ISSN 2350-1022

International Journal of Recent Research in Mathematics Computer Science and Information Technology
Vol. 2, Issue 1, pp: (230-235), Month: April 2015 – September 2015, Available at: www.paperpublications.org

Ephemeral Performance of Choke

Suhitha K.C

M P Nachimuthu M Jaganathan Engineering College, Chennimalai, Erode, India

Abstract: **CHOKe is a simple and stateless active queue management (AQM) scheme. Highly attractive property of Choke is that it can protect responsive TCP flows from unresponsive UDP flows. Packets currently queued in buffer, to penalize the high bandwidth flows. It can be implemented by using RED algorithm. In RED algorithm when packets arrives at a congested router. CHOKe draws a packet at a random from the FIFO buffer and compares it with the arriving packet. If both belong to same flow, then they are both dropped; else randomly chosen packet is left intact and arriving packet is admitted into the buffer with a probability that depend on the level of congestion. These algorithms are typically implemented in the transport protocols (e.g., TCP) of end-hosts. To ensure global fairness, such schemes require all users to adopt them and respond to network congestion properly.**

Keywords: **CHOKe, Random Early Detection (RED), Congestion.**

1. INTRODUCTION

CHOKe is a simple and stateless active queue management (AQM) scheme. Apart from low operational overhead, a highly attractive property of CHOKe is that it can protect responsive TCP flows from unresponsive UDP flows. Choke is able to bound both bandwidth share and buffer share a possible aggregate of UDP Traffic flow on a link. When packets arrives at a congested router. CHOKe draws a packet at a random from the FIFO buffer and compares it with the arriving packet. If both belong to same flow, then they are both dropped; else randomly chosen packet is left intact and arriving packet is admitted into the buffer with a probability that depend on the level of congestion.

2. ADAPTIVE RED

The RED active queue management algorithm allows network operators to simultaneously achieve high throughput and low average delay. The resulting average queue length is quite sensitive to the level of congestion and to the RED parameter settings, and is therefore not predictable in advance. Delay being a major component of the quality of service delivered to their customers, network operators would naturally like to have a rough a priori estimate of the average delays in their congested routers; to achieve such predictable average delays with RED would require constant tuning of the parameters to adjust to current traffic conditions.

Goal of paper is to solve this problem with minimal changes to the overall RED algorithm. We make several algorithmic modifications to this proposal, while leaving the basic idea intact, and then evaluate its performance using simulation. We find that this revised version of Adaptive RED, which can be implemented as a simple extension within RED routers, removes the sensitivity to parameters that affect RED's performance and can reliably achieve a specified target average queue length in a wide variety of traffic scenarios. Based on extensive simulations, we believe that Adaptive RED is sufficiently robust for deployment in routers.

This paper presents Random Early Detection (RED) gateways for congestion avoidance in packet-switched networks. The gateway detects incipient congestion by computing the average queue size. The gateway could notify connections of congestion either by dropping packets arriving at the gateway or by setting a bit in packet headers.

When the average queue size exceeds a preset threshold, the gateway drops or marks each arriving packet with a certain probability, where the exact probability is a function of the average queue size.

RED gateways keep the average queue size low while allowing occasional bursts of packets in the queue. During congestion, the probability that the gateway notifies a particular connection to reduce its window is roughly proportional to that connection's share of the bandwidth through the gateway. RED gateways are designed to accompany a transport-layer congestion control protocol such as TCP. The RED gateway has no bias against burst traffic and avoids the global synchronization of many connections decreasing their window at the same time. Simulations of a TCP/IP network are used to illustrate the performance of RED gateways

3. WHY THE INTERNET TRAFFIC IS BURST IN SHORT TIME SCALES?

Internet exhibits multifaceted business and correlation structure over a wide span of time scales. This is focus on shorter scales, typically less than 100-1000 milliseconds. Our objective is to identify the actual mechanisms that are responsible for creating busty traffic in those scales. We show that TCP self-clocking, joint with queuing in the network, can shape the packet inter arrivals of a TCP connection in a two-level ON-OFF pattern. This structure creates strong correlations and burstiness in time scales that extend up to the Round- Trip Time (RTT) of the connection. This is more important for bulk transfers that have a large bandwidth-delay product relative to their window size. Also, the aggregation of many flows, without rescaling their packet inter arrivals, does not converge to a Poisson stream, as one might expect from classical superposition results. Instead, the burstiness in those scales can be significantly reduced by TCP pacing. In particular, we focus on the importance of the minimum pacing timer, and show that a 10-millisecond timer would be too coarse for removing short-scale burstiness, while A 1-millisecond timer would be clients to make the almost as smooth as a Poisson stream in sub-RTT scales

To achieve satisfying user experiences of diverse applications, quality of service (QoS) guaranteed mechanisms such as per-flow queuing are required in routers. Deployment of per-flow queuing in high-speed routers is considered as a great challenge since its industrial brute-force implementation is not scalable with the increase of the number of flows. In this study, the authors propose a dynamic queues haring (DQS) mechanism to enable scalable per-flow queuing. DQS keeps isolation of each concurrent active flow by sharing a small number of queues instead of maintaining a dedicated queue for each in-progress flow, which is novel compared to the existing methods. According to DQS, a physical queue is created and assigned to an active flow upon the arrival of its first packet, and is destroyed upon the departure of the last packet in the queue. The authors combine hash method with binary sorting tree to construct and manage the dynamic mapping between active flows and physical queues, which significantly reduce the number of required physical queues from millions to hundreds and makes per-flow queuing feasible for high-performance routers.

Per-flow queuing mechanism suffers a great scalability problem with the dramatic increase in the number of in-progress flows. In this paper, we first confirm the measurement result that the number of concurrent active flows in the routers is typically in hundreds even though there may be tens of thousands of flows in-progress. Based on this observation, we have proposed a DQS mechanism to implement scalable per-flow queuing in high-performance routers.

4. ISSUES AND TRENDS IN ROUTER DESIGN

Routers knit together the constituent networks of the global Internet, creating the illusion of a unified whole. While their primary role is to transfer packets from a set of input links to a set of output links, they must also deal with heterogeneous link technologies, provide scheduling support for differential service, and participate in Complex distributed algorithms to generate globally coherent routing tables. These demands, along with an insatiable need for bandwidth in the Internet, complicate their design. Routers are found at every level in the Internet. Routers in access networks allow homes and small businesses to connect to an Internet service provider (ISP). Routers in enterprise networks link tens of thousands of computers within a campus or an enterprise. Routers in the backbone are not usually directly accessible to end systems. Instead, they link together ISPs and enterprise networks with long distance trunks. The rapid growth of the Internet has created different challenges for routers in backbone, enterprise, and access networks. The backbone needs routers capable of routing at high speeds on a few links. Enterprise routers should have low cost per port and a large number of ports, be

easy to configure, and support quality of service (QoS). Finally, access routers should support many heterogeneous high-speed ports and a variety of protocols at each port, and try to bypass the central office voice switch.

This article presents the design issues and trends that arise in these three classes of routers. The following section describes the structure of a generic router. The section after that discusses design issues in backbone, enterprise, and access routers. Present some recent advances and trends in router design. Finally conclude with a description of some open problems. Note that our main topic of discussion is packet forwarding; routing protocols, which create the forwarding tables, are dealt with only in passing

A recently proposed active queue management, CHOKe, aims to protect TCP from UDP flows. Simulations have shown that as UDP rate increases, its bandwidth share initially rises but eventually drops. We derive an approximate model of CHOKe and show that, provided the number of TCP flows is large, the UDP bandwidth share peaks at(e+1)1 = 0:269 when the UDP input rate is slightly larger than the link capacity, and drops to zero as UDP input rate tends to infinity, regardless of the TCP algorithm.

TCP is believed to be largely responsible for preventing congestion collapse while Internet has undergone dramatic growth in the last decade. Indeed, numerous measurements have consistently shown that more than 90% of traffic on the current Internet is still TCP packets, which, fortunately, are congestion controlled. Without a proper incentive structure, this state is fragile and can be disrupted by the growing number of non-rate-adaptive (e.g., UDP-based) applications that can monopolize network bandwidth to the detriment of rate-adaptive applications. This has motivated several active queue management schemes.

5. MAXIMUM AND ASYMPTOTIC UDP THROUGHPUT UNDER CHOKE

A recently proposed active queue management, Choke, is stateless, simple to implement, yet surprisingly effective in protecting TCP from UDP flows. We present an equilibrium model of TCP/CHOKe. We prove that, provided the number of TCP flow is large, the UDP bandwidth share peaks at (+ 1) 1 = 0 269 when UDP input rate is slightly larger than link capacity, and drops to zero as UDP input rate tends to infinity. We clarify the spatial characteristics of the leaky buffer under CHOKe that produce this throughput behavior. Specifically, we prove that, as UDP input rate increases, even though the total number of UDP packets in the queue increases, their spatial distribution becomes more and more concentrated near the tail of the queue, and drops rapidly to zero toward the head of the queue. In stark contrast to a non leaky FIFO buffer where UDP bandwidth shares would approach 1 as its input rate increases without bound, under CHOKe, UDP simultaneously maintains a large number of packets in the queue and receives a vanishingly small bandwidth share, the mechanism through which CHOKe protects TCP flows.

TCP is believed to be largely responsible for preventing congestion collapse while the Internet has undergone dramatic growth in the last decade. Indeed, numerous measurements have consistently shown that more than 90% of the traffic on the current Internet is still TCP packets, which, fortunately, are congestion controlled

6. APPROXIMATE FAIRNESS THROUGH LIMITED FLOW LIST.

Most of router mechanisms proposed for fair bandwidth sharing lack either (1) simplicity due to complexity of intricate per flow management of all connections (e.g., WFQ, SFQ), (2) heterogeneity due to a design targeting a specific traffic type, e.g., RED-PD and Fair RED (FRED) or (3) robustness due to requirement for proper router configurations (e.g., CSFQ). All of these severely impact the scalability of the schemes. This paper proposes a novel router fairness mechanism, namely Approximate Fairness through Partial Finish Time (AFpFT). Key to the design of AFpFT is a tag field the value of which defines the position of the packet in an aggregate queue shared by all flows. The specific of tag computation depends on the router's role—edge or inner— to the flow. While gateways closest to traffic source manage all flows, successive or inner routers only manage a limited subset at flow level. The managed flows are usually of higher rates than fair share. Following the heavy-tailed Internet flow distribution; these flows are indeed the minority in the Internet. Using extensive simulations, we show that the scheme is highly fair and potentially scalable unlike other proposed schemes.

ISSN 2350-1022

International Journal of Recent Research in Mathematics Computer Science and Information Technology
Vol. 2, Issue 1, pp: (230-235), Month: April 2015 – September 2015, Available at: www.paperpublications.org

Designing core-stateless versions of fair PFFQ is not an easy task. The main hurdle is that computation of flow parameters is dependent on other interacting flows and it is impossible to determine these parameters at network edges. The goal of AFpFT is to approximate the fairness of per flow fair queuing algorithms with minimum states possible.

There are two distinct but complementary ways to enforce flow fairness and protection in the Internet. The end-to-end architectural design principle of the Internet, the more classical way has been via congestion control algorithms .These algorithms are typically implemented in the transport protocols (e.g., TCP) of end-hosts. To ensure *global* fairness, such schemes require all users to adopt them and respond to network congestion properly. This requirement can hardly be met for at least two reasons. First, there is no performance incentive to end-users. This is because users who lack the congestion control algorithms, intentionally or otherwise, may end up with a lion share of bandwidth. Second, in order to meet real-time requirements, many applications do not implement congestion control. Hence, to protect responsive (e.g., TCP) flows from unresponsive (e.g., UDP) ones.

CHOKe is a simple and stateless active queue management (AQM) scheme. A highly attractive property of CHOKe is that it can protect responsive TCP flows from unresponsive UDP flows. Choke is able to bound both bandwidth share and buffer share a possible aggregate of UDP Traffic flow on a link.

When packets arrives at a congested router. CHOKe draws a packet at a random from the FIFO buffer and compares it with the arriving packet. If both belong to same flow, then they are both dropped; else randomly chosen packet is left intact and arriving packet is admitted into the buffer with a probability that depend on the level of congestion.

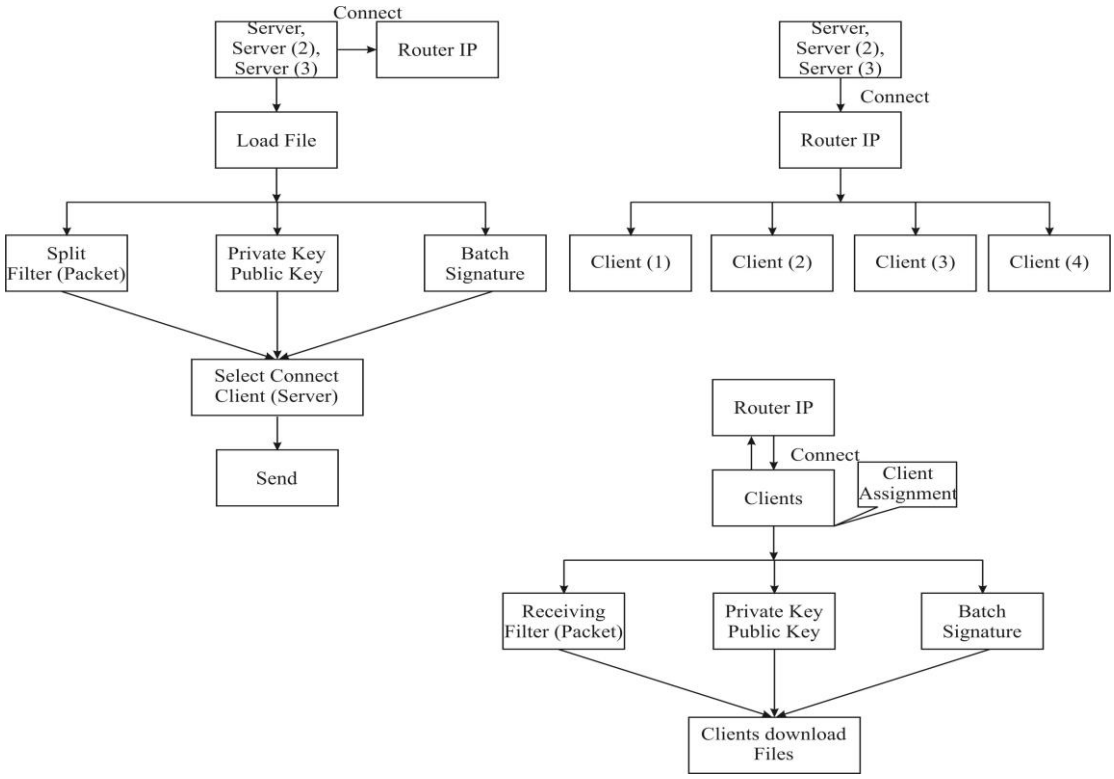

Fig 1 data flow diagram

7. CONCLUSION

This paper provides the first study on CHOKe behavior in the aftermath of rate changes in UDP traffic arrival. In particular, we are concerned with CHOKe queue behaviors during the transient regime, which we model as a transition from one steady queue state to another. We found that the performance limits stipulated in the steady state rarely hold for such transient regimes. Depending on the nature of rate change, the queue exhibits instant fluctuations of UDP bandwidth sharing in reverse direction. This behavior has ramifications on the smooth operation of the Internet where most flows are rate-adaptive. Such flows may see fluctuating available link bandwidth and degrade in performance. By extending and

leveraging the spatial distribution model, this paper analytically: 1) determines the extreme points of UDP utilization (observed within an order of queuing delay after rate change); and 2) tracks the evolution of the transient UDP utilization following rate change. In addition, the model allows us to obtain generic UDP utilization plots that help explain both the transient extreme characteristics and steady-state characteristics. The analytic results have been rigorously validated through extensive simulations. We believe the analytical approach used in this paper also sheds light on studying transient behaviors of other leaky queues.

The analysis is independent of the specific flavor of TCP window control algorithm. The only condition placed upon TCP flows is that they collectively are able to grab the remaining bandwidth left by the UDP flow. This can easily be fulfilled by most TCP algorithms, making the analysis and results applicable to a wide variety of TCP algorithms. Recall from the analysis that the transient behavior is mainly controlled by the UDP source rate and factor of rate change. Therefore, it is plausible to formulate the analysis on a rate-based model, rather than window-based. The latter is commonly used for investigating (microscopic) TCP behavior. The TCP flows are of secondary importance as they merely take up the bandwidth left over by the UDP flow. As a result, detailed modeling of TCP window control algorithms seems orthogonal to this work. However, window-based analysis may be suitable in other situations where the TCP itself is the primary control (or parameter of interest) or where the network behavior is controlled by multiple parameters.

AKNOWLEDEMENT

I am heartily grateful to my Head of the Department Mr. K.N SIVAKUMAR M.E., Associate Professor of Computer Science and Engineering, for his constructive suggestions and support.With immense pleasure I regard my deep sense of indebtedness and gratitude to the Mrs. S.DEVISRI, M.E., Assistant Professor of Department of Computer Science and Engineering, who gave her guidance and support throughout my work and made this as a successful project.

REFERENCES

[1] S. Floyd, R. Gummadi, and S. Shenker, “Adaptive RED: An algorithm for increasing the robustness of RED’s active queue management,” Tech. Rep., 2001.

[2] S. Floyd and V. Jacobson, “Random early detection gateways for congestion avoidance,” IEEE/ACM Trans. Netw., vol. 1, no. 4, pp. 397–413, Aug. 1993

[3] J. H. Saltzer, D. P. Reed, and D. D. Clark, “End-to-end arguments in system design,” Trans. Computer. Syst., vol. 2, no. 4, pp. 277–288, 1984.

[4] H. Jiang and C. Dovrolis, “Why is the Internet traffic bursty in short time scales?,” in Proc. ACM SIGMETRICS, 2005, pp. 241–252.

[5] Hu, Y. Tang, X. Chen, and B. Liu, “Per-flow queueing by dynamic queue sharing,” in Proc. IEEE INFOCOM, 2007, pp. 1613–1621

[6] M. Allman, V. Paxson, and. Blanton, “TCP congestion control,” RFC5681, Sep. 2009.

[7] Demers, S. Keshav, and S. Shenker, “Analysis and simulation of fair queuing algorithm,” in Proc. ACM/SIGCOMM, 1989, pp. 1–12..

[8] J. Wang, A. Tang, and S. H. Low, “Maximum and asymptotic UDP throughput under CHOKe,” in Proc. ACM SIGMETRICS, 2003, pp. 82–90

[9] Tang, J.Wang, and S.H. Low, “Understanding CHOKe: Throughput and spatial characteristics,” IEEE/ACMTrans. Netw., vol. 12, no. 4, pp. 694–707, Aug. 2004.

[10] Eshete and Y. Jiang, “Approximate fairness through limited flow list,” in Proc. Int. Teletraffic Cong., 2011, pp. 198–205.

[11] S. Keshav, An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network. Reading,MA,USA: Addison-Wesley-Longman, 1997.

[12] Kortebi, L. Muscariello, S. Oueslati, and J. Roberts, "Evaluatingthe number of active flows in a scheduler realizing fair statistical bandwidth sharing," in Proc. ACM/SIGMETRICS, 2005, pp. 217–228.

[13] Suter, T. Lakshman, D. Stiliadis, and A. Choudhury, "Buffer management schemes for supporting TCP in gigabit routers with per-flow queueing," IEEE J. Sel. Areas Commun., vol. 17, no. 6, pp. 1159–1169, Jun. 1999.

[14] Lin and R. Morris, "Dynamics of random early detection," Comput.Commun. Rev., vol. 27, no. 4, pp. 127–137, 1997.

[15] R. Mahajan, S. Floyd, and D. Wetherall, "Controlling high-bandwidth flows at the congested router," in Proc. IEEE ICNP, 2001, pp. 192–201.

[16] R. Pan, B. Prabhakar, and K. Psounis, "CHOKe—A stateless active queue management scheme for approximating fair bandwidth allocation,"in Proc. IEEE INFOCOM, 2000, pp. 942–951.

[17] R. Pan, C. Nair, B. Yang, and B. Prabhakar, "Packet dropping schemes, some examples and analysis," in Proc. Allerton Conf. Commun. , Contro, Comput., 2001, pp. 563–572.

[18] F. Baccelli and D. Hong, "AIMD, fairness and fractal scaling of TCP traffic," in Proc. IEEE INFOCOM, 2002, pp. 229–238.

[19] M. E. Crovella and A. Bestavros, "Self-similarity in World Wide Web traffic: Evidence and possible causes," in Proc. ACM SIGMETRICS, 1996, pp. 160–169.

[20] Eshete and Y. Jiang, "Generalizing the choke flow protection," Comput. Netw., vol. 57, no. 1, pp. 147–161, 2013.

[21] Tang, L. L. H. Andrew, K. Jacobsson, K. H. Johansson, H. Hjalmarsson, and S. H.Low, "Queue dynamics with window flow control," IEEE/ACM Trans. Netw., vol. 18,no. 5, pp. 1422–1435, Oct. 2010.

[22] R. Shorten, F.Wirth, and D. Leith, "A positive systems model of TCP like congestion control: asymptotic results," IEEE/ACM Trans. Netw., vol. 14, no. 3, pp. 616–629, Jun.2006.

[23] Tang, L. L. H. Andrew, K. Jacobsson, K. H. Johansson, S. H. Low ,And H.Hjalmarsson, "Window flow control: Macroscopic properties From microscopic factors," in Proc. IEEE INFOCOM, 2008, pp. 91–95.