Design and Analysis of Flow Aware Load Balancing Mechanisms for Multi-Service Networks

Andrés Ferragut*, Daniel Kofman*, Federico Larroca* and Sara Oueslati†
*TELECOM ParisTech - Paris, France - Email: {ferragut, kofman, larroca}@telecom-paristech.fr
†France Télécom R&D - Paris, France - Email: sara.oueslati@orange-ftgroup.com

***Abstract*—Ethernet technology is being deployed in metro and wide area networks. However, despite recent evolutions, Ethernet cannot be considered a carrier class technology due mainly to the lack of facilities for efficient traffic engineering (TE). In this article, we propose a Flow-aware TE approach for carrier class Ethernet networks providing services like those defined by the Metro Ethernet Forum. The flow-aware networking approach we consider is based on the Cross-protect router mechanisms, allowing satisfactory quality for streaming and elastic flows without explicitly identifying traffic classes by the combined use of fair queuing and admission control. Our proposal applies specifically to architectures where a tunnel is established between ingress and egress nodes (like Ethernet over MPLS). In this specific context, the Cross-protect mechanisms are applied on a per tunnel basis. We analyze the proposed approach in terms of flow blocking probabilities for which explicit formula are derived. We also extend the study framework to the case where several paths are established between a pair of ingress/egress nodes, and propose a simple load balancing scheme. We analyze its performance and derive approximate formula for the flow blocking probability in this case. The analysis is validated by extensive simulations.**

***Index Terms*—Ethernet, Traffic Engineering, Flow-Aware Networking, Load balancing.**

I. Introduction

Current evolutions of network architecture show an increasing trend towards Ethernet technologies, which are expanding from small LANs to metro and wide area networks. The MetroEthernet Forum [1] is presently working on the definition of metropolitan Ethernet services, which the network operator should offer to its customers. An Ethernet service consists of a Ethernet service type, one or more Ethernet service Attributes, and one or more parameter values associated with each Ethernet Service Attribute. Examples of Ethernet service types are Ethernet Line Service (E-Line Service) and the Ethernet LAN Service (E-LAN Service). One of the most critical Ethernet Service Attribute is the Bandwidth Profile. This defines how much traffic the customer can send or receive, and is defined in terms of two token buckets. The parameters that configure these token buckets, as well as the performance objectives are part of the agreement between the service provider and the customer (the Service Level Agreement, SLA).

Studies have been carried out within both the IEEE [2], [3] and the IETF [4] standardization bodies to define architectures that provide such services. These approaches rely on "tunnels" (i.e., connection-oriented) to transport Ethernet frames either natively (e.g. PBT, GELS) or using MPLS (e.g. PWE3, VPLS). In such schemes, the use of token buckets as traffic descriptors for SLA specifications has been strongly criticized in the past. In particular, in [5] they have been described as inefficient and inappropriate for characterizing traffic aggregates. This *a priori* traffic descriptor is a very poor characterization of the actual traffic, which leads to the customer to systematically overestimate the traffic parameters. This means that their declared values are of little use for resource allocation.

We propose an alternative Flow Aware TE approach for carrier class Ethernet networks providing services as those defined by the Metro Ethernet Forum. We assume that it is possible to associate a capacity to a given "tunnel" using for instance RSVP-TE in the context of MPLS, or GMPLS for native Ethernet. We further assume that multiple paths could be established between pairs of ingress/egress nodes.

We propose that provider edge routers implement the Cross-protect mechanisms [6], [7] on a per tunnel (or Label Switch Path, LSP) basis. Cross-Protect allows performance guarantees for streaming and elastic flows while preserving the user-network interface of the best effort Internet, i.e., neither packet marking nor per-flow signalling is required to differentiate explicitly between streaming flows, requiring low packet loss and delay, and elastic flows, requiring "as fast as possible" document transfer. Instead, Cross-protect routers identify user-defined flows on the fly and implement per-flow scheduling using Priority Fair Queueing (PFQ). Measurement-based admission control is additionally employed to maintain the fair rate sufficiently high to provide streaming like quality for flows of peak rate up to a chosen threshold.

The adopted flow-aware networking approach provides precise QoS guarantees (suitable for SLA specifications) and is more cost-effective than substantially over-provisioned best-effort networks. Since resources are reserved on a per LSP basis, the Cross-protect mechanisms need only be implemented at the network edge (i.e., transparent to internal nodes). This approach is studied analytically and explicit formula for the flow blocking rate are derived.

We next extend the ingress TE scheme with a simple flow-aware load balancing algorithm, providing greater resilience (enforced fairness, overload control) and potentially better resource utilization. We evaluate its performance analytically and approximative formula are obtained and validated by means of simulations. Simulation results also show the gain achieved with our dynamic load balancing scheme over static load balancing algorithms.

The paper is organized as follows. Section II describes in details the proposed TE mechanisms in the context of the aforementioned architectures. Section III presents a model for evaluating the main QoS parameter in the case where a single LSP is established per ingress/egress node pair. Section IV-A mentions related work on load balancing, and Section IV-B describes the proposed load balancing scheme for the case where multiple LSPs are available. In Section IV-C we introduce the related analytical model and also give approximative formula for the main QoS indicators. Simulations results and comparative analysis with other load balancing mechanisms are presented in Section V. We conclude the paper in Section VI.

II. Proposed Architecture

A. *Cross-protect*

IP traffic can be broadly classified into two categories with clearly different QoS requirements: *elastic* and *streaming*. Elastic traffic is generated by document transfers (e.g., web page, MP3 music file). Associated Elastic flows require "as fast as possible" transfers. Streaming traffic, on the other hand, is produced by real time audio and video applications (e.g. video streaming, VoIP conversation), and requires transparent delivery, i.e., low packet loss rate and delay.

IP networks are meant to support all kinds of service, each coming with its own specific requirements. These are almost always met in present IP backbone networks, in particular, thanks to substantial overprovisioning. Indeed, the inability of network operators to distinguish between kinds of traffic leads to undiscriminated handling of streaming and elastic packets (FIFO queueing). Of course, explicit marking using Diffserv would be possible, but this comes at a certain cost and raises several other issues such as trust in an inter-domain setting.

Overprovisioning is actually a quite satisfactory solution, as it satisfies most users' requirements, while inducing very low operational costs. However, the network remains vulnerable to ill-behaved users, as delivering reasonable QoS depends on users' cooperation in implementing end-to-end congestion control (TCP or alternative "TCP-friendly" protocols). More to the point, the network is oblivious to specific requirements relative to different kinds of traffic. In particular, in the event of an overload situation due, for instance, to a link or equipment failure (as overprovisioning may not take into account all kinds of risk), all traffic is likely to suffer QoS degradation, including critical applications such as voice or TV broadcast.

Moreover, following [5], we consider that traffic engineering and traffic control for predictable QoS is most conveniently performed at *flow* level, rather than *packet* or *aggregate* level. A flow corresponds to some application instance, transported by the network. It is precisely at this level that the user perceives QoS. A flow may, for instance, correspond to a web page download, a voice call, or a music or video streaming. Although in practice a more exact definition is needed, for the present study purpose, it is sufficient to define a flow as a stream of packets sharing common header attributes (e.g. the TCP/IP 5-tuple, or the IPv6 flow label combined with the source or destination address) and a maximum inter-packet time.

According to [5], [8]–[10], the integration of both streaming and elastic flows can be achieved without deteriorating their respective QoS, as long as bufferless multiplexing conditions are assured for streaming flows (handled with priority) and remaining resources are fairly shared between elastic flows. A possible way to achieve this integration is to use *Cross-protect*, an implementation of the so called *Flow Aware Networking* described in [6]. A Cross-protect router consists of two traffic control components. On the one hand, a Priority Fair Queueing (PFQ) scheduler, which is a simple adjustment of a fair queueing scheduler (e.g., Start time Fair Queueing (SFQ) [11] or Deficit Round Robin (DRR) [12]), that implicitly differentiates between streaming and elastic flows. On the other hand, an admission control mechanism that guarantees a minimum QoS to accepted (or protected) flows, as well as the scalability of the scheduler by limiting the number of flows that need to be handled by the scheduler at any given time.

The PFQ scheduler implicitly classifies flows as streaming or elastic on the fly based on the following principle. If a streaming flow is transmitting at rate less than the current instantaneous fair-rate (in practice, this is manifested by the flow being not backlogged), then its packets are classified as streaming and given priority, whereas elastic flows (backlogged flows) share the remaining capacity in a processor sharing (PS) way. Hence, fair sharing is *enforced* by the queueing discipline, and does not rely on the TCP friendliness of the congestion control protocol implemented by end users.[1]

Admission control is used to limit the streaming load (referred to as *priority load* in the remainder, and denoted PL) to be under a relative threshold γ_s, and the current rate obtained by elastic flows (i.e., *fair rate*, denoted in the sequel FR) above another threshold γ_e. If any of these conditions is not satisfied, new flows are blocked. This way, a minimum bandwidth for elastic flows is guaranteed, and the load induced by streaming traffic is controlled. Typical values for γ_s and γ_e are around 80%, and 1%, respectively.

A simplified diagram of the mechanism is illustrated in Figure 1. It is worth noting that the implicit classification is continuously applied to all ongoing flows and not only upon flow arrival to the router. This means that, as the rate of flows evolves, their classification may change too. For instance, the first few packets of a TCP connection in the Slow Start mode are typically assimilated to streaming packets.

A concern present in all flow-level scheduling mechanisms realizing fair bandwidth sharing is scalability. As discussed in [13], the complexity of such schedulers depends on the number of bottlenecked flows, and not in the number of active ones. Although the latter increases with the link capacity, the former does not and remains small (in the orders of tens). In PFQ in particular, another possible concern is the apparent need of per-flow calculations for the implicit flow classification. However,

[1]However, TCP is required for the elastic traffic to *find* its fair rate.

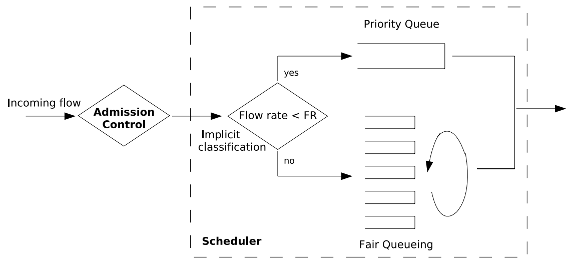

Fig. 1. Cross-Protect diagram

per-flow rate measurements are not made since the scheduling algorithm implicitly gives priority to flows transmitting at less than the fair-rate. FR and PL are periodically measured to be used for admission control purposes only. We refer the interested reader to [7] for more details.

Our proposal for TE in the MetroEthernet architectures is to implement Cross-protect in the edge routers (no impact on core routers). Since edge routers are connected via "tunnels" of a given capacity, overload control (in the form of admission control) can be performed at ingress nodes only. This has the advantage of restricting flow-aware operations to the network edge, and preserving the simplicity of packet forwarding in the core. An example of such a TE architecture, in the case of MPLS tunnels, is illustrated in Figure 2. The Bandwidth Profile can be now specified with the total path capacity and both thresholds.

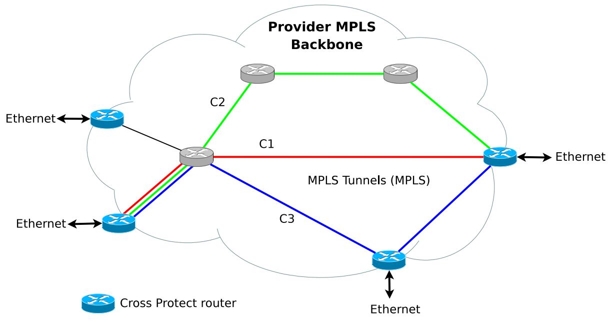

Fig. 2. Proposed architecture

B. Load Balancing

To improve performance and enhance resilience to sudden traffic fluctuations and failure induced overload events, dynamic load balancing can be used by connecting an ingress-egress pair by more than one virtual path [14], [15]. Figure 2 illustrates how several LSPs can connect a pair of edge nodes. Such methods allow the network to respond to changing demands and failures by changing the routing pattern depending on the state of the paths. Since the fair rate and the priority load of each LSP are measured for admission control purposes, it is simple and natural to route each flow depending on these measurements.

A large body of work exists on load-balancing [16]–[20]. These papers propose multi-path architectures which are robust to changes in the traffic matrix and resilient to link and node failures, but they do not specify any load-balancing algorithm in particular. Our load-balancing scheme is flexible enough to be used in any of these architectures too.

In the following sections we first analyze an isolated Cross-protect router (i.e., single path routing). We next propose and analyze a load balancing policy for the case when there are several possible tunnels (or LSPs)[2] between source and destination. We provide possible approximations for the most important QoS parameter for a Cross-protect router, namely the blocking probability. Accepted flows have their other QoS indicators (e.g. the throughput, for elastic flows) guaranteed to a certain minimum by means of admission control.

III. Single Tunnel Analysis

A. Model Description

We model the arrival of elastic flows into our system as a Poisson process of intensity λ_e. Each flow of this type is characterized by the workload offered to the system, distributed as σ_e. We denote by $\omega_e = \mathbb{E}(\sigma_e)$ the mean offered workload and by $\rho_e = \lambda_e \omega_e$ the elastic load in the system. Typically, a reasonable choice for the distribution of σ_e will be heavy tailed, where most flows are very small while the majority of traffic is contained in very long flows.

Streaming flows are modeled as circuit type traffic in telephone networks, and are therefore characterized by an intrinsic rate r (that we suppose fixed so as to simplify the analysis) and a random duration, with mean d_s. We further assume that these flows arrive also as a Poisson process of intensity λ_s. The mean workload introduced by streaming traffic is then rd_s, thus $\rho_s = \lambda_s d_s r$ represents the streaming load of the system.

Suppose C is the virtual path (or LSP) capacity. At the flow time scale, we suppose that C is fixed. The described model for the system can be represented as a PS network of queues with state dependent service rate as shown in Figure 3. We denote by $x = (x_s, x_e)$ the number of ongoing streaming and elastic flows, respectively.

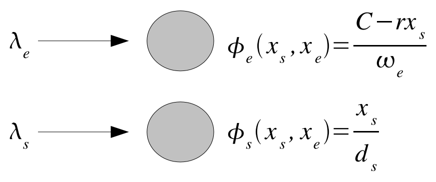
λ_e
$\phi_e(x_s,x_e)=\frac{C-rx_s}{\omega_e}$
λ_s
$\phi_s(x_s,x_e)=\frac{x_s}{d_s}$

Fig. 3. PS network representing a Cross-protect router

One node represents the streaming service component, with arrival rate λ_s and service rate $\phi_s(x_s, x_e) = x_s/d_s$, which is the service rate of an Erlang system. The other node represents the elastic service component, with arrival rate λ_e and service rate $\phi_e(x_s, x_e) = (C - rx_s)/\omega_e$, which is the service rate of a PS queue with the workload characterized by $\mu_e = 1/\omega_e$

[2]In the sequel, the terms tunnel, virtual path, and LSP (Label Switch Path) are used without differentiation.

and a variable capacity, that depends of the current available bandwidth left by the streaming flows (we are supposing that elastic flows realize their fair rate instantly).

In terms of the state x, the admission control conditions introduced in the previous subsection constrain the state space, which is given by:

$$rx_s \leq \gamma_s C \quad \text{and} \quad \frac{C - rx_s}{x_e} \geq \gamma_e C \tag{1}$$

It is important to observe that the implicit classification of the cross-Protect router is not modeled. This means that in our model flows are known to be elastic or streaming a priori, which is valid as long as all streaming flows' transmission rate r remains smaller than the fair rate.

B. Analysis

We now focus on calculating the blocking probability, which is the main performance indicator in the Cross-protect architecture.

The general analysis of the integrated model as presented before is not feasible, even if we make the simplifying hypothesis of exponential workloads. Instead, we apply the same idea as in [21] and assume that we can separate the time scales of streaming and elastic flows. This allows for the study of the elastic queue as if the streaming queue were in the stationary regime (also known as the Quasi-Stationarity (QS) assumption).

The justification behind the QS assumption is the following: in general, the mean duration (d_s) of streaming flows is much greater than the corresponding mean duration of elastic flows. For a given load level ρ_s, ρ_e, this means that the arrival rate of streaming and elastic flows verify that $\lambda_s \ll \lambda_e$. This relationship implies that the events associated to streaming flows occur sparsely in time, and allows the elastic queue to behave in a *quasistationary regime*, under which the elastic flows see the queue as a constant rate server.

The quasi-stationary assumption allows us to analyze the elastic queue as an $M/G/1-PS$ queue with capacity $C-rx_s$ for x_s streaming flows present in the system. As a consequence, we can write the probability of x_e elastic flows in the system, given there are x_s flows in the streaming queue, as follows:

$$P(N_e = x_e | N_s = x_s) = \frac{1 - \rho_e(x_s)}{1 - \rho_e(x_s)^{N_e^{max}(x_s)+1}} \rho_e(x_s)^{x_e} \tag{2}$$

for $0 \leq x_e \leq N_e^{max}(x_s)$, where:

$$\rho_e(x_s) = \frac{\lambda_e \omega_e}{C - rx_s} \tag{3}$$

$$N_e^{max}(x_s) = \left\lfloor \frac{C - rx_s}{\gamma_e C} \right\rfloor \tag{4}$$

We can therefore write the blocking probability conditioned to x_s from (2) by making $N_e = N_e^{max}(x_s)$:

$$B_e(x_s) = \frac{1 - \rho_e(x_s)}{1 - \rho_e(x_s)^{N_e^{max}(x_s)+1}} \rho_e(x_s)^{N_e^{max}(x_s)} \tag{5}$$

To derive the blocking probability under stationary regime, we must average conditional blocking probability $B_e(x_s)$ with respect to the stationary distribution $\pi_s(x_s)$ of the streaming queue:

$$B = \sum_{x_s=0}^{N_s^{max}} B_e(x_s) \pi_s(x_s) \tag{6}$$

where $N_s^{max} = \lfloor \gamma_s C / r \rfloor$.

The streaming queue would behave exactly like an Erlang one with $A_s = \lambda_s d_s$ load (in Erlangs), if streaming flows were only rejected due to the priority load condition. But this is not the case, since both conditions on priority load and fair rate are applied independent of the traffic type. So, the resulting process is a birth-death one, with the birth rate equal to $\lambda_s(1 - B_e(x_s))$ (to account for the blocking state of the elastic queue) and the death rate is equal to $x_s \mu_s$ in state x_s. The stationary distribution then becomes:

$$\pi_s(x_s) = \pi_s(0) \frac{A_s^{x_s}}{x_s!} \prod_{i=0}^{x_s-1} (1 - B_e(i)) \tag{7}$$

where $\pi_s(0)$ can be obtained from the normalization condition.

Equations (5), (6) and (7) then give the blocking probability B of the system.

IV. Multiple Tunnels and Load balancing Analysis

A. Related work

Let us consider the case where there is only elastic traffic present. In this situation, the problem becomes a routing problem between multiple processor sharing queues in parallel. This problem was first studied by Bonomi (see [22] and references therein), and more recently by Koole et. al. [23]. Their results show that the optimal policy, in terms of blocking probability and throughput, based on present state information is to send the arriving flows to the shortest queue (*Join the Shortest Queue*, JSQ) for the case of two identical servers and exponential workloads.

For the general setting, i.e., general workloads and non symmetric systems, the problem is open and the optimal policy is unknown. Hajek in [24] analyzed a more general system allowing different capacities under a markovian setting. He showed, based on dynamic programming approach, that the optimal policy is always switch type, but explicit formula for the switch curve were not provided. We performed numerical calculations to approximate the switch curve of the Hajek model in a routing setting. The result for exponential workloads, in the two server case, is that the optimal policy in terms of throughput is given by the *exact greedy policy* (EGP):

$$\text{Route to route } i \Leftrightarrow i = \arg\max_j \frac{C_j}{x_j + 1} \tag{8}$$

This is a greedy policy, which maximizes the fair rate that a flow will receive upon entering service in the system. In the symmetric case, this boils down to the JSQ policy.

Another approach to load balancing in the elastic case is studied by Jonckheere et. al [25]. The authors propose another class of policies, the *balanced routing*, with the property that they are insensitive to the distribution of the flow workloads. In the class of balanced routing policies, the optimal in terms of blocking probability is defined and is characterized by the following probabilities:

$$\text{Route to queue } i \text{ with prob. } p_i(\overrightarrow{x}) = \frac{n_i - x_i}{\sum_j n_j - x_j} \tag{9}$$

where x_j is the number of customers in the $j-th$ queue and $\overrightarrow{n} = (n_1, \ldots, n_N)$ is the maximum number of customers allowed in each queue. We shall call this policy the *optimal balanced policy* (OBP). The blocking probability B_{obp} in this case can be easily computed.

B. Proposed scheme

We now address the load balancing issue. In the proposed architecture described in II-A, we assume that each pair of ingress-egress nodes are connected via one or multiple LSPs, where the ingress node implements Cross-protect mechanisms to realize QoS. From the flow perspective, this means that there are N virtual paths with capacities $C_1, \ldots, C_N$, that are fixed at the flow time scale. The load balancing between these virtual paths becomes then a routing problem between multiple queues. Note that these queues can be treated as equal choices in terms of resources, that is, the objective here is to minimize the flow blocking probability by choosing the right routing policy. At this time scale, we are not concerned with minimizing the number of physical links composing each route (e.g., via shortest path routing), as resources associated to virtual paths are supposed reserved in advance.

In the Cross-protect setting, where the current fair rate is already measured, we propose to use the following simplified policy, that we call *simplified greedy policy* (SGP):

$$\text{Route to queue } i \Leftrightarrow i = \arg\max_j \, fair\_rate_j \tag{10}$$

This policy is clearly intuitive and is also very simple to implement in the ingress node, not requiring additional measurements. It will also "inherit" the optimality of EGP and, as will be outlined later, OBP. However, corresponding analytical results are hard to derive and some approximations will be needed.

We now focus our attention on the streaming flow long time scale. These flows do not gain any particular advantage from our SGP policy, which is devised to work at elastic flow timescale. However, streaming flow routing is less critical because it represents the minority of traffic. We sacrifice optimality at the streaming time scale in favor of an implicit policy, which makes no distinction in the type of arriving flows. It must be noted once again that the optimality here refers to the blocking probability in the routing case, the other QoS requirements associated with streaming traffic (i.e., low packet loss and delay) are guaranteed by priority handling and admission control.

C. Analysis

We model and analyze the load balancing between paths of equal or different capacities. To clarify the exposition, we will only present here in full detail the load balancing between two paths. The case in which more paths are present can easily be derived from the results presented.

We will assume the two thresholds ($\gamma_{si}C_i$ and $\gamma_{ei}C_i$) are equal in both LSPs. Given the two paths are equivalent, is reasonable to consider these two values to be the same, since they give the minimum QoS each protected flow will receive.

The analysis will be the same as in the isolated case. We will first analyze what happens in the short time scale of elastic flows, estimate the blocking probability given the value of x_{s1} and x_{s2} and then estimate the probability of having x_{s1} and x_{s2} streaming flows in the system so as to make the weighted sum.

For the analysis to load balancing in the elastic case, we turn our attention to the aforementioned work of Jonckheere et al. [25]. In this case, the blocking probability when using OBP can be computed as $B_{obp} = 1/\delta(\overrightarrow{n})$ where δ is given by a simple recursive formula:

$$\delta(\overrightarrow{x}) = 1 + \sum_{i=1}^{N} \frac{\phi_i(x_{si}, x_{ei})}{\lambda_e} \delta(\overrightarrow{x} - \overrightarrow{e_i}) \tag{11}$$

where

$$\phi_i(x_{si}, x_{ei}) = \frac{C_i - x_{si} r}{\omega_e}$$
$$\overrightarrow{x} - \overrightarrow{e_i} = (x_1, \ldots, x_i - 1, \ldots, x_N)$$
$$\delta(\overrightarrow{0}) = 1 \quad \text{and} \quad \delta(\hat{x}) = 0 \quad \forall\, \hat{x} \text{ invalid, e.g. } \hat{x} = (0, -1)$$

The usefulness of optimal balanced routing is that, as verified in [25], the equations above provide a very good approximation for the behavior of the greedy policies. The blocking probability for different parameters of load and workload distribution provide a good approximation of the corresponding ones in the SGP. Moreover, the calculations rely on simple parameters of the input traffic, like the arrival intensity and mean workload, making design calculations easier. We shall use this as an approximation for the loss probability given x_{s1} and x_{s2}:

$$B_e(x_{s1}, x_{s2}) = \frac{1}{\delta(N_{e1}^{max}, N_{e2}^{max})} \tag{12}$$

Note that, although OBP was used as an approximation in our analysis, we believe that the integration of this policy into Cross-protect routers wouldn't be as easy as the proposed one. Our proposal does not require any additional measures and the only operation necessary is to select the path with the highest fair rate. The few number of them limits the complexity of the path selection operation. On the other hand, OBP requires a random assignment with weights which have to be calculated each time a new flow arrives.

Now the only remaining issue is to find the steady-state probability of the streaming queues. The main effect of the policy in the system is to maintain equal *fair rates* in each queue, which can be calculated as C_1/x_{e1} and C_2/x_{e2} respectively. If we equal the two fair rates then $x_{e1}/x_{e2} = C_1/C_2$.

So the proportion of flows to the i-th queue is approximately $C_i/(C_1+C_2)$. Since streaming flows receive the same treatment as elastic ones, they are handed to each queue in the same proportion. This is obviously a gross approximation, but our simulations indicate that this proportion is very accurately maintained for the streaming flows.

Once again, we could calculate $\pi_s(x_{s1}, x_{s2})$ like an Erlang double queue (where each queue can be treated independently, leading to a product form probability) if we assume that the only reason for rejecting streaming flows is the *PL* condition. This is not the case, since streaming flows are rejected due to the *FR* condition too. So, analogous to the single server case, the arrival rate to each streaming queue is finally:

$$\lambda_{si} = \frac{C_i}{C_1+C_2}\lambda_s(1-B_e(x_{s1},x_{s2}))$$

Thus, the streaming part of the system behaves as a birth-death process with transition rates:

$$\begin{aligned} x_s \to x_s + e_i \quad &: \quad \frac{C_i}{C_1+C_2}\lambda_s(1-B_e(x_{s1},x_{s2})) \\ x_s \to x_s - e_i \quad &: \quad \mu_s x_{si} \end{aligned}$$

The blocking probability couples the behavior of the queues, making the exact analysis of the system not tractable. To get approximate values of the steady-state probability, we have to numerically solve the global balance equations ($\pi_s Q = 0$ plus the normalization condition $\sum_i \pi_s(i) = 1$). An alternative is to find an upper bound for the system by finding an easier system to analyze with a conservative behavior (i.e. yielding higher blocking probabilities). In this case, such a system is one which tends to hold more customers in the queue than the original one. This can easily be done by simply making the birth-rate bigger. For instance, if we change the original transition rates to:

$$\begin{aligned} x_s \to x_s + (1,0) : &\frac{C_1}{C_1+C_2}\lambda_s\left(1 - \arg\min_{x_{s2}} B_e(x_{s1},x_{s2})\right) \\ &= \lambda_s(1-B_e(x_{s1},0)) \\ x_s \to x_s + (0,1) : &\frac{C_2}{C_1+C_2}\lambda_s\left(1 - \arg\min_{x_{s1}} B_e(x_{s1},x_{s2})\right) \\ &= \lambda_s(1-B_e(0,x_{s2})) \\ x_s \to x_s - e_i \quad : &\quad \mu_s x_{si} \end{aligned}$$

This system has the advantage that a product-form holds for its stationary distribution and since it is balanced, it is also insensitive [25]. Its steady-state probability can be obtained by analyzing the system as if each queue were independent, so the joint probability is simply the multiplication of both probabilities. So, the complete system can be bounded as:

$$\pi_s(x_{s1},x_{s2}) \underset{st}{\leq} \tilde{\pi}_s(x_{s1},x_{s2}) = \pi_{s1}(x_{s1})\pi_{s2}(x_{s2}) \qquad (13)$$

where

$$\frac{\pi_{s1}(x_{s1})}{\pi_{s1}(0)} = \left(\frac{C_1}{C_1+C_2}A_s\right)^{x_{s1}} \frac{1}{x_{s1}!}\prod_{j=0}^{x_{s1}-1}(1-B_e(j,0)) \qquad (14)$$

$$\frac{\pi_{s2}(x_{s2})}{\pi_{s2}(0)} = \left(\frac{C_2}{C_1+C_2}A_s\right)^{x_{s2}} \frac{1}{x_{s2}!}\prod_{j=0}^{x_{s2}-1}(1-B_e(0,j)) \qquad (15)$$

where $\pi_{s1}(0)$ and $\pi_{s2}(0)$ are obtained from the normalization condition.

The $\underset{st}{\leq}$ means that the balanced system will tend to have more customers in the queue than the original one, thus it will have a higher average number. So, the blocking probability using this system will be an upper bound of the original one. Finally, with equation (12) and (13) we can bound the total blocking probability as:

$$B = \sum_{x_{s1},x_{s2}} \pi_s(x_{s1},x_{s2})B_e(x_{s1},x_{s2}) \leq \sum_{x_{s1},x_{s2}} \tilde{\pi}_s(x_{s1},x_{s2})B_e(x_{s1},x_{s2}) \qquad (16)$$

The tightness of the upper bound depends on the difference between the maximum and minimum blocking probability $B_e(x_{s1},x_{s2})$. As we shall see in the next section, in our simulations the upper bound can be as much as twice as the simulated blocking probability. But this happens only for high load values, where this difference is bigger. As the load decreases, the upper bound gets tighter.

V. Simulation results

A. Flow-level (or fluid) simulations

Several approximations were made in the SGP analysis. First, we considered that TCP flows reach their fair-rate instantly. Then, we made suppositions on the arrival rate at each server and used OBP to calculate the blocking probability for elastic flows. To verify that the fluid system is still very well represented even with these last simplifications, we developed a flow-level simulator which implements a Cross-Protect router.

A comparison between simulation and analytical results for the two server case can be seen in figure 4. In the simulation, the capacity of a link is 1 and the other 2. The total traffic is a mix of elastic traffic whose workload follows an exponential distribution with mean 0.5 and streaming traffic whose rate is 0.05. Streaming traffic represents 20% of traffic. In the graphs, there are two different analytical plots. One drawn using the product form bound $\tilde{\pi}_s(x_{s1},x_{s2})$ (eq. (13)) and the other one using the numerical solution of the global balance equations. It can be seen how the product form estimation is an acceptable upper bound for the real blocking probability. It is also clear that the numerical solution is a very tight approximation, which means that the proposed Markov chain accurately models the system.

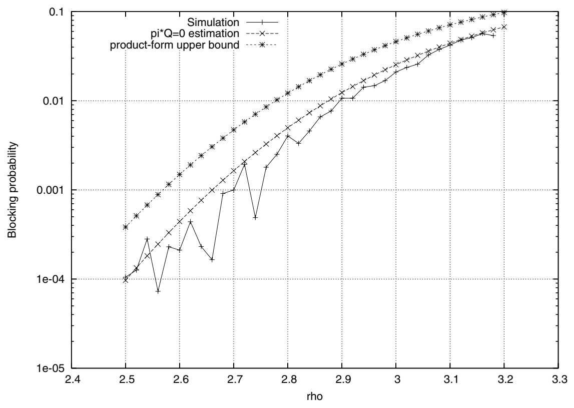

Fig. 4. Blocking probability estimation and simulation results for asymmetric server case

B. Packet-level simulations

In order to further verify our analysis and compare our load balancing scheme with others, we conducted several packet level simulations using ns-2 [26], with the cross-protect implementation used in [7].

A comparison between the blocking probability obtained by simulation and the corresponding estimation can be seen in figure 5 for the single-server case. The case scenario is a mix of elastic traffic whose workload follows a Pareto distribution with mean 20kB and streaming traffic with a fixed rate of 10kbps. Streaming traffic represents 20% of traffic and the channel has a total capacity of 1Mbps.

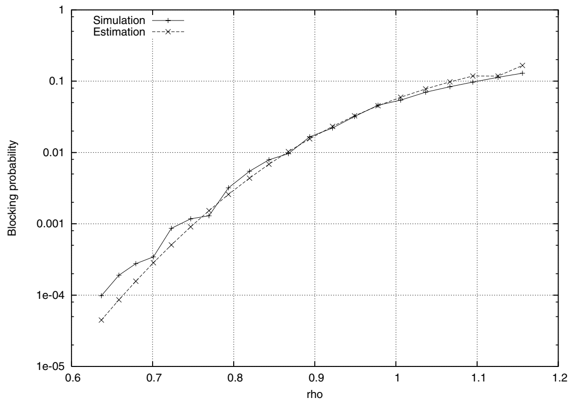

Fig. 5. Blocking probability estimation and simulation results for an isolated cross-protect router

Although the analysis did not take into account packet level dynamics of TCP (e.g., Slow Start), the estimation proves to be very accurate. However, in the presence of TCP packet level dynamics, the implicit classification system assimilates part of TCP traffic to streaming traffic, notably during the Slow Start phase. This results in more priority traffic than expected by the model. If the probability that a flow is blocked due to the PL condition is negligible, then the estimation model will yield accurate predictions. Otherwise, the model tends to underestimate the blocking probability.

A comparison between simulation results and analytical results for the two server case can be seen in Figure 6. The considered scenario is the same as the previous one, but with a different elastic load distribution. The figure features two different simulation plots: one is relative to an exponential elastic flow size distribution, the other plot corresponds to a Pareto distribution. Results show that the scheme is a little bit sensitive to the distribution, all the more as load increases. As the load decreases, blocking probability tends to be the same for both. The graph also shows that the upper-bound is, in this case, actually an approximation. As we saw, in the flow-level simulations it was a strict upper-bound. This increase in the expected blocking probability is then due to the dynamic behavior of TCP.

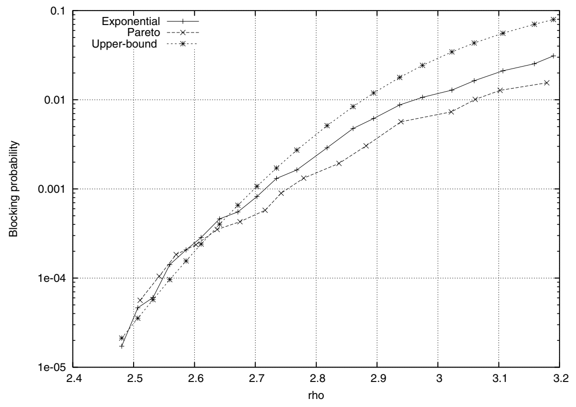

Fig. 6. Blocking probability estimation and simulation results for SGP

We now compare the proposed scheme with other possible load balancing techniques. The most widely used one today is to assign each incoming flow to a random path, where the probability to be sent to the $i-$th queue is $C_i/(\sum C_j)$. In the symmetric case, there is no great advantage in using the proposed scheme. But in the asymmetric case there will be a great gain in a load balancing scheme that takes into account current state of each path. In figure 7 we present a comparison between the two load balancing schemes. In the same example as before, it can be seen that the gain in using the proposed load balancing scheme is considerable, especially when the system is not very loaded, which should be the operation point.

VI. Conclusions

Bringing Ethernet into the Metropolitan Area Network introduces a lot of advantages to both the service provider and the customer (corporate and residential). However the lack of mature TE solutions is seriously delaying the emergence of a “carrier class Ethernet” network. In this paper, we have addressed this critical aspect of Metro Ethernet architectures.

Drawing on the work of Roberts et. al., we have discussed the application of the flow-aware networking paradigm, based on Cross-protect mechanisms, in the context of connection-oriented networks, such as MPLS. We believe this a simple and efficient alternative to the Diffserv-TE solution. Simplicity flows from the ability of differentiating streaming and elastic

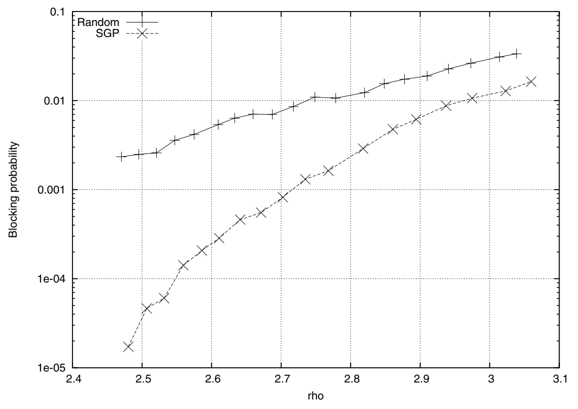

Fig. 7. Blocking probability using SGP and random load balancing

flows associated to a virtual path in an implicit manner. Hence, dispensing with the need of marking packets as in Diffserv. Efficiency, on the other hand, is the result of a better control of QoS, which is realized at flow-level rather than aggregate level (remember aggregates are difficult to characterize). Indeed, minimum QoS guarantees to streaming and elastic flows are enforced by means of admission control.

Note that the implementation of flow-aware networking is particularly interesting in a connection-oriented network, compared to a pure IP network, as Cross-protect mechanisms need only be implemented in edge routers. In fact, our scheme is not restricted to MetroEthernet only, but is applicable to any connection-oriented environment in which a certain capacity can be guaranteed to tunnels. We concentrated on MetroEthernet in particular due to its lack of efficient TE schemes. In other architectures were such schemes do exist, our proposition can be seen as a complement or a substitute.

We have evaluated the performance of the proposed QoS architecture by means of analysis and simulations. In particular, we have derived a closed-form formula for estimating the blocking probability of a Cross-Protect router, as a function of the expected elastic and streaming loads.

In order to further improve network performance and resilience, we have proposed *Simplified Greedy Policy* (SGP), a simple dynamic load balancing scheme which is rather straightforward to implement in a Cross-protect router, as it exploits fair-rate measurements used by the admission control. Our results highlight the gain achieved by SGP over the static random load assignment strategy. In the asymmetric scenario (i.e., parallel LSPs with different capacities), for instance, the static reference strategy may yield a blocking probability that is orders of magnitude higher than that obtained with SGP.

In order to evaluate the performance of a Cross-protect router implementing SGP for balancing the load induced by both elastic and streaming traffic, we have developed a simple analytical fluid model to derive an approximate expression for the blocking probability. The analytical model was verified by means of fluid simulations and packet-level simulations. Fluid simulation results nicely fit the analytical results. The comparative evaluation with packet-level simulation shows that the derived blocking probability formula constitutes a reasonable approximation, though it does no longer constitute an upper bound as in the fluid setting.

Acknowledgments

The authors would like to thank James Roberts, Jean-Louis Rougier and Paola Bermolen for the insightful discussions that enabled this paper to be written.

References

[1] "Metro Ethernet Forum," http://www.metroethernetforum.org/.
[2] "Provider Bridges," http://www.ieee802.org/1/pages/802.1ad.html.
[3] "Provider Backbone Bridges," http://www.ieee802.org/1/pages/802.1ah-.html.
[4] "VPLS - Virtual Private LAN Service," http://www.vpls.org.
[5] J. W. Roberts, "Internet Traffic, QoS and Pricing," *Proceedings of the IEEE*, vol. 92, no. 9, pp. 1389– 1399, September 2004.
[6] S. Oueslati and J. W. Roberts, "A new direction for quality of service: flow-aware networking," *Next Generation Internet Networks*, 2005.
[7] A. Kortebi, S. Oueslati, and J. W. Roberts, "Cross-protect: implicit service differentiation and admission control," in *Workshop on High Performance Switching and Routing, 2004*, pp. 56–60.
[8] T. Bonald, A. Proutiere, and J. W. Roberts, "Statistical Performance Guarantees for Streaming Flows using Expedited Forwarding," *IEEE INFOCOM*, pp. 1104–1112, 2001.
[9] T. Bonald and J. W. Roberts, "Congestion at flow level and the impact of user behaviour," *Computer Networks*, vol. 42, pp. 521–536, 2003.
[10] S. B. Fredj, T. Bonald, A. Proutiere, G. Regnie, and J. W. Roberts, "Statistical bandwidth sharing: A study of congestion at flow level," *ACM SIGCOMM*, pp. 111–122, 2001.
[11] P. Goyal, H. Vin, and H. Cheng, "Start-time fair queueing: a scheduling algorithm for integrated services packet switching networks," *Trans. on Networking, IEEE/ACM*, vol. 5, no. 5, pp. 690–704, October 1997.
[12] M. Shreedhar and G. Varghese, "Efficient Fair Queuing Using Deficit Round Robin," *IEEE/ACM Trans. Netw.*, vol. 4, no. 3, June 1996.
[13] A. Kortebi, L. Muscariello, S. Oueslati, and J. Roberts, "Evaluating the number of active flows in a scheduler realizing fair statistical bandwidth sharing," in *ACM SIGMETRICS*, 2005, pp. 217–228.
[14] A. Elwalid, C. Jin, S. Low, and I. Widjaja, "MATE: MPLS adaptive traffic engineering," *INFOCOM 2001*, vol. 3, pp. 1300–1309.
[15] S. Kandula, D. Katabi, B. Davie, and A. Charny, "Walking the tightrope: responsive yet stable traffic engineering," in *ACM SIGCOMM '05*, pp. 253–264.
[16] T. Erlebach and M. Ruegg, "Optimal bandwidth reservation in hose-model vpns with multi-path routing," *IEEE INFOCOM*, 2004.
[17] M. Kodialam, T. Lakshman, and S. Sengupta, "Efficient and robust routing of highly variable traffic," in *HotNets-III*, 2004.
[18] R. Zhang-Shen and N. McKeown, "Designing a predictable internet backbone with valiant load-balancing." in *IWQoS*, 2005, pp. 178–192.
[19] F. B. Shepherd and P. J. Winzer, "Selective randomized load balancing and mesh networks with changing demands," *Journal of Optical Networking*, vol. 5, pp. 320–339, 2006.
[20] M. Kodialam, T. Lakshman, and S. Sengupta, "Maximum throughput routing of traffic in the hose model," in *INFOCOM 2006*.
[21] N. Benameur, S. B. Fredj, F. Delcoigne, S. Oueslati-Boulahia, and J. W. Roberts, "Integrated admission control for streaming and elastic traffic," *Lecture Notes in Computer Science*, vol. 2156, pp. 69–81, 2001.
[22] F. Bonomi, "On job assignment for a parallel system of processor sharing queues," *IEEE Trans. Comput.*, vol. 39, no. 7, pp. 858–869, 1990.
[23] G. Koole and A. Hordijk, "On the Assignment of Customers to Parallel Queues," *Probability in the Engineering and Informational Sciences*, vol. 6, pp. 495 – 511, 1992.
[24] B. Hajek, "Optimal control of two interacting service stations," *IEEE Trans. on Automatic Control*, vol. 29, no. 6, pp. 491– 499, June 1984.
[25] T. Bonald, M. Jonckheere, and A. Proutiere, "Insensitive load balancing," *Proceedings of the joint international conference on Measurement and modeling of computer systems*, pp. 367–377, 2004.
[26] "The Network Simulator - ns," http://nsnam.isi.edu/nsnam/index.php.