MxCHOKe: To Choke Anomalous Traffic with a Mix Matching Algorithm of Flow ID and IP for Various Network Congestion Avoidance

HAIBO TAN, XIAOFENG LI
Network and Information Center
Hefei Institutes of Physical Science, Chinese Academy of Science
350 Shushanhu Road, Hefei City
CHINA
hbtan@hfcas.ac.cn http://english.hf.cas.cn

Abstract: - Active Queue Management (AQM) plays an important role in network congestion control. Most of AQM algorithms aim to achieve flow fairness. However, with an investigation to congestion-cause traffic, the anomalous packets vary in different aggregating methods, not just within limited high-bandwidth flows. Therefore the previous AQM would lost the ground to penalize the misbehaved traffic scattered in distributed flows. In this paper, an escalated and mix matching of flow ID and IP algorithm (named MxCHOKe) has been proposed for various network congestion avoidance based on their statistical behavior. Simulation results suggest it gained an expected performance of protecting well-behaved traffic in more realistic congested situations.

Key-Words: - Active Queue Management, Anomalous Traffic, Congestion Control, Approximate Fairness

1 Introduction

In the IP networks, burst of traffic pass through an asymmetric path often leads to a network congestion. Active Queue Management (AQM) has been proposed to complement with the end-system protocols such as TCP to alleviate the situation and improve the end-user experience with an optimal state (low delay, high utilization) especially at the bottleneck.

The first AQM scheme is Random Early Detection (RED) proposed by Floyd et al. [1], which followed is a massive growth of RED variants and some new schemes. Among those schemes respect to fair or differ-service bandwidth allocation, there are three types of AQM: (1) AQM with no per-flow information, (2) AQM with per-flow information and (3) AQM with per-flow scheduling, according to [3]. The last will achieve the best result, but will be at the most complex.

As a typical scheme for approximating fair bandwidth allocation with the lowest cost and easy to be adopted, CHOKe has been proposed by Pan et al. [4]. It amended the shortcoming of RED as unable to penalize high-bandwidth unresponsive flows, with the benefit of no flow state information stored or processed.

The mechanism of CHOKe is based on the common recognition that during the congestion, the unresponsive flows will have a high buffer occupancy and a high incoming rate. The simulations and the model analytical from [4] has also suggested it works well in protecting congestion-sensitive flows from several congestion-cause flows. Motivated by improving the performance of CHOKe, a group of derivations has been proposed. Among them, xCHOKe [5] performs better than CHOKe by keeping partial state, with small per-packet computation. Aimed with improve the ability of detecting and control the malicious flows with bandwidths smaller than the link capacity, RECHOKe [6] adopted a method by combining and using the CHOKe hit and CHOKe-RED drop/mark histories. Even more, some new algorithms like LRURC [7] begin to employ a new separate virtual queue and rate check module to pursuit high level of flow fairness.

All of those variants inherit the sample-match mechanism from CHOKe which focused on the flow ID comparing solely. A typical flow ID is a hash of 5-tuple (IPsrc, IPdst, Portsrc, Portdst, Protocol). However the anomalous traffic during congestion varied from aggregating in flow ID to IP etc., not just limited in several high-bandwidth flows. For example, under the extreme situation, every packet generated by a malicious host would be treated as a separated flow from the view of CHOKe. As no incoming packet has the same flow ID with the packets inside the queue, no packets would be dropped by match mechanism, except random packets by the RED drop possibility or the DropTail mechanism to hold the queue length. In this case, the responsive traffic will be squeezed to a very limit link capacity, and all of the end users will suffer a high data loss ratio.

Considered about the great advantage of CHOKe, in this paper, we present an extended CHOKe, named as MxCHOKe (Mix CHOKe) by using a mix matching of flow ID and IP algorithm to alleviate the panic. There were three main contributions of this paper. First, with investigation toward the anomalous traffic, we present MxCHOKe as a solution. Second, simulations has been made which suggest that MxCHOKe achieved an expected results in various congested situations. Furthermore, we create a simple analytical model to understand and reinforce the MxCHOKe under a more complicated situation.

2 MxCHOKe Algorithm

In this section, subspace method has been used to classify the anomalous traffic during congestion. With a suitable classification, we present the detail of escalated mix-match algorithm.

2.1 Characterization of Anomalous Traffic

With an investigation to the anomalous traffic [5], it spans a remarkably wide spectrum of event types, including DoS (Denial of Service) attacks, flash crowds, port scanning, downstream traffic engineering, high-rate flows, worm propagation etc. The only way to precisely identify them required a thoroughly analysis at application level and is beyond the ability of a stateless queue schedule. However, according to [5], the traffic statistics could be counted at the router for identification. Considering about the packet fast accumulating in queue during congestion, it is easy to use sample-match method to drop them with the different subspace of flow ID.

Therefore, we classify the anomalous traffic during congestion into 4 categories and present some typical scenarios for understanding in the table 1.

2.2 Model Analysis

From the table 1, there is a descending order in the dimensions, followed with the relationships of them:

$$\begin{cases} R_2 = \Pi_{\mathrm{IPsrc,IPdst}}(R_1) \\ R_3 = \Pi_{\mathrm{IPsrc}}(R_2) \\ R_4 = \Pi_{\mathrm{IPdst}}(R_2) \end{cases}$$

The first row of the table is the match criteria adopted by CHOKe, relies on a full match of 5-tuple, then comes with a subspace R2, the latter two are just the symmetric subspace match of R2, we will treat them together.

It is obviously that to choose the last match criteria in the first begining, will effectively reduce the tension of congestion. However, it will lead to an abused drop toward the well-behaved sessions. Considered about sample-match by R3 before R1 toward an unresponsive flow sender behind a NAT, all of the other clients will also be harmed.

TABLE 1
Anomalous Traffic Classification

Anomaly Statistical Feature	Characteristic	Examples
High volume data within the same flow pipe	R1 (IP_{src}, IP_{dst}, $Port_{src}$, $Port_{dst}$, Protocol)	an abrupt bandwidth measurement test or selfish UDP flow etc.
High volume data within the same IP pair	R2 (IP_{src}, IP_{dst})	A typical DoS with varied source port of the malicious IP ,or a high speed ports scan etc.
High volume data with the same source IP	R3 (IP_{src})	A network scan of worm or a misbehaved p2p file sharing etc.
High volume data with the same destination IP	R4 (IP_{dst})	A typical DDOS to the victim etc.

R1, R2, R3, R4 are all space symbols, with the n-ordered tuples in the parentheses.

An escalated drop mechanism has been introduced in MxCHOKe based on the mix match of flow ID and IP by dividing the queue occupy space from min_{th} to max_{th} of RED[1] to 3 fairly equal sub-spaces with two intermediate threshold $int1_{th}$ and $int2_{th}$, in order to avoid biases.

Fig.1 shows the flowchart of escalated drop mechanism. The average occupancy of queue (avg_q) and the admit probability of new coming ($p_a = 1-p_b$) has been calculated at the same way of RED [1]. From the flowchart, a new arrived packet will be accepted directly when avg_q less than the min_{th}, or to be compared differentially with flow ID, IP pair (source and destination addresses), IP (source or destination address) based on the tension extent of the queue. What's more, comparison and drop the same still being carried out to accelerate emptying queue, when avg_q is larger than max_{th}.

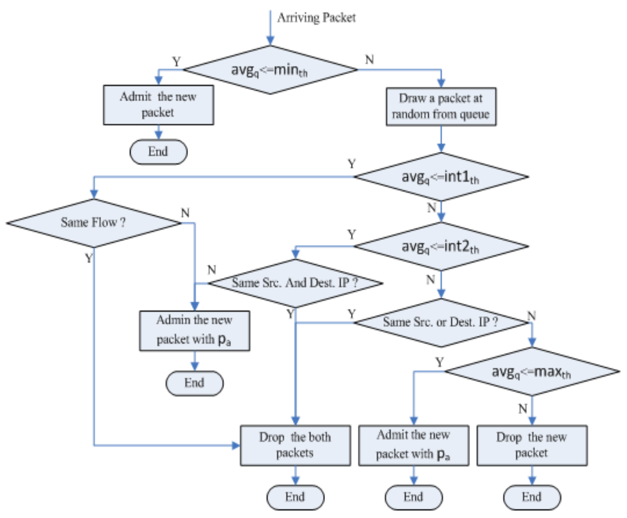


Fig. 1. The MxCHOKe algorithm

3 SIMULATION AND ANALYSIS

The environment is established on the canonical discrete event simulator NS-2 [9]. We extended its UDP module to generate three type CBR (constant bit rate) traffic named as "SELFISH-UDP" (behaved like row 1, Table 1), "DOS-UDP" (behaved like row 2, Table 2) and "WORM-UDP" (behaved like row 3&4, Table 3) for easy to understand.

Though the anomaly could be formed of TCP and other protocols and the worm will bring a complex propagation traffic in the real world. As the algorithm works with protocol independent and we just focus on one-way traffic suppression of AQM, the simulations can represent the behavior of traffic in congestion. Meanwhile, the original CHOKe has been selected for comparing, as all its variants are also flow-fairness respected only.

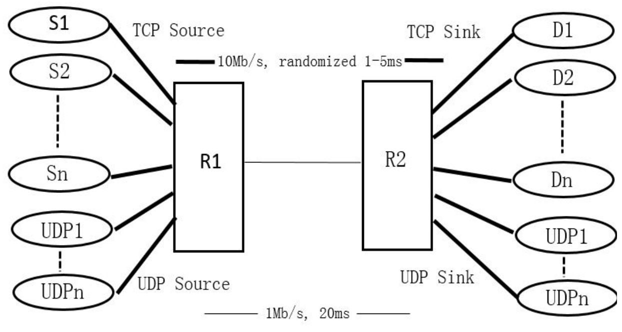


Fig. 2. Network Topology

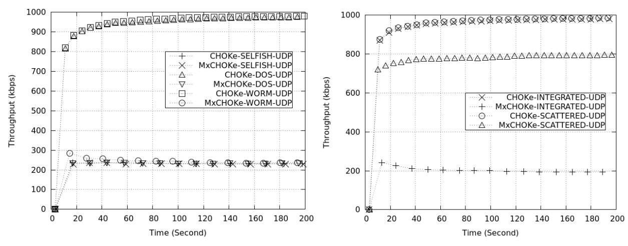
3.1 Simple Route Comparison

We use network topology in Fig.2, the typical value of min_{th}, max_{th} and queue length are set with 100,200,300 packets. The compared schemes are deployed in R1-R2 whereas DropTail for the other queues. There are 32 well behaved TCP sessions with n UDP sources according to the different test scenarios. All of the tests last for 200s, and the TCP nodes start from 0.2s, the UDP nodes start from 2s.

In this section, the test will be carried out in three steps. At first step, the three type of anomalous traffic will be generated respectively with a fixed rate at 2Mb/s from one UDP source to gauge the performance of the compared algorithms. Second, we use the UDP source to generate an integrated traffic of all the three types, and start them simultaneously with a fixed rate at 1Mb/s of each. The total traffic is 3Mb/s, triple the bottle link in this case. Third, we use three UDP sources to generate the traffic separately to simulate the scattered anomalous traffic.

The traffic has been measured at the receiving side, and the throughput results of anomalies are shown in the Fig. 3. The queue stability has also been recorded, MxCHOKe has a similar performance and hold the queue length at expected value in handling different anomalies, whereas CHOKe can only have effect on the SELFISH-UDP. Due to page limit, it is not be listed in this script.

From Fig. 3(a), it is clear that MxCHOKe inherited the ability of suppressing the SELFISH-UDP from CHOKe, it also successfully penalized the DOS-UDP, WORM-UDP traffic, whereas CHOKe failed to. Also the performance descends slightly in handling the different kind of traffic, it could be understood from the view of an escalated match



(a) Individual (b) Hybrid

Fig. 3. Anomalous Throughput Comparison

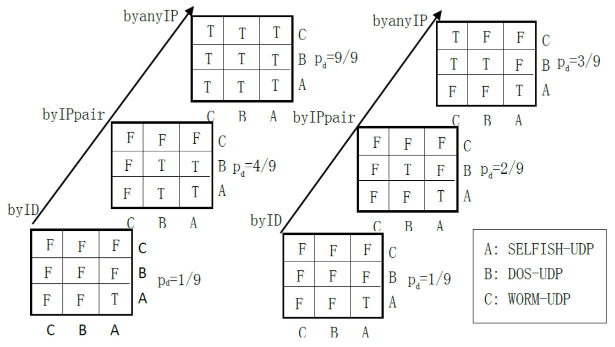
threshold of MxCHOKe.

From Fig. 3(b), we can notice that anomaly occupied almost total link capacity in both situations with CHOKe. As for MxCHOKe, it varied based on different situations, it suppressed the aggregated UDP traffic effectively, but with the scattered UDP traffic, the drop ratio decreased from 93.60% to 73.67%(calculated from the trace file), the total bandwidth of TCP sessions has also been limited to about 200Kb/s.

According to Tang et al. [10], a set of 10 nonlinear equations has been proposed to explicitly model the throughput and spatial characteristics of CHOKe. It is possible to extend the method and the numerical solutions to depict the MxCHOKe's differential behaviour. However, aim to a simplicity and approximate explanation, we introduce two 3-division truth tables shown in Fig.4 to analyse the results.

In Fig.4, we use "p_d" to represent the average sample-match ratio of total anomalous traffic, "byID", "byIPpair", "byanyIP" represent the different match condition in proportion to the tension of queue. The row and column indicates the packet in queue and new arrived separately.

The p_d is calculated based on the following hypotheses: since the anomalies have same high arriving bandwidth, they would gain an equal share of the whole queue and arriving pipe. Then the random selection in queue and arriving possibilities for them are the same value near 1/3. Second, though p_d will be affected by many factors and the



(a) Integrated Situation (b) Scattered Situation
Fig.4. Truth Tables of MxCHOKe Drop Possibilities

TCP sessions will have a certain share, with fixed input, the system will achieve a dynamic balance, we could use the approximate p_d to investigate the trend of MxCHOKe behaviour.

Then with Fig.4, it is easy to understand the test results of CHOKe, as only "byID" has been used, the sample-match possibility for "A" is around 1/9, only 1/9*1/3 $\approx$ 3.7% of total packets are dropped by CHOKe, hence it lost the ability to protect the responsive traffic in both integrated and scattered situations.

For MxCHOKe, in Fig.4, almost every (9/9) anomalous packet of integrated UDP could be dropped under the extreme condition (as all packets originated from same source IP), compared with only 3/9 of scattered UDP could be dropped. It just explained the MxCHOKe behavior in Fig.3 (b). Moreover, supposing same amount of candidate packets passed to the three sample-match phases, the drop possibility ratio of each type will be: p_{dA} : p_{dB} : p_{dC} = (1/9+2/9+3/9) : (2/9+3/9) : (3/9) $\approx$ 2 : 1.67 : 1 with integrated UDP and 3:2:1 with scattered UDP. The ratio is just a trend reflection of escalated drop policy, it will vary according to the tension of queue.

3.2 Improvement for MxCHOKe

As the different type of anomalous traffic often sourced from different hosts. To improve the performance under the scattered UDP, motivated by CHOKe, an escalated multi-selections has been introduced. According to Fig. 4(b), if we multiple the random selection packet number with 1, 2, 3 respectively in different phase, the average drop ratio will arise to the similar value of Fig. 4(a)

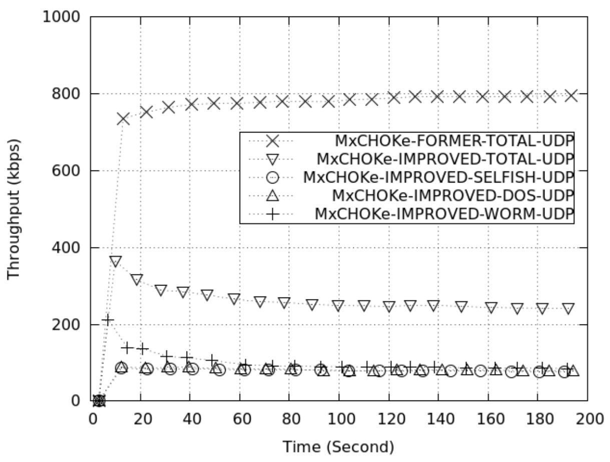


Fig. 5. Throughput Comparison of Improvement

theoretically. The test results are shown in Fig.5.

From Fig.5, with the improvement, total anomalies has been penalized from 795Kb/s to 240Kb/s, meanwhile the 32 well behaved TCP sessions gain almost 76% share of the total link. Inspired with the great advance, it is reasonable to endorse the improved MxCHOKe in more complicated

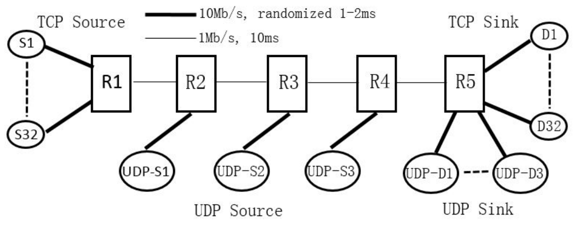
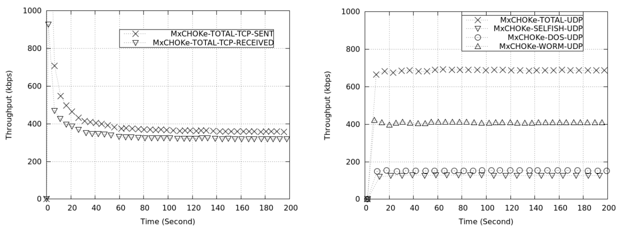


Fig. 6. Topology of Multi-Links

situations.

3.3 Multiple Congested Links

In this section we study the performance of improved MxCHOKe under multi-congestion links. The anomalous traffic generators (UDP-S1 for "SELFISH-UDP", UDP-S2 for "DOS-UDP" and UDP-S3 for "WORM-UDP") are placed along the routing path, shown in Fig.6, with a fixed rate at 2Mb/s. The 32 pair well behaved nodes are placed at the far ends for the longest test. All bottle links are equipped with MxCHOKe, with the same parameters as previous tests.



(a)Responsive TCP (b) Anomalous UDP

Fig. 7. Multi-Links Throughput Comparison

The throughput has been measured, and the total TCP sending rate at R1 has been compared with the receiving at R5 in Fig.7 (a), the three type of UDP and the total are also shown in the Fig.7 (b).

From Fig.7 (a), the self-adaptive flows attain a very steady bandwidth on the long risky journey. Meanwhile, the bandwidth of anomalies is limited to an expected value during the congestion.

From Fig.7 (b), according to the drop possibility and the number of congestion point crossed, the "WORM-UDP" has a higher survival ratio than others. It need be noticed that the name "WORM-UDP" here is just a statistical identification. Hence some normal traffic will also fall into this category, such as a high burst from distributed clients to a centralized upload server.

Furthermore, as we mentioned before, MxCHOKe is protocol independent, therefore the outrageous traffic in other protocols can be suppressed the same way, though we use UDP to simulate them.

4. Conclusion

To summarize, the traditional AQM with respect to flow fairness cannot effectively protect the adaptive flows against the enormous distributed none-adaptive flows or anomalous traffic. In this paper, we classified different types of network congestion-cause traffic based on their statistical behaviour with investigation. Then we present a more realistic AQM algorithm (MxCHOKe) to alleviate the panic of various congestion, aimed to achieve approximate fairness with a minimum cost, and act as a last effort before the queue overflow. Simulation results show that it can effectively penalize multi-types of congestion-cause traffic and achieve an expected fairness in more complexity situation with a similar implementation overhead as CHOKe. Further work involves studying on the adaptability and optimizations of parameters and hardware deployment issues.

References:

[1] S. Floyd and V. Jacobson, "Random early detection gateways for Congestion avoidance," *IEEE/ACM Transactions on Networking*, Vol. 1,No. 4, pp. 397–413, 1993.

[2] R. Adams, "Active Queue Management: A Survey", *IEEE Commun. Surv. and Tut.*, Vol. 15, No. 3, pp. 1425-1476, Jul. 2013.

[3] M. Nabeshima, "Improving the performance of active buffer manage-ment with per-flow information," *IEEE Communications Letters*, Vol. 6,No. 7, pp. 306–8, 2002.

[4] R. Pan, B. Prabhakar, and K. Psounis, "CHOKe: a stateless active queue Management scheme for approximating fair bandwidth allocation," *Proc.of IEEE INFOCOM*, Tel Aviv, Israel, March 2000, Vol. 2, pp. 942–951.

[5] P. Chhabra, A. John, H. Saran, and R. Shorey. "Controlling Malicious Sources at Internet Gateways," *IEEE Int. Conf. Communication,Anchorage*, Alaska, May 2003, vol. 3, pp. 1636-1640.

[6] V. V. Govindaswamy, G. Záruba, and G.Balasekaran, "RECHOKe: A Scheme for Detection, Control and Punishment of Malicious Flows in IP Networks," in Proc. of *IEEE GLOBECOM*, 2007.

[7] Jiang, X. ; Jin, G. ; Yang, J. "LRURC: A Low Complexity and Approximate Fair Active Queue Management Algorithm for Choking Non-adaptive Flows", *Communications Letters*, IEEE Volume: PP , Issue: 99 DOI: 10.1109/LCOMM.2015.2401559 Publication Year: 2015 , Page(s): 1 IEEE EARLY ACCESS ARTICLES

[8] L. Anukool, C. Mark, D.Christophe , "Characterization of network-wide anomalies in traffic flows" Proceedings of the *2004 ACM SIGCOMM Internet Measurement Conference*, IMC 2004, p 201-206.

[9] -, "The Network Simulator version 2", http://www.isi.edu/nsnam/ns/2012.

[10] A Tang, J. Wang, S. H. Low, "Understanding CHOKe: Throughput and Spatial Characteristics", *IEEE/ACM Trans. on Network*, Vol. 12, No. 4, pp.694-707, Apr. 2004.