International Journal of Innovative Computing, Information and Control
Volume **12**, Number **1**, February **2016**

ICIC International ©2016 ISSN 1349-4198
pp. **67–76**

A DYNAMIC BANDWIDTH MANAGEMENT SCHEME FOR HIGH-SPEED NETWORKS

Jui-Pin Yang[1] and Yi-Chang Zhuang[2]

[1]Department of Information Technology and Communication
Shih Chien University
No. 200, University Rd., Neimen Dist., Kaohsiung 845, Taiwan
juipinyang@gmail.com

[2]South
Industrial Technology Research Institute
No. 195, Chung Hsing Rd., Sec. 4, Chutung, Hsinchu 31040, Taiwan

Received June 2015; revised October 2015

Abstract. *Many core-stateless bandwidth management mechanisms have been proposed to resolve the scalability problem of the core routers, but they heavily rely on the edge routers or anterior core routers to attach correct flow information to departure packets. The relevance may result in extremely unfair bandwidth sharing among populated flows; therefore, the dynamic audit fair queueing (DAFQ) scheme is proposed in this paper which is capable of supporting robust fairness of bandwidth sharing. The main idea of the DAFQ is to detect the malicious flows and then adequate control policies are applied to restraining those ones. Computer simulation compares the fairness of the DAFQ with that of some well-known mechanisms. The simulation results show that the DAFQ is suitable to be deployed in the high-speed core routers due to its simplicity. Besides, the DAFQ is able to support robust fairness of bandwidth sharing more than the other compared mechanisms under various traffic situations.*

Keywords: Bandwidth management, Fairness, Router, Scalability

1. Introduction.

In the traditional core routers, only a simple FIFO scheduling might be applied to managing the bandwidth sharing. However, their fairness may be poor if some of the populated flows are aggressive. Many solutions have been proposed to deal with this issue that could be roughly divided into two categories: per-flow queueing and per-flow dropping. In per-flow queueing scheme, a dedicated buffer should be assigned to each populated flow, which is used to accommodate their arrival packets. Afterward they will be delivered to the next routers according to specific scheduling algorithms [1-4]. As for per-flow dropping scheme, only a single FIFO buffer is required, and furthermore the whole buffer capacity is shared for all populated flows. Each arrival packet will be dropped or accepted according to the present traffic conditions such as queue length, and per-flow load [4,8,9]. These above schemes have to maintain per-flow state. In other words, their complexity is approximately proportional to the number of the populated flows. In general, the core routers may reside at least several thousands of concurrent flows. Hence, they may be infeasible to be deployed in the high-speed core routers.

To be aimed at the feasibility, a core-stateless scheme, core-stateless fair queueing (CSFQ) was proposed by Stoica et al. [10]. In this scheme, the whole routers are classified as edge or core routers. The edge routers need to maintain per-flow state (i.e., flow arrival rate), and furthermore the information should be inserted into the corresponding departure packets of individual flow. The core routers only equip with a single FIFO buffer. Besides, a probabilistic dropping algorithm is adopted to decide that arrival packets are dropped or not. Although the CSFQ could eliminate the sophisticated implementation in the core routers, it converges toward the correct fair share rate slowly. The drawback may give rise to poor fairness, so that some of improved mechanisms were proposed, which could support better fairness [11-14].

It is a common sense that core-stateless bandwidth management mechanisms strongly depend on the edge routers or preceding core routers to attach correct flow information to corresponding departure packets. If they carry with incorrect flow information due to some reasons such as malice or device malfunction, the core routers are incapable of handling the condition. Hence, some new mechanisms were proposed which could alleviate the influence of the malicious flows on the degree of fairness [15-20]. Particularly, self-verifying CSFQ (SV-CSFQ) [16], a variation of the CSFQ attracts a lot of attention. However, it has several performance drawbacks that may cause the unsuitability of the SV-CSFQ. Firstly, the sizes of the VerifyTable and ContainTable parameters are constant, so it could not adapt well to the various traffic conditions. Secondly, if the arrival packets carry with extremely inaccurate flow information, a long containment interval may be produced. This might result in the core routers to maintain the unneeded flow information, because the flows might transfer from the malicious state to the normal state already. Finally, the punishment on the malicious flows might be excessive that could result in poor fairness and lower throughput. To improve above disadvantages and upgrade the fairness further, a simple and effective fair queueing mechanism, DAFQ, is proposed in this paper which can support good and robust fairness under various traffic conditions. Finally, this paper is an extended and corrected version of our previous study [21].

The reminder of the paper is organized as follows. In the next section, we review some previous work related with fairness of bandwidth sharing. In Section 3, we describe the basic operation and details of the DAFQ scheme. In Section 4, we use computer simulation to compare the fairness of the DAFQ, CSFQ, SV-CSFQ, DRR and FIFO schemes. Finally, the conclusions and future studies are presented in Section 5.

2. **Related Work.** In the past, fair queueing mechanisms were proposed which could support quite perfect fairness of bandwidth sharing [1,2]. However, their implementation is too complicated to suit the high-speed requirement of the core routers. One variant, deficit round robin (DRR) could reduce the complexity, but certain fairness is also degraded [3]. The DRR needs to classify each arrival packet in accordance with individual flow, and then places them on the corresponding flow queues. In addition, it has to maintain the deficit counter and populated packets for each current flow, i.e., per-flow state. The DRR is quite infeasible, if a large number of populated flows exist. Next, a mechanism was proposed as an enhancement of the DRR [4], which could be viewed as a combination of the DRR and random early detection (RED) [5]. It probabilistically discards some arrival packets before their corresponding flow queues are full. This is useful to avoid the drop-tail phenomenon that may result in unfair bandwidth sharing. As for the RED, it utilizes average queue size and a probabilistic model to decide whether an arrival packet is qualified to enter the buffer. Based on the RED, some other variants aim at upgrading the performance of the RED while the simplicity is kept [6,7].

The RED is simple, but it could not support robust fairness due to aggressive or unresponsive flows. Hence, a more complicated mechanism, flow random early discard (FRED) was proposed [8]. Unlike the DRR, the FRED only needs a single FIFO buffer. Besides, the number of discarded packets, average of queued size, minimum and maximum thresholds for each populated flow must be maintained. The FRED could correctly evaluate that how many arrival packets for individual flow should be dropped, so it could support approximately perfect fairness. In a word, the above mentioned mechanisms that support good fairness need to maintain per-flow state. To maintain the per-flow state may be extremely complicated, so they are unsuitable to be deployed in high-speed core routers.

Correspondingly, core-stateless fair queueing (CSFQ) was proposed which could simplify the implementation and provide reasonable fairness in the core routers [10]. To support better fairness, an improved mechanism combines the CSFQ with a probabilistic method which is used to estimate the number of populated flows in the routers [11]. Besides, the rainbow fair queueing (RFQ) is also a revised scheme from the CSFQ [12], but its implementation is different from the CSFQ. Firstly, the state information carried by the arrival packets is the color layers, rather than explicit flow arrival rate. Secondly, the CSFQ needs exponential averaging to estimate the fair share rate, but the RFQ only evaluates the threshold-based dropping. The RFQ is relatively simple, so it is more suitable for core routers. However, it has the granularity drawback (i.e., classification of flow arrival rate) that may result in poor fairness.

Most core-stateless mechanisms strongly rely on the arrival packets with the correct flow information; in other words, incorrect flow information may cause them to fail. Correspondingly, CHOKEs scheme is unneeded to maintain any per-flow state neither the edge routers nor the core routers [13]. Although the complexity in the edge routers is removed, the degree of fairness may be degraded due to insufficient flow information. To boost the performance of the CHOKEs, an enhancement scheme, comparing and controlling unresponsive flows (CCU) was proposed. It adds a new method to detect and control the unresponsive flows [15]. The CCU varies from the RED scheme, so it performs better under specific traffic situations. Furthermore, a self-verifying CSFQ (SV-CSFQ) was proposed to reinforce the fairness of the CSFQ by resisting the effect of incorrect flow information [16]. Although some mentioned researches could improve the fairness by suppressing the malicious flows, their capabilities to support good and robust fairness still leave space for further improvements.

3. Dynamic Audit Fair Queueing.

The DAFQ is capable of fair bandwidth sharing through instant detection and restraints on the malicious flows. In the first place, the DAFQ identifies the aggressive flows by discovering the populated flows that have greater number of resided packets in the FIFO buffer. Consequently, these flows will be enrolled into the aggressive flow list which is constructed by comparing each admissible packet with a randomly selected packet in the FIFO buffer. When the IP addresses of both packets are equivalent, a hit happened. If the hit number of some populated flows is larger than the average hit number during two continuous time intervals, they will be identified as aggressive flows. A formal definition is described as follows.

Definition 3.1. *If $H_i(n) \geq H(n)/N(n)$, and $H_i(n-1) \geq H(n-1)/N(n-1)$, then the flow i is recognized as an aggressive flow.*

$H_i(n)$: The hit number of flow i during time interval $[n-1, n]$
$H(n)$: The total hit number during time interval $[n-1, n]$
$N(n)$: The number of populated flows with hit during time interval $[n-1, n]$

To distinguish the aggressive flows is meaningful, because they occupy more bandwidth than the others at the moment. Hence, the unfair bandwidth sharing may occur. The non-aggressive flows almost have no impact on the degree of fairness, because they probably occupy bandwidth less than fair share rate. As a result, it is unnecessary to further recognize whether they are malicious or not. The flow arrival rate for each flow in the aggressive flow list should be estimated at the end of the next time interval. In the meantime, the flow arrival rate carried by each arrival packet of individual aggressive flow should be averaged. Obviously, there is always a gap between both above coefficients due to traffic burstiness, buffer effect, estimation algorithms, etc. It is unneeded to estimate both precisely, because the information is viewed as a criterion to decide the malicious flows further. To avoid excessively identifying aggressive flows with malicious, an adjustment parameter (θ) is added. This can effectively limit the number of malicious flows to certain quantity that is beneficial to decrease the computing overhead on the core routers.

Definition 3.2. *If* $\frac{\sum_{k=1}^{m} r_i(k,n+1)}{m} < \frac{F_i(n+1)}{T}.(1-\theta)$ *or* $\frac{\sum_{k=1}^{m} r_i(k,n+1)}{m} > \frac{F_i(n+1)}{T}.(1+\theta)$*, aggressive flow i is identified as a malicious flow further.*

T*: The duration of each time interval*

$F_i(n+1)$*: The amount of arrival packets of flow i during time interval* $[n, n+1]$

$r_i(k, n+1)$*: The flow arrival rate of the kth arrival packet of flow i during time interval* $[n, n+1]$

m*: The total number of arrival packets of flow i during the time interval* $[n, n+1]$

Once these malicious flows are recognized, they will be enrolled into the malicious flow list. According to this list, arrival packets that belong to malicious or non-malicious flows will go through different dropping algorithms respectively. The algorithm that estimates the fair share rate is originated from the concept of piecewise linear method, namely (1).

$$\frac{\alpha_{n+1} - \alpha_n}{C - A(n)} = \frac{\alpha_n - \alpha_{n-1}}{A(n) - A(n-1)} \tag{1}$$

$A(n)$: The whole acceptance rate at time n

α_{n+1}: The fair share rate for time interval $[n, n+1]$

C: The outgoing rate of a single output link

In addition, the $A(n)$ is estimated using (2).

$$A(n) = \left(1 - e^{-T/K_d}\right) \cdot \frac{L(n)}{T} + e^{-T/K_d} \cdot A(n-1) \tag{2}$$

K_d: A constant parameter

$L(n)$: The amount of total admissible packets during time interval $[n-1, n]$

By simplifying (1), the formula that calculates α_{n+1} can be derived and shown in (3).

$$\alpha_{n+1} = \alpha_n + \frac{C - A(n)}{A(n) - A(n-1)} \cdot (\alpha_n - \alpha_{n-1}) \tag{3}$$

If $A(n) \neq A(n-1)$ and $\alpha_n \neq \alpha_{n-1}$, then (3) is meaningful; otherwise, (4) is applied to handling the exceptions in (3).

$$\alpha_{n+1} = \frac{\alpha_{n-1} + \alpha_n}{A(n-1) + A(n)} \cdot C \tag{4}$$

To decide whether arrival packets for the non-malicious flows are admissible to enter the FIFO buffer, a simple probabilistic dropping algorithm is adopted, namely (5).

$$\max\left(0, 1 - \frac{\alpha_{n+1}}{r_i(k, n+1)}\right) \tag{5}$$

As for the malicious flows, only the limited amount of $\alpha_{n+1} \cdot T$ bits can be accepted. However, this may cause them with global synchronization because of drop-tail. If they are compliant with TCP, they will decrease their transmission rates. This is helpful to support good fairness because the phenomenon can be viewed as an additional punishment policy regarding the malicious flows. Consequently, the degree of fairness might go up.

The DAFQ manages the non-malicious flows with core-stateless policy, but stateful policy is used to control the malicious flows. However, the overall complexity of the

DAFQ may be close to the core-stateless mechanisms. In general, the number of the malicious flows is very few, because most users are well-behaved and short of professional network skill. Once an arrival packet of the non-malicious flows was accepted, whose flow rate should be relabeled as α_{n+1}, in case $r_i(k, n+1) > \alpha_{n+1}$. For all admissible arrival packets of the malicious flows, all of them should be relabeled as α_{n+1} without exception.

Furthermore, two supplements are attached; firstly, if $A(n) < C$, this time interval will not add to the elapsed time interval. Secondly, if a flow is already recorded in the malicious flow list, it should be monitored at the next detection. Because of the malicious behavior is possible to persist. Otherwise, the flow information will be removed from the list. In addition, a fairness deviation (σ) is defined in Equation (6). The equation is used to compare the degree of fairness for different bandwidth management mechanisms.

$$\sigma = \left[\frac{\sum_{i=1}^{n} \left\{ [D_i - \min(r_i, f)] / \min(r_i, f) \right\}^2}{n} \right]^{1/2} \tag{6}$$

D_i: The shared flow rate of flow i
f: The fair share rate
n: The amount of populated flows
r_i: The flow arrival rate of flow i

The D_i, r_i and n obtain from the simulation conditions and results. As for f, it can be derived from the equation of max-min fairness in (7).

$$\sum_{i=1}^{n} \min(r_i, f) = C \tag{7}$$

Certainly, the sum of outgoing rate for all populated flows is smaller or equal to the outgoing rate of an output link. The formula is described as

$$\sum_{i=1}^{n} D_i \leq C \tag{8}$$

From (6), (7) and (8), the worst σ is derived and the final result is presented (9).

$$\sigma = \left(\frac{(C/f - 1)^2 + n - 1}{n} \right)^{1/2} \tag{9}$$

4. **Simulation Results.** A congested link has a capacity of 10 Mbps, a latency of 1 ms, buffer size of 60 KB, simulation time of 200 sec, $T = 200$ ms and packet size of 1 KB. Each flow is imitated as an "*ON-OFF*" traffic model which has infinite data to deliver. The model consists of two states: one is "*ON*" state and the other one is "*OFF*" state. The duration of "*ON*" state and "*OFF*" state belongs to the geometrical distribution with parameter $P_{on\text{-}off}$ and $P_{off\text{-}on}$, respectively. $P_{on\text{-}off}$ represents the transition probability from "*ON*" state to "*OFF*" state, but $P_{off\text{-}on}$ represents the transition probability from "*OFF*" state to "*ON*" state. When the model stays in "*ON*" state, one packet will be generated; otherwise, no packet is generated.

In the DAFQ, $K_d = 200$ ms and $\theta = 0.2$. As for the CSFQ, K, K_α, and K_c are all set at 100 ms. Furthermore, the parameters of the SV-CSFQ are set as follows: $H_u = 0.2$, buffer threshold of 16 KB, $K = 100$ ms, and $K_\alpha = 200$ ms. When flow number is equal to 10, 20 and 30, the size of shared table is set at 8, 15 and 20, respectively. Correspondingly, the maximum number of entries of the contained flows is set at 5, 10 and 15. In the DRR, total buffer is shared and then dedicated to each flow. Besides, the designate quota of a round for each flow is of 1 KB. Unless otherwise specified, all the mentioned parameters are adopted throughout the computer simulation.

In the first case, total number of populated flows is twenty including three malicious flows all with different accuracy ratios (ε) of flow arrival rate. Their traffic load is all equal to 9 Mbps with $P_{on\text{-}off} = 0.0556$ and $P_{off\text{-}on} = 0.5$. The other seventeen non-malicious flows are well-behaved and their flow arrival rates are all equal to 1 Mbps with $P_{on\text{-}off} = 0.5$ and $P_{off\text{-}on} = 0.0556$. Furthermore, the accuracy ratios for all malicious flows change at the same time. According to the traffic conditions, the link between the core router 1 and core router 2 is severely congested, and f is equal to 0.5 Mbps.

The fairness deviation (σ) of the DAFQ, SV-CSFQ, CSFQ, DRR and FIFO is shown in Figure 1. No matter how ε shifts, σ of FIFO and DRR always performs like a horizontal flat line. Because both schemes are independent of flow information which is used to decide whether arrival packets should be dropped or not, the change of ε of flow arrival rate of the malicious flows has no effect on the degree of fairness. Additionally, the σ of the DRR exceeds our expectations. The cause of the result is that the buffer capacity is insufficient in the DRR (i.e., each flow only has buffer size of 3 KB), so many burst arrival packets of the non-malicious flows are discarded. This can give rise to the poor fairness of the DRR. However, the DRR still performs better than the FIFO. As for the FIFO, the obtained bandwidth for each flow is approximately proportional to individual traffic load. Hence, the FIFO has the worst fairness when $\varepsilon \geq 0.1$.

The σ of the CSFQ increases nearly linear when ε decreases. The malicious flows can seize bandwidth from the non-malicious flows because incorrect flow arrival rate results in the estimation of fair share rate to fail. Correspondingly, the non-malicious flows are unable to fairly get their bandwidth. Ultimately, the CSFQ performs worse than the FIFO when $\varepsilon < 0.1$. The result reveals that CSFQ scheme is incapable of resisting the attack from the malicious flows. Specifically, its fairness will become very worse if the ε is

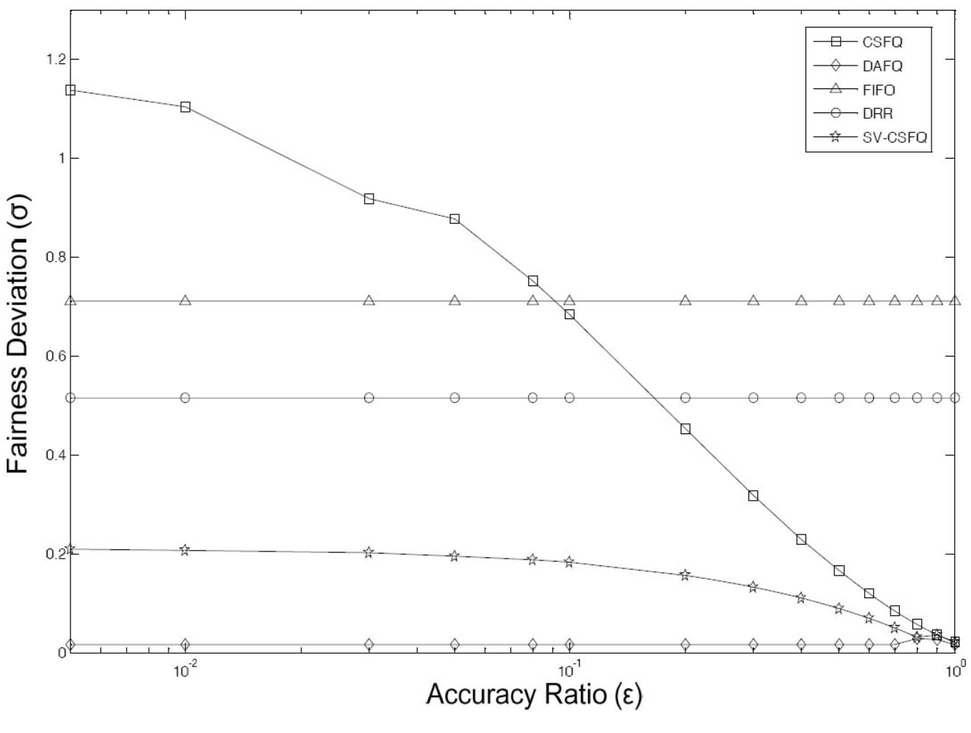

FIGURE 1. Fairness deviation versus accuracy ratio of three malicious flows among twenty populated flows

relatively low. In a word, a simple FIFO scheme may perform better than a well-organized CSFQ under certain specific traffic conditions.

For all ε, the SV-CSFQ has the best fairness at $\varepsilon = 1.0$, because the malicious flows carry correct flow arrival rate. The σ at $\varepsilon = 0.9$ is higher than $\varepsilon = 1.0$, because the malicious flows may be seldom identified due to $H_u = 0.2$. Hence, these flows could occupy more bandwidth over fair bandwidth sharing. The σ at $\varepsilon = 0.8$ is lower than $\varepsilon = 0.9$. The malicious flows are more possible to be detected and then constrained, so they may acquire less bandwidth. However, the malicious flows may occupy more bandwidth if they are not detected. By the counteraction, this result could be easily figured out. Also, the σ is close to a constant when $\varepsilon < 0.1$. The majority of arrival packets of the malicious flows are dropped due to excessive containment, so their deserved bandwidth is completely occupied by the non-malicious flows. In general, the malice may result from device malfunction or estimation algorithms, etc., so the SV-CSFQ may punish the innocent flows. This may cause the SV-CSFQ to have poor fairness and low throughput.

In the DAFQ, σ is the highest at $\varepsilon = 0.8$. The malicious flows are more possible to be identified, so the σ should be lower. However, the result is reversed. The significant reason is that more their arrival packets may be accepted whenever the malicious flows are not identified. From the simulation result, the latter apparently has a greater influence than the former. At $\varepsilon = 0.9$, flows are seldom identified as malicious because of $\theta = 0.2$. The malicious flows can only occupy a limited additional bandwidth, because the ε is high. The result differs from the SV-CSFQ, because the DAFQ does not excessively punish the malicious flows and still guarantees their deserved bandwidth. Furthermore, the σ is almost a constant when $\varepsilon \leq 0.7$.

In the second case, there are ten populated flows including one malicious flow, whose traffic load is equal to 6 Mbps with $P_{on\text{-}off} = 0.1667$ and $P_{off\text{-}on} = 0.25$. Besides, there are nine non-malicious flows including five ones whose traffic load is all equal to 2 Mbps with $P_{on\text{-}off} = 0.5$ and $P_{off\text{-}on} = 0.125$, and four ones whose traffic load is all equal to 6 Mbps with $P_{on\text{-}off} = 0.1667$ and $P_{off\text{-}on} = 0.25$. The simulation result is depicted in Figure 2. Besides, f is equal to 1 Mbps. The DRR also has the best fairness because the buffer capacity is still sufficient for each flow. In the CSFQ, the σ is a constant when $\varepsilon < 0.05$; hence, it is faster than the third case. There is only a malicious flow, so it can get the maximum required bandwidth (i.e., 6 Mbps) more quickly at larger ε.

The SV-CSFQ performs much better than the CSFQ. Only a malicious flow is contained, so the containment has less impact on the fairness of the SV-CSFQ. Besides, the single malicious flow is completely contained when $\varepsilon \leq 0.03$. As a result, the σ is stable and equal to 0.33. In the DAFQ, the σ begins to stabilize again when $\varepsilon \leq 0.7$. Like the third case, the DAFQ still performs close to the DRR. From above simulation results, DAFQ scheme is capable of supporting good and robust fairness under different traffic situations. Next, we will discuss the effect of the adjustment parameter (θ) related with the DAFQ.

In the last case, there are ten flows including one malicious flow whose traffic load is equal to 9 Mbps with $P_{on\text{-}off} = 0.0556$ and $P_{off\text{-}on} = 0.5$, and nine non-malicious flows whose traffic load is all equal to 1 Mbps with $P_{on\text{-}off} = 0.5$ and $P_{off\text{-}on} = 0.0556$. The simulation result of DAFQ scheme related with different θ is depicted in Figure 3. Besides, f is equal to 1 Mbps. If ε is equal to 1.0, 0.9 and 0.8, their σ is reasonable for any θ, because ε of the malicious flow is higher at the moment. Besides, the smaller ε will result in worse σ, because the malicious flow is able to occupy more bandwidth. The malicious flow may not need to be contained when $\varepsilon \geq 0.8$. Furthermore, this also can reduce the unneeded overhead to maintain the malicious flow information.

At $\varepsilon = 0.7$, the σ becomes larger suddenly when $\theta \geq 0.4$. If the malicious flow is not identified with smaller ε, it will occupy more bandwidth. At $\varepsilon = 0.6$, the σ becomes larger

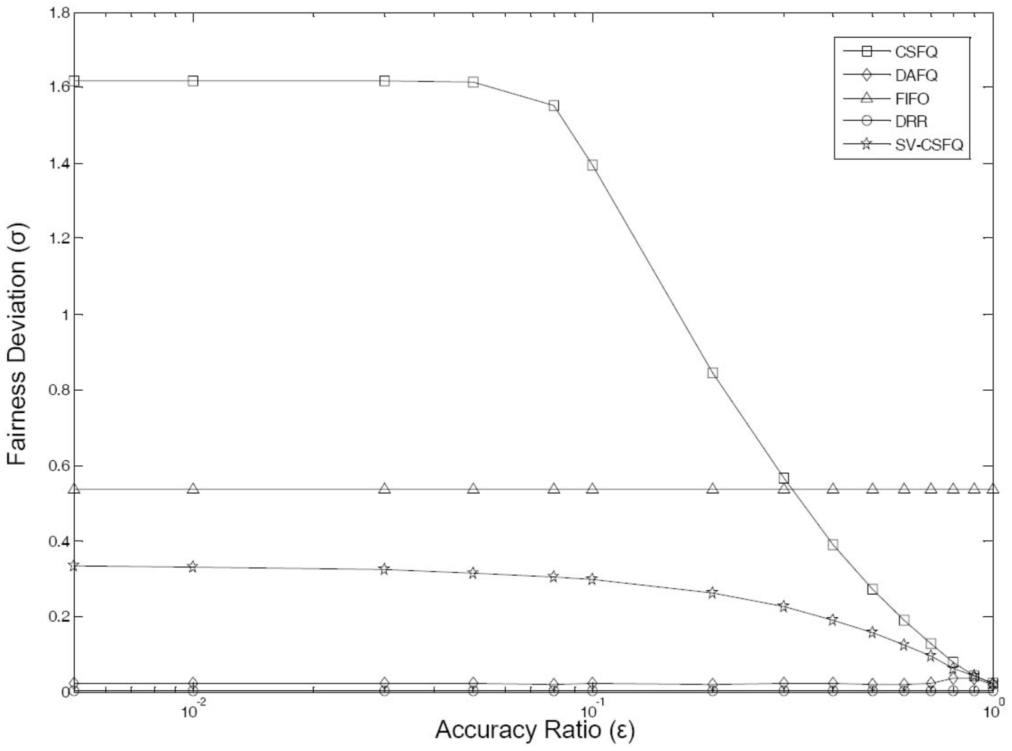

FIGURE 2. Fairness deviation versus accuracy ratio of a single malicious flow among ten populated flows

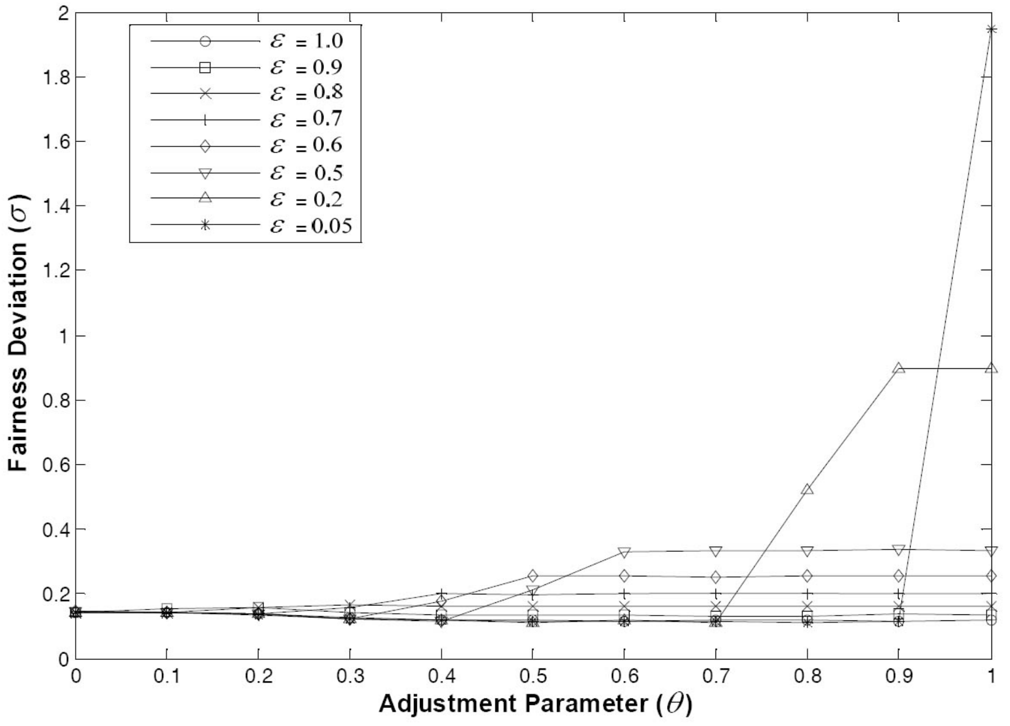

FIGURE 3. Fairness deviation versus adjustment parameter in the DAFQ with a single malicious flow among ten populated flows

suddenly when $\theta \geq 0.5$. From both results, if the malicious flow is not identified with smaller ε, the containment or not has greater impact on the fairness. The difference is the most obvious at $\varepsilon = 0.05$. In general, the DAFQ has good and robust fairness at $\theta = 0$ for all ε. However, the overhead may be huge. Once a flow is identified as aggressive, it must be identified as malicious further. Hence, the DAFQ maintains unneeded flow information. On the other hand, the DAFQ provides poor fairness at $\theta = 1$ for smaller ε. However, the overhead might be tiny because almost no malicious flow is identified. To consider the overhead and performance both, the adequate value of θ is between 0.1 and 0.3. From our computer simulation, the DAFQ is able to support good and robust fairness.

5. **Conclusions.** In this paper, an approximately core-stateless fair queueing mechanism, DAFQ is proposed. The scheme is able to deal with malicious behavior caused by some of flows that may lead to unfair bandwidth sharing. On the contrary, the stateful fair queueing mechanisms need to maintain per-flow information, so they are too complicated to achieve the high-speed requirement in the core routers. The DAFQ only needs to maintain a little flow information related to the malicious flows. Hence, the increased complexity of the DAFQ is quite limited. On the other hand, the DAFQ does not need to check out the flow information of the arrival packets of the malicious flows after they had been identified. As a whole, the complexity of the DAFQ is lower than that of well-known core-stateless mechanisms but it can support more robust fairness. From the simulation results, the DAFQ is capable of supporting robust and much better fairness than that of the compared mechanisms except for the DRR scheme. However, the DAFQ is able to perform better fairness whenever the DRR equips with insufficient buffer size.

Acknowledgment. This work is partially supported by the Shih Chien University under the grant number USC 104-08-01007.

REFERENCES

[1] A. Parekh and R. Gallager, A generalized processor sharing approach to flow control: The single node case, *IEEE/ACM Trans. Networking*, vol.1, no.3, pp.344-357, 1993.
[2] A. Demers, S. Keshav and S. Shenker, Analysis and simulation of a fair queueing algorithm, *Proc. of the SIGCOMM Symposium on Communications Architectures and Protocols*, pp.1-12, 1989.
[3] M. Shreedhar and G. Varghese, Efficient fair queuing using deficit round-robin, *IEEE/ACM Trans. Networking*, vol.4, no.3, pp.375-385, 1996.
[4] G. Hasegawa, T. Matsuo, M. Murata and H. Miyahara, Comparisons of packet scheduling algorithms for fair service among connections on the Internet, *Proc. of the IEEE INFOCOM*, pp.1253-1262, 2000.
[5] S. Floyd and V. Jacobson, Random early detection gateways for congestion avoidance, *IEEE/ACM Trans. Networking*, vol.1, no.4, pp.397-413, 1993.
[6] W. C. Fang, D. D. Kandlur, D. Saha and K. G. Shin, A self-configuring RED gateway, *Proc. of the IEEE INFOCOM*, pp.1320-1328, 1999.
[7] T. Alemu and A. Jean-Marie, Dynamic configuration of RED parameters, *Proc. of the IEEE GLOBECOM*, pp.1600-1604, 2004.
[8] D. Lin and R. Morris, Dynamics of random early detection, *Proc. of the ACM SIGCOMM*, pp.127-137, 1997.
[9] F. M. Anjum and L. Tassiulas, Fair bandwidth sharing among adaptive and non-adaptive flows in the Internet, *Proc. of the IEEE INFOCOM*, pp.1412-1420, 1999.
[10] I. Stoica, S. Shenker and H. Zhang, Core-stateless fair queueing: A scalable architecture to approximate fair bandwidth allocations in high-speed networks, *IEEE/ACM Trans. Networking*, vol.11, no.1, pp.33-46, 2003.
[11] P. Wang and D. L. Mills, A probabilistic approach for achieving fair bandwidth in CSFQ, *Proc. of the IEEE NCA*, pp.59-66, 2005.
[12] Z. Cao, Z. Wang and E. Zegura, Rainbow fair queueing: Fair bandwidth sharing without per-flow state, *Proc. of the IEEE INFOCOM*, pp.922-931, 2000.

[13] R. Pan, B. Prabhakar and K. Psounis, CHOKEs: A stateless active queue management scheme for approximating fair bandwidth allocation, *Proc. of the IEEE INFOCOM*, pp.942-951, 2000.
[14] T. Kurimoto, T. Shimizu and R. Kawamura, Core-stateless RED algorithm for improving fairness in a best-effort network, *IEICE Trans. COMMUN.*, vol.E84-B, no.5, pp.1413-1421, 2001.
[15] J. Zheng, M. Hu and L. Zhao, Enhancing Internet robustness against malicious flows using active queue management, *Proc. of the CESS*, pp.501-506, 2005.
[16] I. Stoica, H. Zhang and S. Shenker, Self-verifying CSFQ, *Proc. of the IEEE INFOCOM*, pp.21-30, 2002.
[17] A. Kuzmanovic and E. W. Knightly, Low-rate TCP-targeted denial of service attacks and counter strategies, *IEEE/ACM Trans. Networking*, vol.14, no.4, pp.683-696, 2006.
[18] R. G. Garroppo, S. Giordano, D. Iacono and L. Tavanti, A radio-aware worst-case fair weighted fair queuing scheduler for WiMAX networks, *International Journal of Communication Systems*, vol.27, no.1, pp.13-30, 2014.
[19] L. Xue, S. Kumar, C. Cui and S.-J. Park, A study of fairness among heterogeneous TCP variants over 10 Gbps high-speed optical networks, *Optical Switching and Networking*, vol.13, pp.124-134, 2014.
[20] J.-P. Yang, Adaptive filtering queueing for improving fairness, *Applied Sciences*, vol.5, no.2, pp.122-135, 2015.
[21] J.-P. Yang and Y.-C. Zhuang, Dynamic audit fair queueing, *Proc. of the IEEE ISCIT*, pp.1319-1324, 2007.