Specifications for End to End IP Rate Control
(Version 1.0)

Abdul Aziz Mustafa, Mahbub Hassan and Sanjay Jha
Network Research Laboratory
School of Computer Science and Engineering
The University of New South Wales
Sydney 2052, NSW
Australia
Email : {amustafa, mahbub, sjha}@cse.unsw.edu.au

UNSW-CSE-TR-0005
October 2000

THE UNIVERSITY OF
NEW SOUTH WALES

School of Computer Science and Engineering
The University of New South Wales
Sydney 2052, NSW
Australia

Abstract

Currently no network-level flow control exists in the IP-based networks. In a recent paper [1], we proposed a network-level flow control architecture, called End-to-End IP Rate Control. The motivation behind IP Rate Control is to provide a new network service which will provide users fast access to any unused network resources. This report details the specifications of the IP Rate Control architecture which can be used to implement the service in a given networking platform.

1 Introduction

Figure 1 illustrates the concept of End to End (E2E) IP rate control which is similar to the rate control used in Asynchronous Transfer Mode (ATM) networks [2]. Using Internet Control Message Protocol (ICMP), the source periodically sends probe packets to the destination (Forward Probe Message or FPM). The destination returns these probe packets to the source (Backward Probe Message or BPM). Routers along the route compute the fair share for the flow to update the probe messages. When a probe packet returns to the source, it carries the bottleneck fair share in the end-to-end path. All traffic sources are required to shape their traffic according to the feedback returned in the probe packets. Since it is difficult to trust all users in the commercial environment, *edge routers* (ERs) police every incoming flow into the network to ensure that users do not violate their feedback rate.

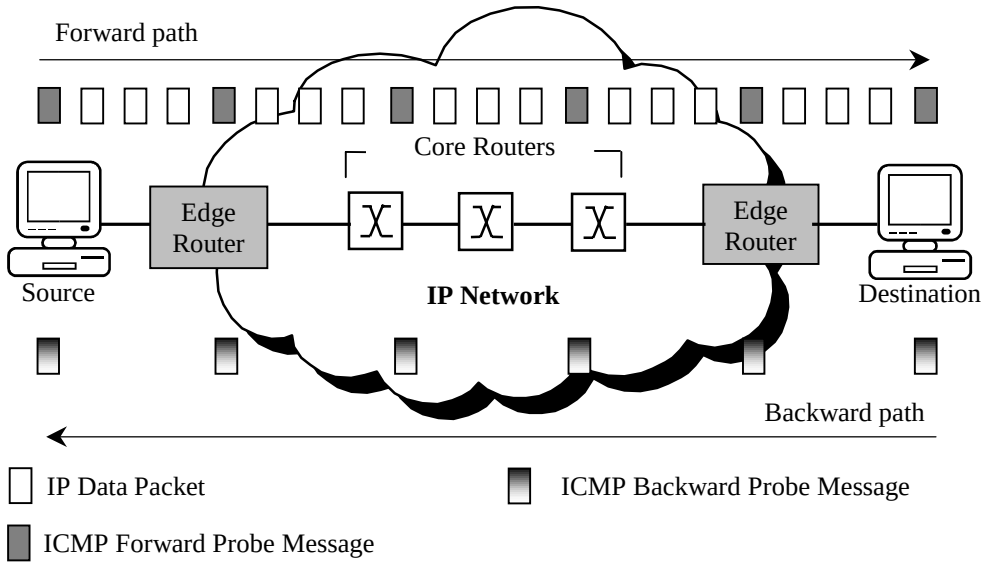

Figure 1 - E2E IP rate control using ICMP probe messages

Although IP rate control sounds very similar to ATM ABR flow control, there are several new challenges in designing such end-to-end rate control for IP networks. IP networks are connectionless without having virtual circuits set up for each flow. To implement E2E IP Rate Control, the three entities (i.e. IP hosts, edge routers and core routers) must play different roles. The design specifications of these three entities are presented in the following sections.

2 IP host

The primary role of IP host is to regulate its traffic based on the *explicit feedback rate* (EFR) provided by the network. The IP host contains four functional components as shown in Figure 2. They are flow classifier, traffic shaper, ICMP FPM generator (not shown) and feedback handler. Traffic generated by applications is classified into different flows by the classifier. Each flow is identified based on contents of some fields in packet's header. For IPv4 [3], classification is carried out by examining packet's source and destination addresses, source and destination port numbers and protocol identifier. For IPv6 [4], packets are classified into a flow based on their source and destination addresses and flow label fields in the header.

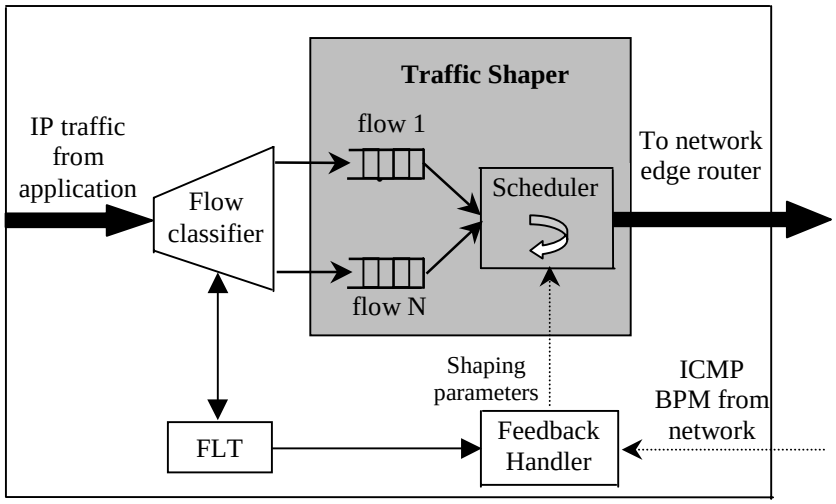

Figure 2 - Functional design of IP hosts

Each outgoing flow from the classifier is channelled to a traffic shaper element. The traffic shaper is employed to regulate the rate of a flow based on the explicit feedback rate (EFR) provided by the network. The EFR rate of a flow is computed based on the bottleneck fairshare in routers along the end-to-end path. The EFR is communicated to the source via a feedback handler residing in the host using Internet Control Message Protocol (ICMP). Each time the handler receives an ICMP Backward Probe Message (BPM), it identifies the flow relevant to the ICMP BPM message and reads the content in the EFR field. The EFR is sent as shaping parameters to the traffic shaper.

2.1 Flow classifier

In providing end-to-end IP rate control, traffic from each source must be classified into a flow to allow traffic to be shaped according to the EFR provided by the network. The classifier maintains a flow look-up table (FLT), which contains entries of current flows. Figure 3 shows the interaction between a classifier and the FLT. Figure 4 illustrates the format of the FLT.

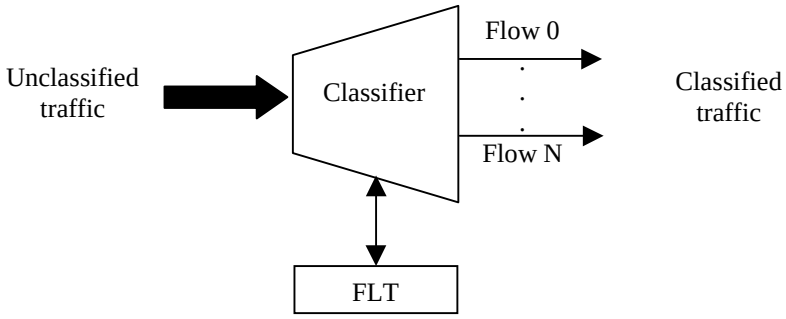

Figure 3 - Flow classification at IP source

Flow ID	*IP Version*	*Source Addr.*	*Destn. Addr.*	*Source Port No.*	*Destn. Port No.*	*Protocol (TCP=6, UDP=17)*	*Flow Label*
0	4	x	x	x	x	6	NA
1	4	x	x	x	x	17	NA
2	6	x	x	NA	NA	NA	x
3	4	x	x	x	x	17	NA
4	6	x	x	NA	NA	NA	x
5	4	x	x	x	x	6	NA
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
N	4	x	x	x	x	17	NA

Figure 4 - Format of a Flow Look-up Table (FLT)

Flow classifier is used to differentiate packets (from applications) into different flows by examining the contents of some fields in packet's header. Figure 5 illustrates the classification process. Whenever the classifier receives a packet from higher layer of an IP source, it examines the contents of the packet's header fields. If this is a new

packet (i.e. flow id does not exist in flow look-up table (FLT)), a new entry is created for the flow. Subsequent packets originating from the same source of flow will be mapped to the same flow id accordingly. Packets that have been classified are sent to their corresponding traffic shaper for the flow.

A time-out timer (T_{out}) is associated for each flow entry in the table. The timer is used to track a flow's activity. This is to allow idle flow to be deleted and reused for new flow to minimise the FLT entry size. Every time a packet is received, the timer resets to zero and starts timing until the next packet arrives. If no packet is received after the time-out period, the flow is assumed to be no longer active and its entry will be deleted from the FLT. A delay between 1 to 2 minutes is recommended for the time-out timer. This is because for TCP applications, the RTT on a LAN can be milliseconds while across a WAN can be seconds [5]. For UDP applications, similar assumption to TCP is made because UDP does not employ mechanism that depends on RTT.

An alternative approach is the use of Least Recently Used (LRU) method which does not involve the use of any timer. Each time a new flow arrives, the flow's entry is placed on top of a stack. For instance, given a FLT entry size of 256, in the event of a new flow arrives and no more entry is available in the FLT (all entries are fully occupied), the entry at the bottom of the stack is purged from the FLT to accommodate for the new flow.

The flow ids for all active connections are maintained in the FLT locally and accessible (read/write) by the classifier. Information in the FLT will also be required by feedback handler when processing the explicit feedback rate (EFR) of a flow.

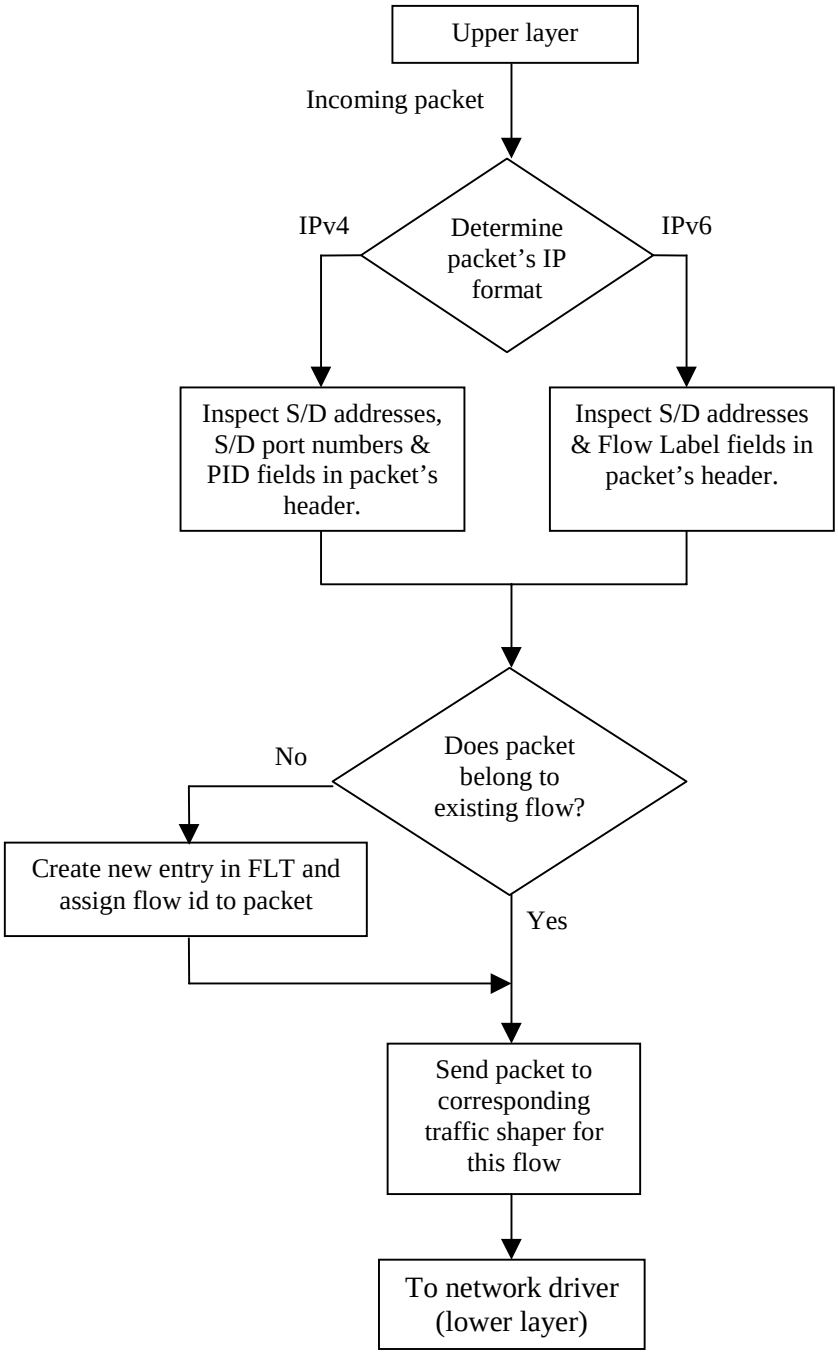

Figure 5 - Classification of transmitted packet into a flow at IP host

2.2 Traffic shaper

The purpose of employing a traffic shaper in an IP host is to regulate a flow based on the explicit feedback rate (EFR) provided by the network. A shaper is used to modify the rate of a flow to bring traffic into compliance with its EFR. The EFR is communicated in the EFR field of ICMP BPM by the network via a feedback handler in the IP host.

Packets that have been classified into a flow are sent to the corresponding buffer in the traffic shaper for that flow as shown in Figure 6. Packets are drained out from the buffer at a rate set to the EFR, by the scheduler and dispatched to the output queue in the network interface card (NIC). The maximum bit rate, MBR_{ts}, specifies the shaping rate set to EFR for a given flow.

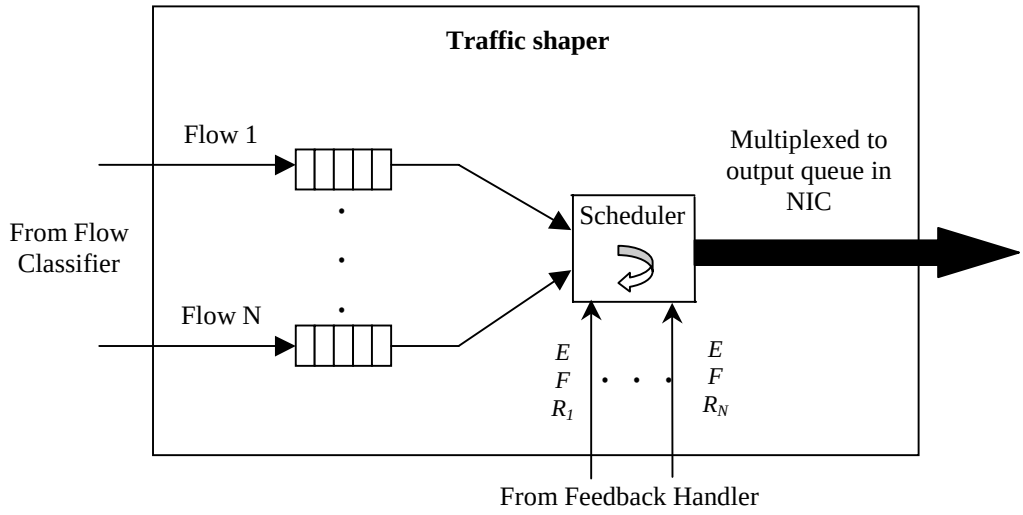

Figure 6 – Shaping process at IP host

2.3 ICMP Forward Probe Message (FPM) generator

For every N data packets sent in a flow, the FPM generator creates an ICMP FPM packet to be sent among the data packets to the destination. There is one FPM generator for each flow.

2.4 Feedback handler

The feedback handler is used to process ICMP Backward Probe Message (BPM). Every time it receives an ICMP BPM from the edge router, the feedback handler consults the FLT to determine which flow it belongs to. The feedback handler then extracts the EFR value from the EFR field in the ICMP BPM and sends it to the scheduler (see Figure 6).

3 Network edge router

Network edge router can be defined as the ingress or egress router (depending on the direction of traffic) and situated at the boundary of a network. An edge router (ER) connects end systems to the network. An edge router (ER) consists of a flow classifier, traffic policer and feedback engine as shown in Figure 7.

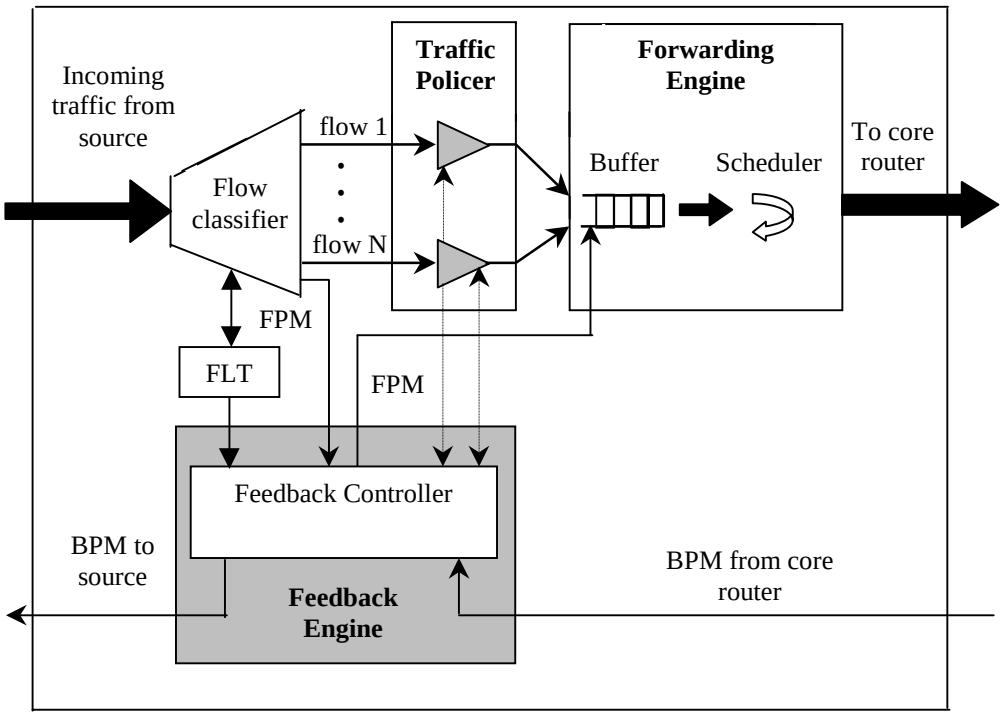

Figure 7 - Functional design of network edge routers

3.1 Flow classifier

At the network edge router, incoming traffic from one or more IP hosts are classified into different flows. The flow classification process is similar to that in the IP host except that now, each flow is output to a traffic policer. The classifier also maintains a flow look-up table which is used to store all current flows information. This information will later be required by the feedback engine to determine which flow a feedback signal (EFR rate) belongs to whenever the feedback engine receives an ICMP

BPM from the downstream core router. The EFR rate is used as policing parameters for the traffic policer of a flow.

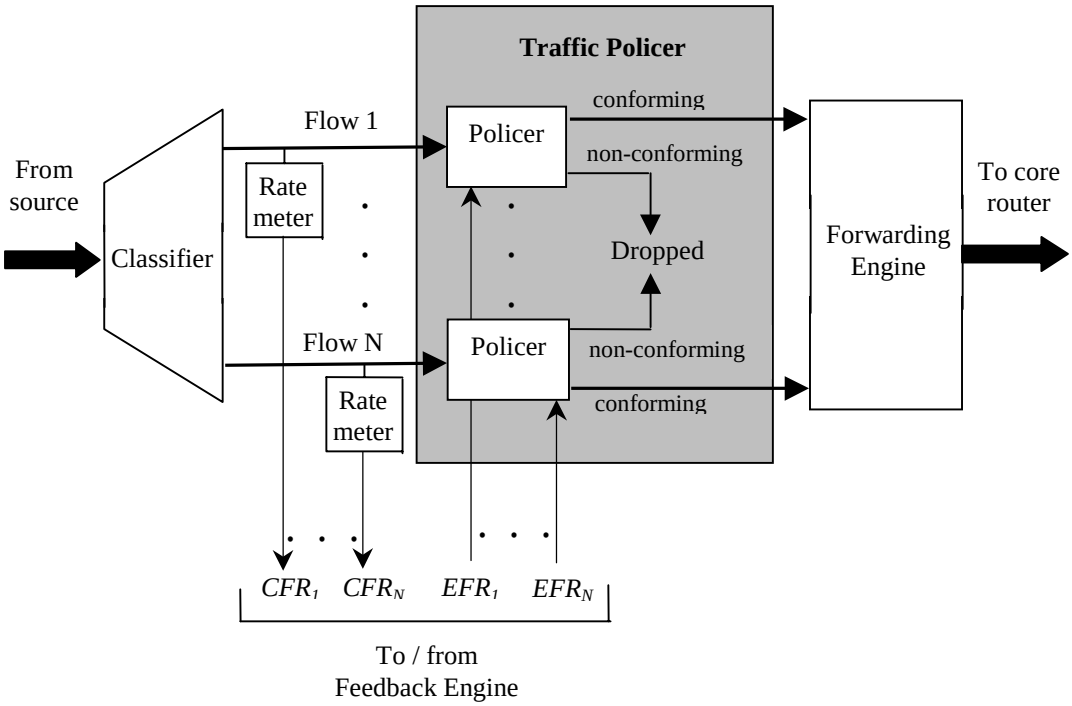

Figure 8 – Flow classification and policing at edge router

3.2 Traffic policer

A traffic policer is employed at the ingress network edge router to protect the network from unfriendly traffic sources by ensuring that a source does not send its traffic exceeding the EFR rate as prescribed by the network. Packets sent from a source exceeding its EFR rate (non conforming packets), are strictly dropped to prevent the source from seizing more than its allocated fair share in the network.

A traffic policer is implemented using a leaky bucket algorithm [6]. The leaky bucket behaves like a bucket with a hole in its bottom. If data flows into the bucket is faster than it flows out of the bucket, then the bucket eventually overflows, causing data to be discarded. The basic principle of the leaky bucket is depicted in Figure 9.

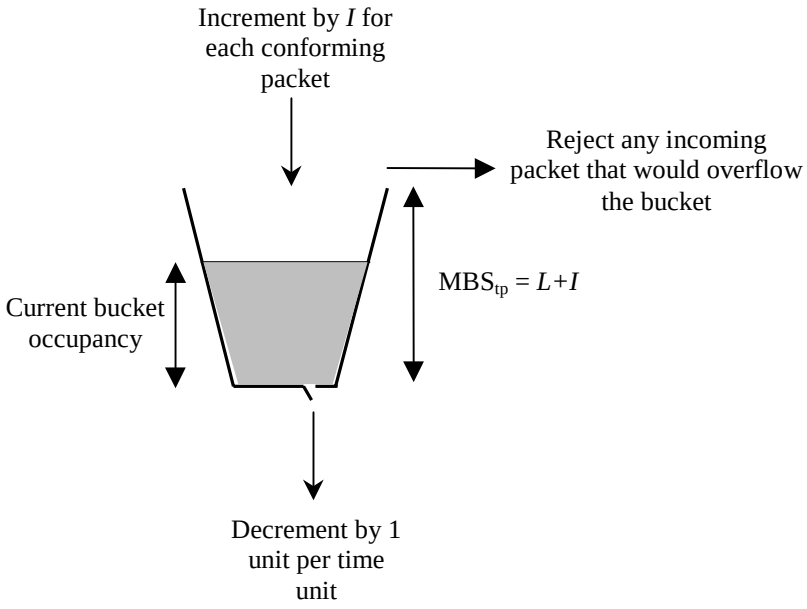

Figure 9 – Leaky bucket algorithm

The algorithm maintains a running count of the cumulative amount of data sent in a counter *X*. The counter is decremented at a constant rate of one unit per time unit (i.e. set to EFR of flow) to a minimum value of zero, which is equivalent to a bucket that leaks at a rate of 1. The counter is incremented by *I* for each arriving packet and subject to restriction that the maximum counter value is $I + L$. Any arriving packet that would cause the counter to exceed its maximum is defined as nonconforming.

The leaky bucket consists of two control parameters:-

- MBR_{tp} **–** the rate at which packets are allowed to enter the network (equal to EFR of a flow)

- MBS_{tp} - number of packets that are allowed to accumulate in the bucket at per unit time interval ($L + I$). In general, this is usually set to few packets (one or two) to accommodate some delay tolerance due to hardware and software elements.

Packets that arrive within EFR are referred as conforming packets, and are sent directly to forwarding engine. On the other hand, packets that arrive at a rate higher than EFR are referred as non-conforming and are dropped. This is shown in Figure 8.

3.2.1 Rate meter

A rate meter is used to estimate the current flow rate of a flow at the edge router. The current flow rate, CFR of a flow *i* is computed every time a new packet arrives by using exponential averaging formula [7] :-

$$CFR_i^{new} = (1 - e^{-T/K}) \frac{l}{T} + e^{-T/K} CFR_i^{old} \quad (1)$$

where l is the size of packet, T is packet inter-arrival time between the current and previous packets, CFR_i^{old} is the value of CFR_i^{new} before the updating, and K is a constant. The choice of K in the above expressions $e^{-T/K}$ presents several tradeoffs. Details of these tradeoffs can be found in [7]. An appropriate value for K would be between 0.1 to 0.5 seconds.

It is worth mentioning here that the output from each rate meter is also required by max-min fair share algorithm employed at the edge router to compute the bottleneck fair share for each flow. This is discussed in Section 3.4.

3.3 Feedback engine

The main function of the feedback controller is to compute the bottleneck fair share (EFR) for every active flow using max-min fair share algorithm. The EFR for a given flow is computed every time the feedback controller receives an ICMP BPM for that flow.

3.4 Max-min fair share algorithm

The concept of max-min fairshare was first proposed in [8] to address the issue of allocating fair amount of network resources among active connections. If there are N connections sharing a link, each source is assigned one Nth fairshare of the link bandwidth. If a connection cannot use its fairshare bandwidth because it has a lower source rate or it had been assigned a lower bandwidth on an upstream link, the excess bandwidth is split fairly among all other connections on the link.

The max-min fair share algorithm computes the bottleneck fair share for each flow at the edge router. The max-min fair share for a flow is computed based on the current flow rate (CFR) of all active flows and the capacity of the output link.

As mentioned earlier, a rate meter is used to monitor the incoming CFR of each flow. Each flow is associated with a constrained flag (CF) to determine whether or not a flow is constrained at the edge router. Table 1 shows the information maintained by the max-min fair share scheme at the edge router. The constrained flag is used to allocate bandwidth according to the max-min fairness.

Flow id	***Explicit Feedback Rate (EFR)***	***Current Flow Rate (CFR)***	***Constrained Flag (CF)***
Integer	Float	Float	Boolean

Table 1 - Flow information at edge router

The max-min fair share algorithm is described as follows. Let *FS* be the fair share of the bandwidth for unconstrained flows.

$$FS = \frac{C - \sum_{i \in F_C} CFR_i}{\left| F_{UC} \right|} \qquad (2)$$

where C is the capacity of output link, and CFR_i is CFR of ith flow. F_C and F_{UC} are sets of constrained and unconstrained flows, respectively. $|F_{UC}|$ represents the number of unconstrained flows. For *FS*, the constrained flag for each flow is updated by taking the minimum between the computed *FS* and the unconstrained flows in the set, F_{UC}. The new constrained flow is moved from F_{UC} to F_C. The iteration is repeated until there is no change in constrained flags. When the iteration process has completed, each flow is assigned to its max-min fair share at the edge router. Figure 10 shows the flow chart for computing the max-min fair share for each flow.

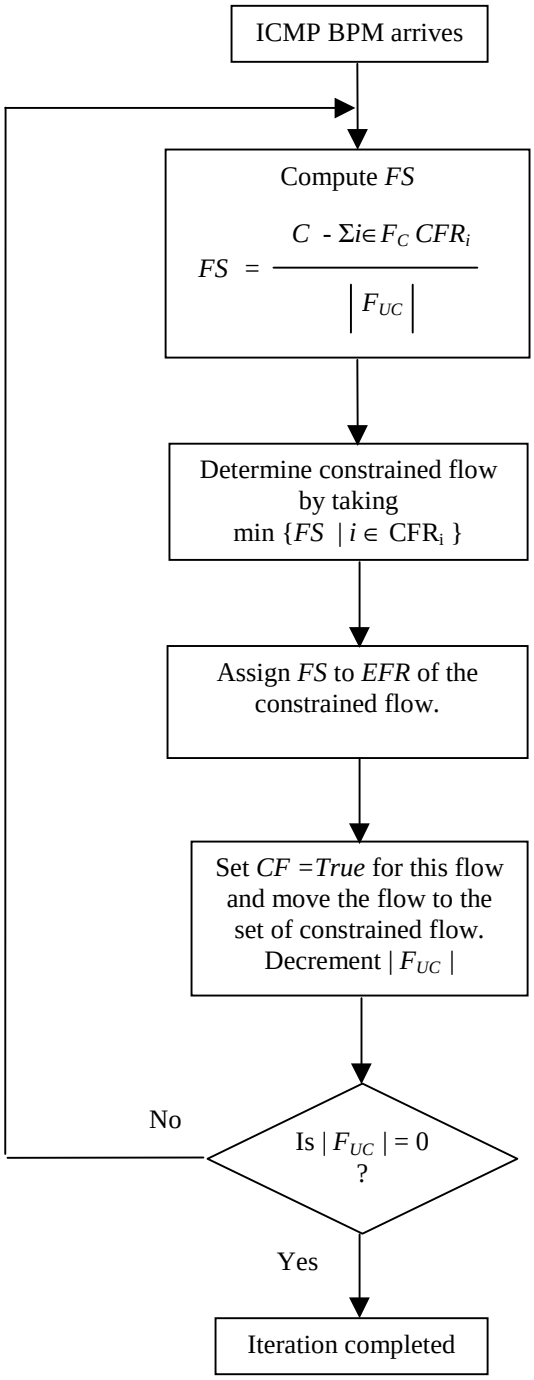

Figure 10 - Flow chart to compute max-min fair share.

Following, we provide an example to illustrate how the max-min fair share algorithm computes fair share for each flow. Assume there are four active flows traversing the edge router. These flows are 2 kbps, 2.6 kbps, 4 kbps and 5 kbps with output link capacity of 10 kbps. The edge router first determines the fair share to all flows. This is achieved by applying Equation (2) above, such that,

$$FS = (10 - 0) / 4$$
$$= 2.5 \text{ kbps}$$

Note that the sum of set of constrained flows ($\sum_{i \in F_C} CFR_i$) is initially set to zero because all flows are assumed to be unconstrained during the first iteration. After the first iteration, the computed *FS* is compared to the set of unconstrained flows to determine whether or not a flow is constrained at the edge router. For this example, the flow with rate 2 kbps is constrained because it requires lower rate than the *FS*. The constrained flag for the flow is set and the flow is moved to the set of constrained flow. The iteration is repeated.

$$FS = (10 - 2) / 3$$
$$= 2.67 \text{ kbps}$$

Similarly, the *FS* is compared with the set of unconstrained flows. The next constrained flow is the flow with rate of 2.6 kbps. The constrained flag is set and this flow is moved to the set of constrained flows. The process is repeated for the remaining of unconstrained flows until there is no change in the constrained flag for all flows. Table 2 shows result of max-min fair share assigned to each flow in this example.

Flow$_i$	***Current Flow Rate (CFR)***	***Max-min fair share***
1	2 kbps	2 kbps
2	2.6 kbps	2.6 kbps
3	4 kbps	2.7 kbps
4	5 kbps	2.7 kbps

Table 2 - Example of max-min fair share allocation

4 Core router

The design of the core router is similar to the edge router except that now, incoming traffic from upstream edge router is sent directly to the forwarding engine as shown in Figure 11. A feedback engine in a core router is used to process ICMP BPM. Every time an ICMP BPM for a flow arrives, the feedback engine computes the bottleneck fairshare of the flow. The bottleneck fair share for a flow at the core router is computed based on max-min fair share approximation to avoid the need to maintain per flow state and to prevent scalability problem where the number of active flows can be possibly large in contrast to the number of incoming flows at the edge router. Thus, it is recommended that E2E IP rate control uses an approximation technique as presented in [6].

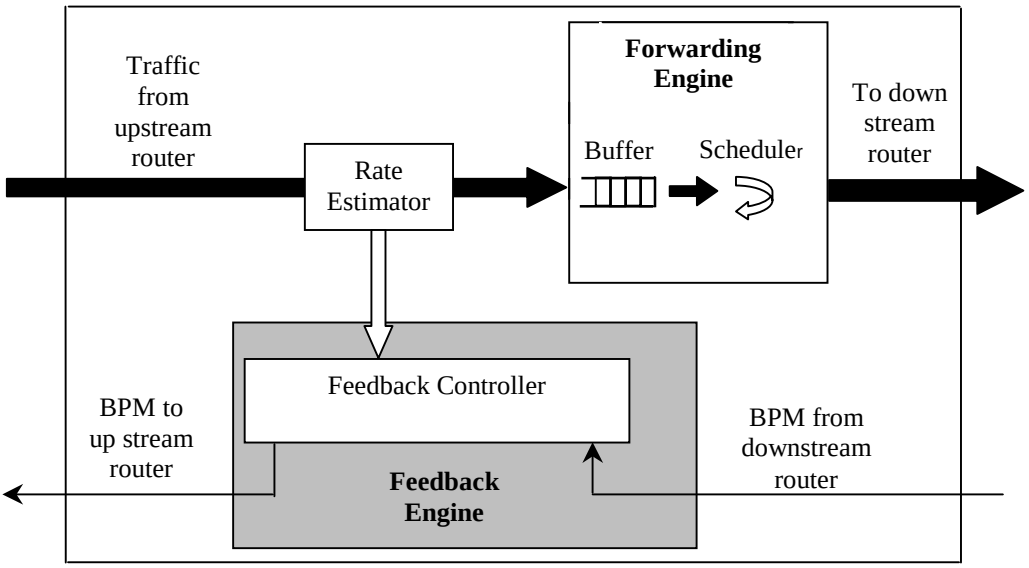

Figure 11 - Functional design of core routers

The max-min fair share for a given flow is computed based on the current flow rate (CFR) of the source, aggregated rate of all active flows currently traversing the core router and output link bandwidth. The CFR for a flow *i* is computed by a rate meter at the edge router and conveyed to the core router using ICMP FPM.

The aggregated rate of all active flows is computed by a rate estimator in the core router. As shown in Figure 11, packets from all flows are aggregated into the buffer in

the forwarding engine. We use analogous formula in Equation 1 to compute the aggregated rate at every packet arrival :-

$$A^{new} = (1 - e^{-T/K}) \frac{l}{T} + e^{-T/K} A^{old} \quad (3)$$

Finally, the approximate max-min fair share for a given flow F_i, can be computed as follows [6]:-

$$F_i = CFR_i^{new} * C / A^{new} \quad (4)$$

where C is the output link bandwidth at a router. The max-min fairshare is computed every time a core router receives an ICMP BPM.

5 ICMP Probe Message

ICMP PM is a special control packet used to carry information related to a traffic flow. The ICMP PM consists of several fields and its format is shown in Figure 12. ICMP message Type 40 is used to differentiate ICMP Probe Message (PM) from other ICMP messages that already exist in the current Internet. Other than the standard ICMP fields such as the Type, Code and Checksum, five important fields are defined to convey information related to a flow in the end-to-end path. These are the D (Direction) bit, CFR (Current Flow Rate), the EFR (explicit feedback rate), the source port and destination port addresses. Table 3 explains the meaning of these fields.

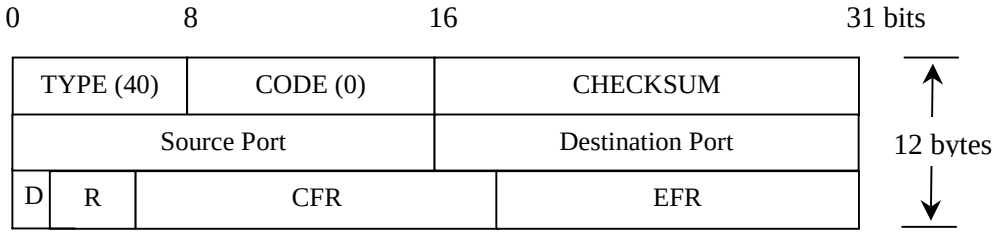

Figure 12 - ICMP probe message format

ICMP Fields	***Size***	***Description***
Type (40)	1 byte	Type 40 is used to identify the ICMP probe message (PM).
Code (0)	1 byte	Code 0 is used to indicate that the ICMP may be received from a router or host.
Checksum	2 bytes	The checksum is the 16-bit ones's complement of the one's complement sum of the ICMP message starting with the ICMP Type.
Source Port	16 bits	Source port address of TCP or UDP transport protocol.
Destination Port	16 bits	Destination port address of TCP or UDP transport protocol.
Direction (D)	1 bit	Set to 0 to indicate forward direction and 1 for backward direction ICMP PM.
Reserved (R)	3 bits	Currently unused.
Current Flow Rate (CFR)	14 bits	14 bits binary floating point representation (5 bit exponent and 9 bit mantissa)
Explicit Feedback Rate (EFR)	14 bits	14 bits binary floating point representation (5 bit exponent and 9 bit mantissa)

Table 3 - ICMP probe message fields

There are two types of ICMP PMs used in the architecture; ICMP Forward Probe Message (FPM) and Backward Probe Message (BPM). The ICMP FPM is used to convey the source's sending rate to routers along the path. The ICMP module in IP host periodically generates an ICMP FPM for every N (default $N = 256$) number of data packets sent by IP source. An ICMP FPM generated by the source IP host is initialized with its D (direction) bit, CFR (current flow rate) and EFR (explicit feedback rate) all reset to '0'. A D=0 indicates forward direction probe message. The contents of the CFR and EFR fields are processed by routers in the network. However, it is important that these values are initialised to '0' by the source IP host.

One function of the ingress edge router is to monitor incoming traffic rate of each flow and compute its current flow rate (CFR). Whenever an ICMP FPM is received by the ingress network edge router, the router updates the CFR and EFR fields with its computed rate. The CFR rate is computed based on the packets arrival rate at the edge router. When the ICMP FPM is updated, it is forwarded to the downstream core router until the ICMP FPM eventually reaches the destination IP host.

At the destination, the ICMP FPM is looped back into the network with its D bit set to '1' as ICMP Backward Probe Message. The ICMP BPM is then returned into the network towards the IP source in reverse direction to the traffic flow. When a core router receives ICMP BPM, the router first computes the max-min fair share (EFR) of the flow based on the CFR value carried in the CFR field of the ICMP BPM. The router then compares its computed max-min fair share with the value in the EFR field. If the EFR value is greater than the router's computed EFR rate, the content shall be replaced. Otherwise, the ICMP BPM is unchanged and sent to the next upstream router. The process is repeated in core routers along the backward direction until the ICMP BPM reaches the network edge router.

On receiving the ICMP BPM, the edge router computes the max-min fair share (EFR) for the flow. The edge router examines the value in the EFR field and compares with its computer EFR rate. The value in the EFR field will be replaced if it is greater than the router's computed EFR rate. Otherwise, the EFR field is not modified. The ICMP BPM is then sent to the IP source.

When an ICMP BPM for a flow is returned to the IP source, it is processed by a feedback handler. The feedback handler determines which flow does the ICMP BPM refers to by consulting the FLT. The handler then reads the content of EFR field and sent the EFR value to the corresponding traffic shaper for the flow.

6 List of parameters in the E2E protocol specifications

In this section, we provide a summary of all parameters required in the End-to-End IP Rate Control architecture. A description for each parameter in provided including its unit used and recommended range. The last column in the summary Table 4 (below) specifies the component and its location in the architecture.

Parameter	Description	Units and Range	Component/ Location
Tout	Time-out period before a connection is assumed inactive. A flow with expired Tout will be deleted from the FLT entry in the classifier.	Minutes Range: (1 - 2)	Classifier / IP host and Edge Router
CFR	Estimated current flow rate of an IP source.	Bits per second	Rate meter / Edge router
EFR	Computed explicit feedback rate (*bottleneck fairshare*) of a flow.	Bits per second	Feedback engine/ All routers
N	Maximum number of data packets a source may send for each forward direction ICMP FPM packet.	Power of 2 Range: (2 - 256)	IP host
MBR_{ts}	Maximum Bit Rate defines the maximum rate a source may transmit in a flow.	Bits per second	Traffic shaper / IP host
MBR_{tp}	Maximum Bit Rate defines the maximum rate of traffic in a flow that is allowed into the network.	Bits per second	Traffic policer / Edge router
MBS_{tp}	Maximum Burst Size defines the maximum number of packets that are allowed to accumulate in the bucket.	Bits 1-2 packets size (in bits)	Traffic policer / Edge router
K	A constant value coefficient of the exponential averaging function used to compute the incoming rate of a flow and aggregated rate of all flows in a router.	Milliseconds Range: (100 - 500)	Rate meter (Traffic policer) / Edge router & Rate estimator / Core routers
CF	Constrained flag	Boolean	Feedback engine / Edge router

Table 4 - Summary of parameters

References

[1] Abdul Aziz Mustafa and Mahbub Hassan, “End to End IP Rate Control”, accepted for publication in 8th International Conference on Advanced Computing and Communications (ADCOM2000), to be held in Cochin, India December 14-16 2000. Available online at http://www.cse.unsw.edu.au/~mahbub/PUBS/adcom00.pdf.

[2] Thomas M. Chen, Steve S. Liu and Vijay K. Samalam, “The Available Bit Rate Service for Data in ATM Networks”, IEEE Communications Magazine, May 1996.

[3] J. Postel, “Internet Protocol”, STD 5, RFC 791, September 1981.

[4] S. Deering and R. Hinden, “Internet Protocol, Version 6 (IPv6) Specification”, RFC 2460, December 199

[5] W.R. Stevens, " Unix Network Programming- Networking APIs: Sockets and XTI", Vol. 1 Second Edition, 1998, Chap. 2, pp. 32-33.

[6] ATM Forum, ATM User-Network Interface Specification Version 4.0, af-sig-0061.000, July 1996.

[7] I. Stoica, S. Shenker and H. Zhang, “Core-Stateless Fair Queuing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks”, in Proc. of ACM SIGCOMM, September 1998, pp.118-130.

[8] J.M. Jaffe, “Bottleneck Flow Control”, in IEEE Transactions on Communications, Vol. 29, No. 7, July 1981, pp. 954-962.