On Fair Bandwidth Allocation in Connection-less Networks

Babak Behsaz, Pawel Gburzynski, and Mike MacGregor
Department of Computing Science
University of Alberta
Edmonton, Alberta, CANADA T6G 2E8
Email: [behsaz,pawel,macg]@cs.ualberta.ca

***Abstract*—We introduce a transport-independent mechanism for network-wide fair bandwidth allocation in the Internet based on a natural hierarchy of authorities. With our proposed approach, routers identify a dynamic set of so-called fluxes derived form descriptors (ranks) disseminated in the network in a hierarchical fashion. The scheme assumes no particular implementation of the sessions contributing to the fluxes: cooperation is simply in their best interest, as otherwise, their received bandwidth will be below the attainable fair share. We illustrate the behavior of our mechanism in a simulated environment and argue about its potential to solve many QoS problems that traditionally have been associated with call admission.**

I. Introduction

Among its overwhelming advantages, the connection-less paradigm of the Internet is tainted with at least one nasty flaw: the essentially independent datagrams handled by routers are not accountable for their true *users*. In particular, during a congestion, a router would like to identify meaningful recipients of the fair treatment that it may want to extend to the forwarded packets. Traditionally, those recipients have been either explicit transport-layer streams, typically TCP sessions [1], [2], [3], or various *flows* described heuristically by some properties of packets [4], [5].

The idea of viewing TCP sessions as the fundamental flows contributing to network load dates back to the infamous collapse of the Internet in 1988-1989 and its salvage by Jacobson [6]. That approach would have solved the fairness problem in an ideal world where: 1) all TCP implementations are compliant to [6]; 2) users are synonymous with TCP sessions, so that a fair treatment of those sessions translates into a fair treatment of actual users; 3) TCP traffic is all there is. Unfortunately, none of these conditions holds in today's Internet. As the behavior of a TCP session is solely up to the edge host, there is little that a router can do to effectively enforce its compliance [7]. This leaves the system open for various tricks, whereby a non-compliant TCP session can steal bandwidth from its well behaved counterparts. But more importantly, TCP sessions are not synonymous with users (or even applications): an application needing a lot of bandwidth can easily spawn multiple TCP sessions in parallel. Also, all the measures aimed at policing TCP sessions can be circumvented by resorting to UDP and implementing flow control within the application [8], [9].

While routers often strive to identify various types of flows in the forwarded stream of packets, be it for bandwidth policing [4], [5], traffic modeling [10], [11], or detecting network abuse [12], [13], the local (router-based) heuristics used for this purpose unavoidably share the same problem: the inherent inability to authoritatively assess how much of the global bandwidth has been consumed by the actual *users* of those flows. For an extreme example, envision a collusion of hosts within a collaborating group of users, whereby different parts of the same session follow different paths in the network, e.g., as in BitTorrent [14], [15].

Consequently, our proposed scheme of fairness enforcement employs mechanisms at the routers, rather than relying on the transport behavior of endpoint applications. The solution is non-local, in the sense that the routers have to exchange some information in order to arrive at a collectively meaningful identification of network *users*.

II. Fluxes

The "flow" concept that we introduce here is intended to describe an authoritative recipient of network bandwidth from the viewpoint of its fair allocation. To distinguish such *flows* from traditional flows, we shall call them *fluxes*.

Define a flow as the smallest relevant traffic component that can be discerned by a router. Such a flow corresponds to a single transport layer "stream" and is described by the 4-tuple $< S, P_s, D, P_d >$ where S is the source IP address, P_s is the source port, D is the destination IP address, and P_d is the destination port. Let A be the set of all flows passing through a router. By a flux we understand a pair $F = [\Phi, q]$ where $\Phi \subseteq A$ and $q > 0$ is a number. Thus, a flux is a subset of the set of all flows handled by the router. The role of q is to provide a means for implementing intentional bias reflecting the subjective importance of each flux.

A router builds a description of the fluxes passing through it. This description is derived from flux descriptors communicated to the router by its neighbors and its own flux policy. Edge routers receive flux descriptors from hosts. An uninformed "vanilla" router, which has acquired no descriptors from the outside and implements no internal flux policy defines a single flux denoted $[< *, *, *, * >, 1]$. The asterisks serve as wildcards and provide a shorthand notation for all possible values of the respective flow parameters. The default flux description of an "uninitialized" router simply says that all flows that the router ever sees belong to a single flux; thus, there is no regulation of competition for bandwidth among those flows. For the feel of a possible specification, we may assume that fluxes are equivalenced with sets of 5-tuples (as above) where any of the first four items can be wildcards. An incoming flow is qualified to the flux whose description includes the 5-tuple that most closely matches the flow's attributes.

The allocation of bandwidth at a router deals with the router's output links and is carried out on a per-output-link basis. The router identifies the different fluxes competing for a given output link and views them as the contenders that should be treated fairly.

A. Ranking Fluxes

Consider the trivial network shown in Figure 1 and suppose that host H1 defines three fluxes, host H2 two fluxes, and host H3 four fluxes. Router R will build its description of fluxes derived from the information received from the hosts, normalizing the ranks of the inherited fluxes to their population. If R's policy is to treat the three hosts equally, then each flux of H1 will receive the rank of $1/(3 \cdot 3) = 1/9$, each flux of host H2 will receive the rank of $1/(3 \cdot 2) = 1/6$, and each flux of H3 will be ranked $1/(3 \cdot 4) = 1/12$. These numbers directly represent the fraction of the router's global bandwidth that will be assigned to every flux.

The situation is slightly complicated by the fact that the hosts (and more generally neighbors, which can be other routers) of R may pre-rank their fluxes. Let us consider a general case and denote the collection of all fluxes received by a router from its m neighbors as F_i^j, $1 \le i \le m$, $1 \le j \le n_i$, where n_i is the number of fluxes defined by neighbor i. Let q_i^j denote the rank of flux F_i^j as assigned by neighbor i. We shall assume that $\sum_{j=1}^{n_i} q_i^j = 1$, otherwise the router can renormalize the ranks to fulfill this condition.

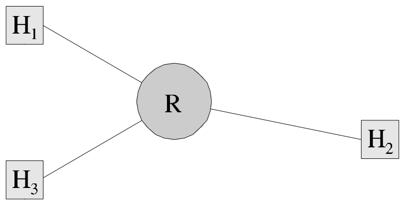

Fig. 1. A trivial network

The router's policy is described by the ranks that the router assigns to its neighbors, denoted r_i, $1 \le i \le m$, such that $\sum_{i=1}^{m} r_i = 1$. The ranks of the incoming fluxes are rescaled with respect to the router's policy ranks as $\rho(F_i^j) = r_i \cdot q_i^j$, $1 \le i \le m$, $1 \le j \le n_i$, where $\rho(F_i^j)$ is the effective policed rank of flux F_i^j at router R. The goal of this procedure is to arrive at a ranking of fluxes that would allow a router to respect the ranks passed to it by other authorities, while superimposing on them its own ranking policy. If a given incoming flux were entirely forwarded over a single output link, it would be easy to propagate the rank information. Specifically, if $\{G_u^k\}$, $1 \le k \le n_u$ is the set of all fluxes sent on link u, the router would pass to the neighbor connected to it via link u the ranks:

$$s_u^k = \frac{\rho(G_u^k)}{\sum_{j=1}^{n_u} \rho(G_u^j)} \tag{1}$$

reflecting the router's effective policed ranks of the respective fluxes. Generally, however, because of its aggregate character, a single incoming flux is not unlikely to be split over multiple output links. Let $\gamma_u(F_i^j, G_u^k)$ denote the normalized fraction of flux F_i^j directed to output link u and indexed as the k'th flux going out over that link. By "normalized" we mean that $\sum_{u=1}^{m} \gamma_u(F_i^j, G_u^k) = 1$ over the m output links at the router. Then:

$$\Psi_u = \{F_i^j | \gamma_u(F_i^j, G_u^k) \neq 0\}$$

is the set of fluxes that feed into output link u. The rank of (output) flux G_u^k communicated by the router to its neighbor over link u is calculated as:

$$\rho(G_u^k) = \frac{\rho(F_i^j) \cdot \gamma_u(F_i^j, G_u^k)}{\sum_{F_i^j \in \Psi_u} \rho(F_i^j) \cdot \gamma_u(F_i^j, G_u^k)} \tag{2}$$

The router determines the value $\gamma_u(F_i^j, G_u^k)$ by monitoring the input flux and measuring its fraction routed over link u. Thus, $\gamma_u(F_i^j, G_u^k)$ is dynamic and will tend to change over time. The dynamic adjustments to the set of fluxes perceived by a router or a host can be propagated via a special protocol, akin to the standard protocols for disseminating routing or QoS information (OSPF [16], BGP [17], RSVP [18]).

B. Identifying Unfair Fluxes

Let x_u^k denote the offered rate of each flux in Ψ_u. That is, $x_u^k = \gamma_u(F_i^j, G_u^k) \cdot \lambda_i^j$ where λ_i^j is the measured arrival rate of flux F_i^j from neighbor i. Let B_u stand for the capacity of link u. The goal of the fairness enforcement scheme is to prescribe a method for dropping packets belonging to different fluxes under contention, i.e., when $\sum_{k=1}^{n_u} x_u^k > B_u$. The router executes the following algorithm to isolate the unfair fluxes from the ones that use their fair share:

```
$C = \{1, ..., n_u\};\ s = n_u;\ B_u^{\text{cont}} = B_u;$
while $(\exists k \in C | x_u^k < B_u \cdot \rho(G_u^k))$ {
    $C = C - \{k\};\ B_u^{\text{cont}} = B_u^{\text{cont}} - x_u^k;\ s = s - 1;$
}
```

The right hand side of the *while* condition, $b_u^k = B_u \cdot \rho(G_u^k)$, represents the fair share of the link's bandwidth that flux k should receive under contention. Note that from Eq. (2) we have that $\sum_{k=1}^{n_u} \rho(G_u^k) = 1$. The usage of $\rho(G_u^k)$ in the algorithm above explains the interpretation of the rank of a flux: it directly determines the proportion of the link's bandwidth that the flux should receive in a fair allocation under contention. The algorithm calculates two values: B_u^{cont}, which is the bandwidth subject to contention after all fluxes that remain within their fair shares have been allocated their requested bandwidth, and C, which is the set of indexes identifying the unfair fluxes.

C. Fair Allocation of Bandwidth

We shall say that a function $m(x_i, b_i)$ of incoming rate x_i and fair share b_i (as defined above) is a fair maximum bandwidth function if 1) for any two fluxes i and j such that $x_i/b_i = x_j/b_j$ we have $m(x_i, b_i)/b_i = m(x_j, b_j)/b_j$, and 2) for any two fluxes i and j such that $x_i/b_i < x_j/b_j$ we have $b_i/m(x_i, b_i) \geq b_j/m(x_j, b_j)$. The first condition ensures that if two fluxes have the same ratio of offered load to fair share, then they will be given the same ratio of maximum bandwidth to fair share. The second condition ensures that a higher ratio of offered load to fair share does not increase the ratio of maximum bandwidth to fair share, i.e., the function gives an upper bound on the bandwidth that a router will allocate to a flux.

By our definition, a bandwidth allocation mechanism which allocates bandwidth $\hat{x_I}$ to a flux F_i is a transport-independent fair allocation mechanism with respect to a fair maximum bandwidth function $m(x_i, b_i)$ and ranks r_i, if it satisfies the following conditions for all links u:

1) $\sum_{k=1}^{n_u} x_u^k \leq B_u$
2) $x_u^k \leq b_u^k \Rightarrow \hat{x}_u^k = x_u^k$
3) $x_u^k > b_u^k \Rightarrow \hat{x}_u^k \leq m(x_u^k, b_u^k)$
4) If an unfair flux changes its offered load from $x_u^k(t_1) > m(x_u^k, b_u^k)$ to $x_u^k(t_2) > x_u^k(t_1)$, and there are no changes in the offered loads of any other fluxes, its allocated rate changes from $\hat{x}_u^k(t_1)$ to $\hat{x}_u^k(t_2) \leq \hat{x}_u^k(t_1)$.

The first condition constrains the allocated rates to the capacity of the link. The second condition says that a flux requesting its fair share will be allocated what it requests. The third condition says that a greedy flux (i.e., one exceeding its bandwidth limit) will be allocated no more than its maximum fair bandwidth. The fourth condition ensures that a flux cannot obtain more bandwidth (in excess of its fair share) by strategically increasing its offered load—that is, by being greedy. Informally, the condition says this to the flux: “the more traffic you submit in excess of your fair share, the *less* of the actual bandwidth you are going to get.” Consequently, it becomes the flux’s objective to probe bandwidth for the current fair share and navigate towards it, as this is the only way the flux can *maximize* its performance. This is what we mean by the transport-independence of our bandwidth allocation scheme.

III. Illustration

There are many interesting fair maximum bandwidth functions that satisfy the conditions in section II-C. One example of such a function is:

$$m(x_u^k) = x_u^k (b_u^k / x_u^k)^\alpha \tag{3}$$

where $\alpha \geq 1$ is a constant. Note that it ensures that fluxes cannot exceed their fair share. To illustrate its behavior, we apply it to the single-router setup of Figure 1 catering to four types of fluxes listed in Table I. These flux types are not meant to be accurate representations of specific protocols, although their general characterizations as “rational”, “greedy”, etc. was chosen to convey their intent. In Table I, x is the offered load, b is the bandwidth allocated, d is the actual demand the flux seeks to reach, and V_{RA} and V_{NG} are threshold values. For the following experiments we set $\alpha = 1.5$, $V_{RA} = 1.5$ and $V_{NG} = 3.0$.

TABLE I
Generic flux types

Type	Description
Rational (RA)	$b \geq x \Rightarrow x = x + 1$ $(b < x) \wedge (x/b \leq V_{RA}) \Rightarrow x = x - 1$ $(b < x) \wedge (x/b > V_{RA}) \Rightarrow x = x/2$
Near-rational (NR)	$b \geq x \Rightarrow x = x + 1$ $b < x \Rightarrow x = x/2$
Greedy (GR)	$b \geq x \Rightarrow x = d$ $b < x \Rightarrow x = x + d - b$
Near-greedy (NG)	$b \geq x \Rightarrow x = d$ $(b < x) \wedge (x/b \leq V_{NG}) \Rightarrow x = x + d - b$ $(b < x) \wedge (x/b > V_{NG}) \Rightarrow x = x/2$

First we consider the case where the capacity of all links is so low that all fluxes are allocated bandwidth less than their demand. We set $B_1 = B_2 = B_3 = 1000$ with the local policy described as $r_1 = 0.3$, $r_2 = 0.4$ and $r_3 = 0.3$. The population of fluxes is defined in Table II.

TABLE II
Fluxes used in first scenario

Flux	Src	Type	Dmnd	Inc Rank	Lnk1 Fract	Lnk2 Fract	Lnk3 Fract
1	1	RA	3000	0.3	-	1.0	0.0
2	1	GR	4000	0.5	-	0.0	1.0
3	1	NG	3000	0.2	-	1.0	0.0
4	2	NR	5000	0.5	1.0	-	0.0
5	2	NG	5000	0.5	0.0	-	1.0
6	3	RA	2500	0.4	0.0	1.0	-
7	3	NR	2500	0.3	1.0	0.0	-
8	3	GR	2500	0.1	1.0	0.0	-
9	3	NG	2500	0.2	0.0	1.0	-

The loads offered by a rational flux and a near-greedy flux are shown in Figure 2. These are fluxes 1 and 3 on link 2. The rational flux has a fair share of 273 and converges relatively quickly to an offered load of this value by t=100. The near-greedy flux has a fair share of 182 but persists in trying to achieve a much higher rate and displays an offered load which oscillates between about 190 and 3250. Figure 3 shows the results of applying the fair maximum bandwidth function to these two fluxes along with flux 6 (rational, fair share = 364). The rational fluxes both achieve their fair share and converge to offered load at that value resulting in zero drop rate. The near-greedy flux suffers an increasing penalty in the phase where it increases its offered load. It only achieves its fair share after reducing the load it offers. The other near-

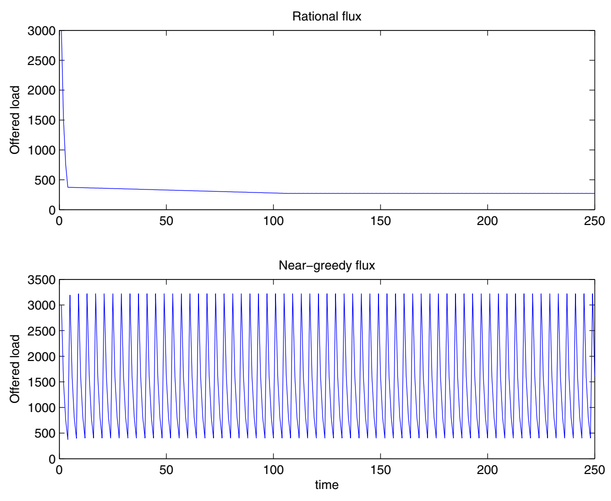

Fig. 2. Example offered loads

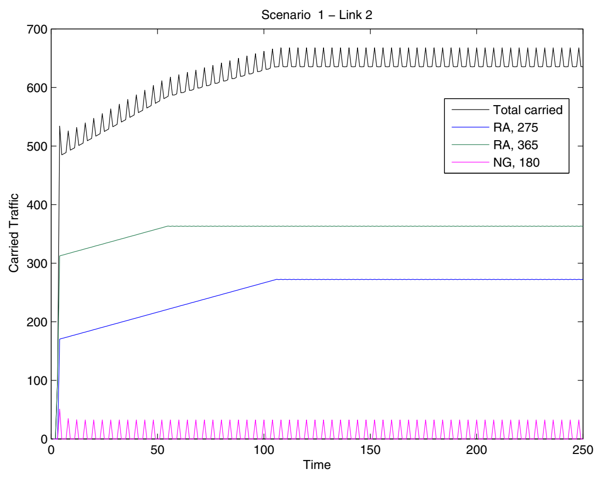

Fig. 3. Link 2 in the first scenario

greedy flux traversing this link, flux 9, is treated similarly: its carried load is bounded from above by its fair share. Clearly, despite the relatively simple form of the maximum fair bandwidth function, it is effective in enabling co-operative fluxes to achieve their goals while preventing greedy fluxes from dominating the network.

A second set of results was obtained by changing the form of the fair maximum bandwidth function to:

$$m(x_u^k) = x_u^k - \beta \cdot (x_u^k - b_u^k) \; \forall x_u^k > b_u^k \tag{4}$$

For the following results we chose $\beta = 1.05$. We used the same set of fluxes as in the first scenario but different fractions of the incoming fluxes were distributed to the outgoing links (see Table III). The capacity of link 3 was increased to $B_3 = 10000$ so that we could examine the behavior of a mix of fluxes on a non-bottleneck link. In Figure 4 we present the carried traffic for link 2 (which is still a bottleneck link). On link 2 the

TABLE III
REVISED FLUXES FOR SECOND SCENARIO

Flux	Src	Type	Dmnd	Inc Rank	Lnk1 Fract	Lnk2 Fract	Lnk3 Fract
1	1	RA	3000	0.3	-	0.5	0.5
2	1	GR	4000	0.5	-	0.0	1.0
3	1	NG	3000	0.2	-	0.5	0.5
4	2	NR	5000	0.5	0.7	-	0.3
5	2	NG	5000	0.5	0.3	-	0.7
6	3	RA	2500	0.4	1.0	0.0	-
7	3	NR	2500	0.3	1.0	0.0	-
8	3	GR	2500	0.1	1.0	0.0	-
9	3	NG	2500	0.2	1.0	0.0	-

rational flux converges fairly quickly to its fair share of 600 and also reduces its offered load to that value, thus achieving a zero drop rate. In contrast, the near-greedy flux shows an offered load which oscillates between about 1100 and 2200. It achieves a maximum carried load of just 365 while its fair share is actually somewhat higher, at 400. Link 2 is operating at over 90% utilization with no impact from the near-greedy flux on the rational flux.

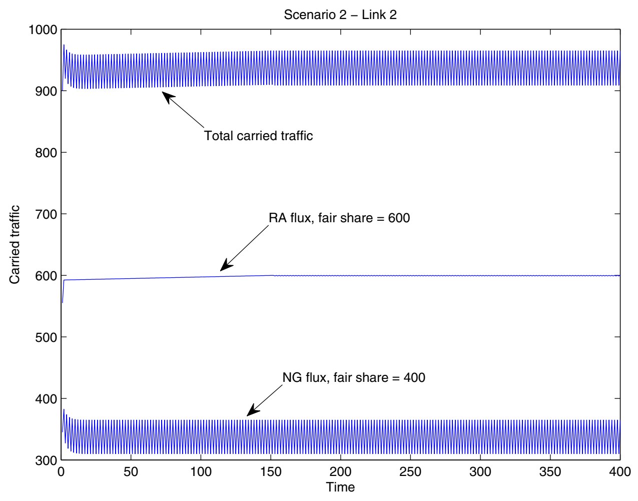

Fig. 4. Link 2 in the second scenario

The activities seen on the non-bottleneck link (Figure 5) are quite interesting. Again, the rational flux converges fairly quickly to its fair share. The near-rational flux takes longer to converge as a result of starting out at an offered load of 1500 which is slightly higher than its fair share of 1410. After cutting its offered load in half, the flux gradually advances towards the fair share. The two near-greedy fluxes almost achieve their fair shares, oscillating at carried loads just slightly below them. However, they suffer significant and oscillating drop rates. The greedy flux behaves in a completely unreasonable fashion despite the fact that its fair share of 3530 is a large proportion of its total demand of 4000. As the fair maximum bandwidth function tries to persuade this flux to reduce its offered load, the flux consistently increases what it

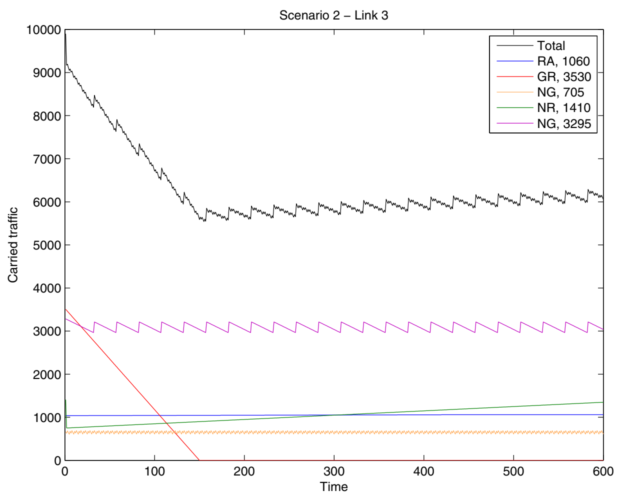

Fig. 5. A non-bottleneck link

offers in an attempt to exploit the network. The net result is that by time 150 the greedy flux is achieving a carried load of zero because of the extreme level to which it has increased its offered load. Link 3 is quite successfully carrying a mix of fluxes with widely different patterns. The rational and near-rational fluxes achieve their fair shares at zero or near zero drop rates. The near-greedy fluxes approach their fair shares, but suffer significant drop rates. Notably, the greedy flux is completely thwarted in its efforts to escape control.

IV. Conclusions

Our scheme makes it possible to implement a fair operation of a large network without interfering with its connectionless paradigm. The scope of a flux and the identity of its *user* depend on the place in the network. For example the different users of a single host can be viewed by the host administrator as contributors of different fluxes, while all the packets contributed by that host can be treated as a single flux by its edge router. Notably this hierarchy need not be strictly enforced at absolutely all levels, which makes it possible to deploy the scheme gradually and meaningfully at the same time.

Our solution can be viewed as a way to equip the connectionless network with the minimum characteristics of a connection-oriented system to provide for (fair) *bandwidth guarantees*. Note that, if required, our framework makes it possible to provide *hard* bandwidth guarantees without imposing undue restrictions on the transport-layer mechanisms involved in connection setup. In the simplest case, one can declare a highly ranked flux encompassing a single transport-layer session. For as long as there is no contention along this flux's path, it will share bandwidth with other fluxes in a flexible manner. As soon as the preferred flux finds itself competing for bandwidth at a congested link, the router will appropriately trim the other fluxes, as to make sure that the preferred flux receives the kind of treatment it needs. With the proper interpretation of ranks, this mechanism can be applied in a hierarchical fashion. This way, the rigid connection-oriented approach to quality of service becomes a trivial subset of our scheme, which, at its full power, is incomparably more flexible.

While the present results are preliminary, the simple illustration in section III carries enough convincing power to warrant a further study of our proposed scheme. Note that, as seen by the network operator, the ranks assigned to fluxes should reflect the operator's policy. The problem of assigning ranks to fluxes can be defined as an optimization problem aimed at maximizing the network's *utility* function. We are currently investigating several designs of *controllers* that take offered loads of the fluxes as input and dynamically optimize link ranks and carried loads in the spirit of our scheme described in section II.

References

[1] S. Jin, L. Guo, I. Matta, and A. Bestavros. A spectrum of TCP-friendly window-based congestion control algorithms. *IEEE/ACM Trans. Netw.*, 11(3):341–355, 2003.

[2] S. Low, L. Peterson, and L. Wang. Understanding TCP Vegas: a duality model. In *SIGMETRICS '01: Proceedings of the 2001 ACM SIGMETRICS Intl. Conf. on Measurement and Modeling of Computer Systems*, pages 226–235, New York, NY, USA, 2001. ACM.

[3] Z. Wang and J. Crowcroft. Eliminating periodic packet losses in the 4.3-Tahoe BSD TCP congestion control algorithm. *SIGCOMM Comput. Commun. Rev.*, 22(2):9–16, 1992.

[4] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker. Controlling high bandwidth aggregates in the network. *SIGCOMM Comput. Commun. Rev.*, 32(3):62–73, 2002.

[5] R. Mahajan, S. Floyd, and D. Wetherall. Controlling high-bandwidth flows at the congested router. *ICNP*, 00:0192, 2001.

[6] V. Jacobson. Congestion avoidance and control. In *SIGCOMM '88: Symposium proceedings on Communications architectures and protocols*, pages 314–329, New York, NY, USA, 1988. ACM.

[7] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley. Inferring TCP connection characteristics through passive measurements. In *INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 3, pages 1582–1592 vol.3, 2004.

[8] Y. Gu and R. L. Grossman. UDT: UDP-based data transfer for high-speed wide area networks. *Computer Networks*, 51(7):1777–1799, 2007.

[9] E. He, J. Leigh, O. Yu, and T. A. DeFanti. Reliable blast UDP: Predictable high performance bulk data transfer. In *CLUSTER '02: Proceedings of the IEEE International Conference on Cluster Computing*, page 317, Washington, DC, USA, 2002. IEEE Computer Society.

[10] K. chan Lan and J. Heidemann. A measurement study of correlations of Internet flow characteristics. *Comput. Netw.*, 50(1):46–62, 2006.

[11] W. Shi, M. MacGregor, and P. Gburzynski. Synthetic trace generation for the Internet: An integrated model. In *SPECTS 2004: Symposium on Performance Evaluation of Computer and Telecommunication Systems*, pages 471–477. Society for Computer Simulation, 2004.

[12] S. Chen, Y. Tang, and W. Du. Stateful DDoS attacks and targeted filtering. *J. Netw. Comput. Appl.*, 30(3):823–840, 2007.

[13] S. S. Kim and A. L. N. Reddy. Statistical techniques for detecting traffic anomalies through packet header data. *IEEE/ACM Trans. Netw.*, 16(3):562–575, 2008.

[14] W.-C. Liao, F. Papadopoulos, and K. Psounis. Performance analysis of BitTorrent-like systems with heterogeneous users. *Perform. Eval.*, 64(9-12):876–891, 2007.

[15] D. Qiu and R. Srikant. Modeling and performance analysis of BitTorrent-like peer-to-peer networks. *SIGCOMM Comput. Commun. Rev.*, 34(4):367–378, 2004.

[16] J. Moy. RFC 2328: OSPF version 2, Apr. 1998. See also STD0054. Obsoletes RFC2178. Status: STANDARD.

[17] Y. Rekhter and T. Li. RFC 1771: A Border Gateway Protocol 4 (BGP-4), Mar. 1995. Obsoletes RFC1654. Status: DRAFT STANDARD.

[18] P. White. RSVP and integrated services in the Internet: a tutorial. *IEEE Communications Magazine*, 35(5):100–106, 1997.