $\delta p_n$ ;

$\delta p_{p+} + \delta p_{p-} + \delta p_n = \delta p$  ;  $L_{red} = \frac{p_{max}}{\max_{th} - \min_{th}}$ ,

$K = -\frac{\log_e(1-\alpha)}{\delta}$ ;  $\max_{th}$  is maximum threshold;  $\min_{th}$  is

minimum threshold;  $\alpha$  is average queue weight, used to calculate the average queue length;  $\delta$  is Sample time and

$p_{max}$  is the maximum dropping probability.

According to PC-RED algorithm, we have

$$C_{p+}(s) \in \left\{ \frac{N_{p+}}{N} \times 0, \frac{N_{p+}}{N} \times \frac{1}{q} \right\} = \left\{ 0, \frac{N_{p+}}{N \times q} \right\} \quad (2)$$

When  $C_{p+}(s) = 0$ ,

$$\begin{aligned} C_{p-}(s) &= \frac{N_{p-}}{N} + \frac{N_{p+}}{N} \times \frac{N_{p-}}{N - N_{p+}} = \frac{N_{p-}}{N - N_{p+}} \\ C_n(s) &= \frac{N_n}{N} + \frac{N_{p+}}{N} \times \frac{N_n}{N - N_{p+}} = \frac{N_n}{N - N_{p+}} \end{aligned} \quad (3)$$

When  $C_{p+}(s) = \frac{N_{p+}}{N \times q}$ ,

$$\begin{aligned} C_{p-}(s) &= \frac{N_{p-}}{N} + \left( \frac{N_{p+}}{N} - \frac{N_{p+}}{N \times q} \right) \times \frac{N_{p-}}{N - N_{p+}} \\ C_n(s) &= \frac{N_n}{N} + \left( \frac{N_{p+}}{N} - \frac{N_{p+}}{N \times q} \right) \times \frac{N_n}{N - N_{p+}} \end{aligned} \quad (4)$$

#### 4. PC-RED PARAMETER SETTING

The objective of this section is to analyze the model presented in Section 3 thereby guiding the PC-RED parameters setting.

Apart from the traditional RED parameters, such as  $\max_{th}$ ,  $\min_{th}$ ,  $\alpha$  and  $p_{\max}$ , which have been introduced earlier, the extra pre-defined parameters in PC-RED are  $r_{p\max}$  and  $w$ .  $r_{p\max}$  is the maximum threshold of  $r_p$ , the ratio of the high-priority dataflows to the overall data load  $N$ , above which the PC-RED has difficulty in guaranteeing high throughput to high-priority dataflows.  $w$  is the weight factor which is used to calculate the average dropping rate  $a_d$  and can be tuned to adjust the responding speed of  $a_d$  to the dropping action.

##### A. $r_{p\max}$ and System Stability

Comprehensively consider the PC-RED algorithm and Figure 2, it is observed that the main objective of implementing PC-RED as an AQM is to guarantee that high-priority dataflow could achieve high-level throughput, in terms of manipulating  $\delta p_{p+}$  in order to keep  $a_d$  stable and oscillating around the desired dropping limit  $L$  so to control the perturbation of the high-priority dataflow window size  $\delta W_{p+}$ .

Assume all of the TCP dataflows are with the same set of parameters and the packet losses to dataflow  $i$  are described by a Poisson process with time varying rate. Thus at time  $t$ , from Figure 2, the critical state of the system is the entire  $N_p$  has become  $N_{p+}$  due to the lack of control, the following equation can be observed.

$$\begin{aligned} \delta \dot{W}_{p+} &= -\frac{2N_{p+}}{R^2 C} \delta W_{p+} - \frac{RC^2}{2N_{p+}^2} \cdot \frac{N_{p+}}{N \cdot q(t)} \cdot \delta \Phi(t-R) \\ &\approx -\frac{2r_p N}{R^2 C} \delta W_p - \frac{RC^2}{2r_p \cdot N^2 \cdot q(t)} \delta \Phi(t-R) \end{aligned}$$

Where  $\delta \dot{W}$  denotes the time-derivative of the perturbation of the window size.

Thus, the frequency response of the transfer function could be generated as follow.

$$L(j\omega) = \frac{L_{red} \frac{(RC)^3}{4r_p^2 N^3 q} e^{-j\omega R}}{\left( \frac{j\omega}{K} + 1 \right) \cdot \left( \frac{j\omega}{\frac{2r_p N}{R^2 C}} + 1 \right) \cdot \left( \frac{j\omega}{\frac{1}{R}} + 1 \right)}$$

According to  $L(j\omega)$ , if  $r_p$  follows the equation below:

$$r_p < \frac{RC}{2N} \quad (5)$$

when set  $\omega \in [0, \omega_g]$ ,  $\omega_g = \frac{r_p N}{5R^2 C}$ , and the RED parameters setting follows

$$L_{red} \frac{(RC)^3}{4r_p^2 N^3 q} \leq \frac{\omega_g}{K} \quad (6)$$

the close loop system is stable. As we can obtain,

$$\begin{aligned} |L(j\omega_g)| &< \frac{L_{red} \frac{(RC)^3}{4r_p^2 N^3 q}}{\frac{\omega_g}{K}} < 1 \\ \angle L(j\omega_g) &\geq \angle \frac{L_{red} \frac{(RC)^3}{4r_p^2 N^3 q}}{\frac{j\omega_g}{K} + 1} - \omega_g R \geq -90^\circ - 0.1 \frac{180^\circ}{\pi} > -180^\circ \end{aligned}$$

Thus, from Bode Plots, a stable system is achieved.

Therefore, the PC-RED limits  $r_p$  by embedding an advanced packet scheduling mechanism. When the mechanism detects the percentage of high-priority dataflows  $N_p$  has exceeded the ideal limit  $r_{p\max}$ , it would redirect the packet to another path in the network. This algorithm will not be further discussed in the following content.

##### B. $w$ and System Performance

Once the  $r_{p\max}$  is settled, by setting the parameters following (6), a stable system, relatively to both high-priority and non-priority dataflows, can therefore be constructed. However, the PC-RED is using the difference value between  $a_d$  and  $L$  to determine the QOSA setting therefore to trigger

the special dropping mechanism. The way that  $a_d$  fluctuating around  $L$  is also an important issue as the sensitivity of  $a_d$  directly impacts on the PC-RED behaviour.

EWMA (Exponentially Weighted Moving Average) is used here as a low-pass filter to calculate the average-dropping rate  $a_d$ . Thus the packet droppings caused by a short-term increase of the average queue size will not result in a significant increase in the average-dropping rate.

If the  $a_d$  is calculated in every standard dropping time interval  $g$ , and the packet size is fixed under the circumstances of research simplicity, assume the dropping occurs when the number of interval is  $n$ , after  $m$  intervals, there is another dropping.

Thus the following equation can be developed:

$$a_{d(n+m)} = (1-w)^m \times a_{d(n)} + w \times x_{(n+m)} \quad (7)$$

Where

$$m = \frac{t - t_n}{g}$$

$t$  = the current packet dropping time;

$t_n$  = the last packet dropping time;

$g$  = the standard dropping time interval;

$w$  = the weight factor to calculate  $a_d$ ;

$x_{(n+m)}$  = the total amount of dropped packets at time  $t$ .

Take one high-priority dataflow for example, based on (7), assume that the average-dropping rate is initially set to be half of the desired limit  $L$ , the dropped packet is fixed as  $X$  throughout the transmission and there is packet-dropping in every standard interval, thus we have:

$$a_{d(n)} = f(w, n) = (1-w)^n \times \frac{L}{2} - (1-w)^n \times X + X$$

Figure 3 shows the average-dropping rate  $a_d$  as the function of the weight factor  $w$  and the packet dropping counter  $n$ , when the desired limit  $L$  is converted into size and set to be 300 and  $X$  is 500. It is shown that the ascending extent of  $a_d$  increases evidently to the increase of the weight factor  $w$ .

If  $w$  is too large, then the dataflow would be very sensitive to the packet dropping. Thus the probability of  $a_d$  exceeding the desired limit would also increase. This might be propitious to the transmission of high-priority dataflows. However, this might also lead to a complete sacrifice of non-priority dataflows' transmission, as in equations (2), (3) and (4), a large  $N_p$  would result in large  $C_{p-}(s)$  and  $C_n(s)$ , which are the factors used to calculate the dropping probability  $p_{p+}$  and  $p_n$ .

If  $w$  is too small, then  $a_d$  would not be a reasonable reflection of the current dropping action, as it would respond too slowly to the dropping. In this case, the PC-RED would not be able to

detect the current level of QoS the high-priority dataflows are being treated with.

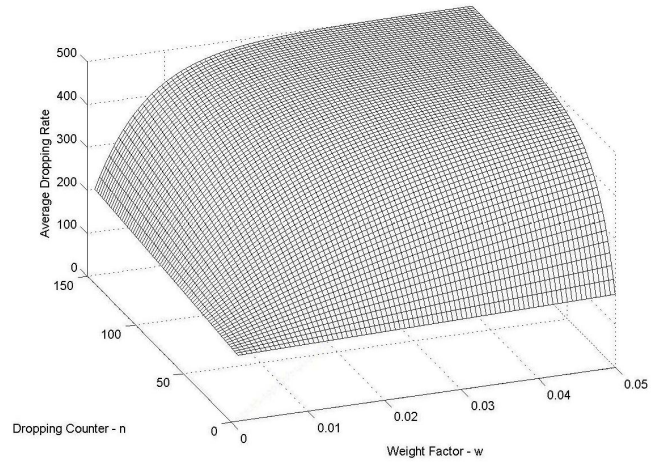


Figure 3  $a_d$  as the function of the weight factor  $w$  and the packet dropping counter  $n$

In the scenario presented in Figure 3, if the PC-RED is expected to respond the dropping action within 50 to 100 standard intervals, the  $w$  should be set in the range of approximate 0.003 to 0.03.

## 5. SIMULATION

Our simulation is implemented in NS2. The aim is to analysis how the change of  $w$  could affect the stability of the transmission of high priority dataflows and non-priority dataflows and how the percentage of high priority dataflows  $N_p$  in overall data load  $N$ , referred as  $r_p$  later in the paper, could influence the PC-RED QoS performance.

The experiment includes the following steps. First, we use traditional RED to analysis the throughput of a network with non-priority dataflow. And then change some node into high priority dataflows and use PC-RED to provide different level QoS. After that, we change the  $w$  in PC-RED to make the average dropping rates more or less sensitive, and study the system performance. Finally, we simulate networks with 10%, and 30% high priority dataflows separately to exam the effect of  $r_p$ .

The single bottle-link network topology is shown in Figure 4.

### Experiment 1

Following the proposition provided in [10], we set a network with 20 dataflows, 100ms delay, link capacity 125 packets/s, and the router parameters as follow:  $p_{\max} = 0.1$ ,  $\max_{th} = 35$ ,  $\min_{th} = 10$ , Buffer = 50,  $\alpha = 4e-3$ . We set dataflow  $i$  to be the only dataflow with priority in the network. PC-RED parameters are:  $N_p = 1$ ,  $w = 3e-3$ .



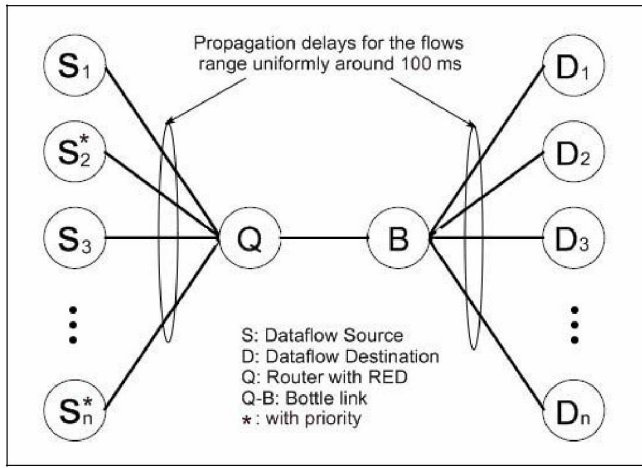


Figure 4 Network Topology

To compare PC-RED with RED, the average queue size and the throughput of one of the dataflows when the router using RED and PC-RED are shown in Figure 5.

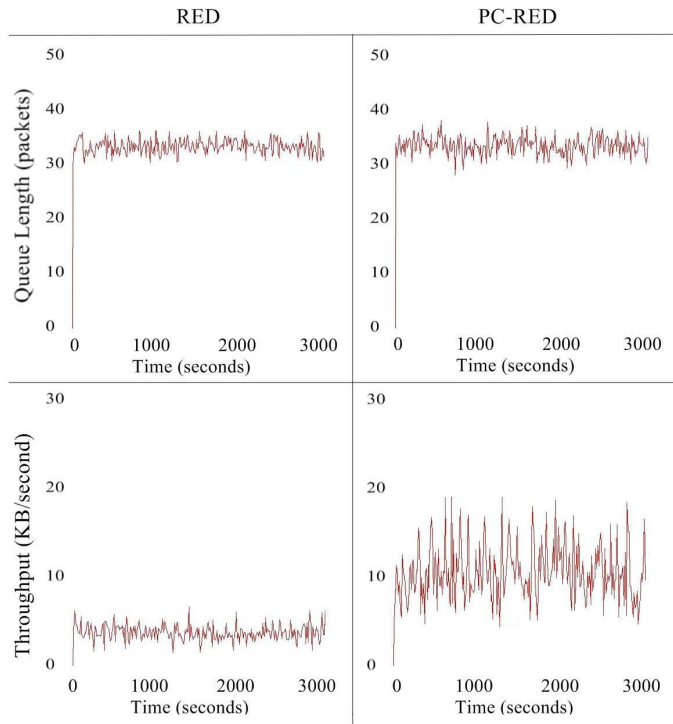


Figure 5 Comparison of RED and PC-RED

The queue length of RED and PC-RED are both stable and oscillating around 33 packets. However, due to the extra operating time caused by the PC-RED Priority List Maintenance Function and the special Pick Packet to Drop Mechanism, Figure 5 shows that the fluctuation of queue length in PC-RED is larger than the one in RED. However, the performance is still satisfying.

The throughput of high-priority dataflow increases from around 4KB/s in traditional RED to around 12KB/s in PC-RED as shown in Figure 5.

So, comprehensively considering the queue length and the throughput, PC-RED is a better queue management method which can provide higher level of QoS to high-priority

dataflows than RED which is enhancing the fairness among dataflows.

### Experiment 2

In this simulation, we set  $w=5e-3$  and  $w=3e-5$ , then compare the  $a_d$  and throughput of the dataflow under these two conditions in Table I. The desired limit  $L$  of high-priority dataflow is converted into size and set to be 70 bytes per second and the simulation duration is 3000 seconds throughout the following simulations in the rest of the paper.

TABLE I  
COMAPRTION OF DIFFERENT  $w$

$w$	$a_d$ (bytes/s)	Fluctuation (bytes/s)	Throughput (KB/s)		
			min	max	average
$5e-3$	71	8	3	14	8.5
$3e-3$	Under $L$ until 2700 s	-	0	5	2.5

When  $w=5e-3$ , it is large enough to make the  $a_d$  sensitive enough to detect the dropping action, which is affecting the data transfer. Therefore, by calculating the difference value between  $a_d$  and  $L$ , the Priority List Maintenance Function can set the QOSA bit in time to inform the Pick Packet to Drop Mechanism. This action guarantees a certain level of quality of service to high-priority dataflow, and ensures a higher rank of throughput, which is around 8.5 KB/s as shown in Table I.

When  $w$  is small, the  $a_d$  responses tardily to the dropping action. In this simulation, although the PC-RED drops packets from this high-priority dataflow throughout the entire 3000 seconds, it takes 2700 seconds for the  $a_d$  to smoothly grow up to the desired limit  $L$ . And the PC-RED cannot help the high-priority dataflow to achieve a satisfying throughput in a long scale of time due to the lack of QOSA caution caused by the slow reaction of  $a_d$ . As a result, the throughput is less than one third of the previous one.

### Experiment 3

The compare of the throughput of the high-priority dataflow and the throughput of the non-priority dataflow is shown in Table II when  $r_p$  is set to be 10% and 30%. In this simulation we set  $w=e-3$  and remain all of the other parameters the same as pervious.

TABLE II  
COMAPRTION OF DIFFERENT  $r_p$

$r_p$	Priority	$a_d$ (bytes/s)	Fluctuation (bytes/s)	Throughput (KB/s)		
				min	max	ave
10%	High	70	2	2.5	19	10.8
10%	Non	120	23	0	6	3
30%	High	80	4	4	12	8
30%	Non	130	45	0	1	0.25

When there is 10% load level with high-priority in the entire network, the throughput of high-priority dataflow is about 3.2 times as the throughput of non-priority dataflow. And the average dropping rate  $a_d$  of high-priority dataflow is controlled around the desired limit  $L$  as well. However, the  $a_d$  of the non-priority dataflow is 71% larger than the one of high-priority dataflow.

When  $r_p$  rises up to 30%, since the ratio of  $N_p$  to  $N$  is larger, according to (4), the dropping probability of high-priority dataflow would also increase. And this certainly leads to a rise of the  $a_d$  as well. Based on (5) and (6), the system has reached its critical state due to the large  $r_p$ . Although the PCRED priority checking function has been enabled, it is still unable to control the  $a_d$  of high-priority dataflow under the allowable dropping rate range  $L$ . The  $a_d$  is 14% beyond the limit. And the  $a_d$  of non-priority dataflow is even higher and oscillates more acutely than previous, which leads to an extremely low level of throughput. Throughout the simulation, the throughput of non-priority dataflow remains in an extremely low level, mainly is zero, and goes up to 1KB/s occasionally. In addition, the throughput of high-priority dataflow is also lower than it was in a network with smaller amount of high-priority dataflows.

## 6. CONCLUSION

In this paper, a new RED with Priority Checking function is provided as an AQM method of a network supporting both high-priority dataflows and non-priority dataflows. The Priority Checking function does not involve the traditional RED parameters. But the extra function process times do affect the queue status and slightly increase the fluctuation when the performance is still stable.

The NS2 simulation shows that the PCRED has guaranteed a certain level of average dropping rate of high-priority dataflows and helped the data source to achieve desirable transient performance. The influences of the average dropping rate weight factor and the percentage of high-priority dataflows in overall load level have been studied.

How to guarantee the transient performance of non-priority dataflows in a network supporting PCRED and the implementation of such AQM method with Priority-Checking function in a wider range of networks such as multi-switched and wireless network and the further research on the design of the advanced packet scheduling mechanism in PC-RED would be studied in the future.

## ACKNOWLEDGEMENTS

Thanks to Loughborough University Computer Science Department for sponsoring and to Fang YAO, Khusvinder GILL, and Wu CHEN for their help and inspiration.

## REFERENCES

[1] L. Le, J. Aikat, K. Jeffay and F. D. Smith, The effect of active queue management on Web performance, in

Proceedings of SIGCOMM'03, Karlsruhe, Germany, Aug. 2003, pp. 265-276.

[2] S. Floyd and V. Jacobson, Random Early Detection Gateways for Congestion Avoidance, IEEE/ACM Transactions on Networking, vol. 1, no. 4, pp. 397-413, Aug. 1993.

[3] B. Wydrowski and M. Zukerman, GREEN: An Active Queue Management Algorithm for a Self Managed Internet, Proceedings of ICC 2002, New York, vol. 4, pp. 2368-2372, 2002.

[4] S. Athuraliya, V. H. Li, S. H. Low, and Q. Yin, REM: Active Queue Management, IEEE Network, vol. 15, no. 3, pp 48-53, May 2001.

[5] M. May, J. Bolot, C. Diot and B. Lyles, Reasons not to deploy RED, in Proceedings of 7th International Workshop on Quality of Service, London, 31 May-4 June 1999, pp. 260-262.

[6] Holot, C. V.; Misra, V.; Towsley, D.; Gong, W., Analysis and Design of Controllers for AQM Routers Supporting TCP Flows, IEEE TRANSACTIONS ON AUTOMATIC CONTROL - 2002 ; VOL 47; PART 6 ; Pages: 945-959

[7] D. Katabi and C. Blake, A note on the stability requirements of adaptive virtual queue, LCS Document Number: MIT-LCS-TM-626, 2-13-2002. [Online] Available: <http://www.lcs.mit.edu/publications/pubs/pdf/MIT-LCS-TM-626.pdf>.

[8] S. Floyd, R. Gummadi and S. Shenker, Adaptive RED: an algorithm for increasing the robustness of RED's active queue management. [Online] Available: <http://www.icir.org/floyd/papers/adaptiveRed.pdf>, Aug. 2001.

[9] M. Christiansen, K. Jeffay, D. Ott and F. D. Smith, Tuning RED for Web traffic, in Proceedings of ACM SIGCOMM 2000, Stockholm, Sweden, 2000, pages 139-150.

[10] W. CHEN, Y. LI, S.H. YANG, An Average Queue Weight Parameterization in a Network Supporting TCP with RED, IEEE ICNSC, 2007

[11] IBM Corporation, Quality of Service White Paper: Integrated QoS IBM WebSphere and Cisco Can Deliver End-to-End Value (IBM Corporation, NC USA, 2001)

[12] K. Niles, M. Goncalves, IPv6 Networks (McGraw-Hill, NY, 1998).

[13] C. Armitage, Quality of Service in IP Networks Foundation for a Multi-Service Internet (MTP, Lucent Technologies, USA, 2000)

[14] Zhang, L., Zheng, L., IPv6 traffic with multi-class QoS in VPN, COMPUTER NETWORKS -AMSTERDAM- - 2001 ; VOL 37; NUMBER 3-4 ; Pages: 263-275

[15] V. Misra, W.B. Gong, and D. Towsley, Fluid-based Analysis of a Network of AQM Routers Supporting TCP Flows with an Application to RED, in Proceedings of ACM/SIGCOMM, 2000.

[16] C. Holot, V. Misra, D. Towlsey, and W. Gong, A control theoretic analysis of red, in the proceedings of IEEE Infocom 2001, Anchorage, Alaska, Apr. 2001.