Leveraging the Rate-Delay Trade-off for Service Differentiation in Multi-Provider Networks

Maxim Podlesny, *Member, IEEE,* and Sergey Gorinsky, *Member, IEEE*

***Abstract*—The single best-effort service of the Internet struggles to accommodate divergent needs of different distributed applications. Numerous alternative network architectures have been proposed to offer diversified network services. These innovative solutions failed to gain wide deployment primarily due to economic and legacy issues rather than technical shortcomings. Our paper presents a new simple paradigm for network service differentiation that accounts explicitly for the multiplicity of Internet service providers and users as well as their economic interests in environments with partly deployed new services. Our key idea is to base the service differentiation on performance itself, rather than price. We design RD (Rate-Delay) network services that give a user an opportunity to choose between a higher transmission rate or low queuing delay at a congested network link. To support the two services, an RD router maintains two queues per output link and achieves the intended rate-delay differentiation through simple link scheduling and dynamic buffer sizing. We evaluate the performance, deployment, and security properties of the RD network services in various network topologies and traffic scenarios including delay-sensitive VoIP (Voice over Internet Protocol) applications.**

***Index Terms*—Service differentiation, rate, delay, trade-off, incremental deployment, incentive.**

I. Introduction

Numerous architectures with diversified network services have been proposed to remedy the inability of the Internet architecture to serve different applications in accordance with their diverse communication needs. IntServ (Integrated Services) [1], a prominent representative of the architectural innovations, offers users a rich choice of services that include guarantees on end-to-end throughput and delay within a packet flow. The IntServ design incorporates complicated admission control [2] and link scheduling mechanisms such as WFQ (Weighted Fair Queuing) [3] and WF^{2}Q (Worst-case Fair Weighted Fair Queuing) [4]. While IntServ failed to gain ubiquitous adoption, early IntServ retrospectives attributed the failure to the complexity of supporting the per-flow performance guarantees, especially in busy backbone routers. DiffServ (Differentiated Services) [5], a subsequently proposed architecture, addresses the scalability concerns by restricting complex operations to the Internet edges and offering just few services at the granularity of traffic classes, rather than individual flows. DiffServ did not deploy widely either in spite of its simpler technical design.

The deployment failures of the diversified-service architectures suggest that technical merits of an innovative solution is not the main factor in determining its success. Economic and legacy issues become a crucial consideration because the current Internet is a loose confederation of infrastructures owned by numerous commercial entities, governments, and private individuals [6]. The multiplicity of the independent stakeholders and their economic interests implies that partial deployment of a new service is an unavoidable and potentially long-term condition. Despite the partial deployment, the new service should be attractive for adoption by legacy users and ISPs (Internet Service Providers).

Our paper explores a simple novel paradigm for network service differentiation where deployment is viewed as the primary design concern. We believe that partial deployment is unavoidable and that the new design should be attractive for early adopters even if other ISPs or users refuse to espouse the innovation. Moreover, we require that the benefits of network service diversification should not come at the expense of legacy traffic. The imposed constraints are potent. In particular, they imply that the new architecture cannot assume that traffic shaping, metering, pricing, billing, or any other added functionality will be supported by most ISPs, even by most ISPs on Internet edges. To resolve the deployability challenge, we utilize built-in performance incentives as a basis for network service differentiation. While prior studies have established a fundamental trade-off between queuing delay and link utilization [7], the Internet practice has been favoring full utilization of bottleneck links at the price of high queuing delay. Although Internet link capacities are typically provisioned in excess of average rate demands in order to keep average delay low, even relatively infrequent instances of congestion result in spikes of high queuing delay that disrupt significantly the overall human perception of VoIP (Voice over Internet Protocol) conversations and other delay-sensitive applications.

Our proposal of RD (Rate-Delay) services resolves

M. Podlesny is with the Department of Computer Science, University of Calgary, Calgary, AB, T2N 1N4, Canada e-mail: (mpodlesn@ucalgary.ca).

S. Gorinsky is with Institute IMDEA Networks (Madrid Institute for Advanced Studies in Networks), Leganes, Madrid, 28918, Spain (email: sergey.gorinsky@imdea.org).

Manuscript received June 17, 2010; revised December 1, 2010.

the tension between the rate and delay by offering two classes of service: an R (Rate) service puts an emphasis on a high transmission rate, and a D (Delay) service supports low queuing delay. Each of the services is neither better nor worse per se but is simply different, and its relative utility for a user is determined by whether the user's application favors a high rate or low delay. Hence, the RD architecture provides the user with an incentive and complete freedom to select the service class that is most appropriate for the application. Packet marking in the sender realizes the selection of the R or D service. Thus, even if the link is not saturated all the time, our scheme supports the rate differentiation during the periods of its saturation and ensures that packets of delay-sensitive applications never face long queuing at network links. The interest of users in the low-delay D service is viewed as an indirect but powerful incentive for ISPs to adopt the RD services. By switching to the RD architecture, an ISP attracts additional customers and thereby increases revenue.

The RD design achieves its objectives primarily through packet forwarding in routers. The RD router serves each output link from two FIFO (First-In First-Out) queues and supports the intended rate-delay differentiation through dynamic buffer sizing and simple transmission scheduling. The RD router treats legacy traffic as belonging to the R class. The simplicity of the RD forwarding makes the design amenable to easy implementation even at high-capacity links.

The overall architecture remains in the best-effort paradigm and modifies forwarding but not routing. Neither R nor D service offers any rate, loss, or end-to-end delay guarantees. Moreover, although the RD services provide users and ISPs with incentives to adopt the services, the architecture does not eliminate most security problems of the Internet. Nevertheless, by assuring low queuing delay at individual links, our design expands dramatically the communication range with consistently low end-to-end delay. This success is due to the congestion patterns in the real Internet where a typical path suffers from congestion at only one or (more rarely) two bottleneck links [8], [9]. While queuing at non-bottleneck links is marginal, even a partial adoption of our simple RD architecture at congested links is able to reduce end-to-end delay of Internet paths significantly.

This paper integrates and extends our prior work on the RD network services [10], [11] and is organized as follows. Section II describes the conceptual framework of the proposed architecture. Section III clarifies the analytical foundations for RD router operation. Section IV elaborates the details of our design. Section V reports the extensive performance evaluation. Section VI considers the security aspects of the architecture. Section VII discusses related work. Section VIII suggests directions for future work. Finally, Section IX concludes the paper with a summary of its contributions.

II. Conceptual Design

The constraint of the partial deployment excludes the common approach of pricing and billing, e.g., because a user should be able to opt for the RD services despite accessing the Internet through a legacy ISP that provides no billing or any other support for service differentiation. With financial incentives not being an option, our key idea is to make the performance itself a cornerstone of the service differentiation. While the performance is subject to a fundamental trade-off between queuing delay and link utilization [7], different applications desire different resolutions to the tension between the two components of the performance. Hence, the RD services consist of two classes:

- R (Rate) service puts an emphasis on a high transmission rate, and
- D (Delay) service supports low queuing delay.

The utility of each service for a user depends on the user's application needs, i.e., whether an application requires a high rate or low delay. Since the network services are aligned with the application needs, each user receives an incentive to select the service of the most appropriate type, and the RD service architecture empowers the user to do such selection by marking the headers of transmitted packets.

To support the RD services on an output link, a router maintains two queues for packets destined to the link. We refer to the queues as an R queue and D queue. Depending on whether an incoming packet is marked for the R or D service, the router appends the packet to the R or D queue respectively. The packets within each queue are served in the FIFO (First-In First-Out) order. Whenever there is data queued for transmission, the router keeps the link busy, i.e., the RD services are work-conserving.

By deciding whether the next packet is transmitted from the R or D queue, the router realizes the intended rate differentiation between the R and D services. In particular, the link capacity is allocated to maintain a rate ratio of

$$k = \frac{r_R}{r_D} > 1 \tag{1}$$

where r_R and r_D are per-flow forwarding rates for packet flows from the R and D classes respectively, and a flow refers to a sequence of IP (Internet Protocol) [12] packets with the same source address, source port, destination address, and destination port.

The router supports the desired delay differentiation between the R and D services through buffer sizing for the R and D queues. As common in current Internet routers, the size of the R buffer is chosen large enough so that the oscillating transmission of TCP (Transmission Control Protocol) [13] and other legacy end-to-end congestion control protocols utilizes the available link rate fully. The D buffer is configured to a much smaller dynamic size to ensure that queuing delay for each forwarded packet of the D class is small and at

most d. The assurance of low maximum queuing delay is attractive for delay-sensitive applications and easily verifiable by outside parties. An interesting direction for future studies is an alternative design for the D service where queuing delay stays low on average but is allowed to spike occasionally in order to support a smaller loss rate. In agreement with our overall design philosophy, parameters k and d are independently determined by the ISP that owns the router. The ISP uses the parameters as explicit levers over the provided RD services. Our experiments suggest that the parameter settings of $k = 2$ and $d = 10$ ms generally support appropriate rate-delay differentiation incentives.

During early stages of adopting the RD architecture, legacy users will continue to dominate the customer bases of ISPs and remain the major sources of revenue. This economic insight dictates that adoption of the RD services by an ISP should not penalize traffic from legacy end hosts. While the R service and legacy Internet service are similar in putting the emphasis on a high transmission rate rather than low queuing delay, the legacy traffic and any other packets that do not explicitly identify themselves as belonging to the D class are treated by an RD router as belonging to the R class, i.e., the router diverts such traffic into the R queue. Since those flows that opt for the D service acquire the low queuing delay by releasing some fraction of the link capacity, the adopters of the D service also benefit the legacy flows by enabling them to communicate at higher rates.

Due to the potentially partial deployment of the RD services, R and D flows might be bottlenecked at a link belonging to a legacy ISP. Furthermore, the R and D flows might share the bottleneck link with legacy traffic. This has an important design implication that end-to-end transmission control protocols for the R and D services have to be compatible with TCP, e.g., the applications that communicate over UDP (User Datagram Protocol) [14] are expected to exercise TCP-compatible transmission control.

Our RD design offers users two services but can be straightforwardly extended to support additional levels of queuing delay. For example, a simple extension can offer three services with: (1) smallest rate with lowest delay, (2) middle rate with middle delay, or (3) largest rate with highest delay. As the number of additional rate-delay trade-off levels increases, their marginal utility decreases but the implementation complexity increases. By adding only one low-delay D service, the presented RD design overcomes the long queuing that hampers delay-sensitive applications in the current Internet.

III. Analytical Foundation

While Section II outlined the conceptual design of the RD services, we now present an analytical foundation for our specific implementation of RD routers.

A. Notation and assumptions

Consider an output link of an RD router. Let C denote the link capacity and n be the number of flows traversing the link:

$$n = n_R + n_D \tag{2}$$

where n_R and n_D represent the number of R (including legacy) and D flows respectively. For analytical purposes, we assume that both R and D queues are continuously backlogged and hence

$$C = R_R + R_D \tag{3}$$

where R_R and R_D refer to the service rates for the R and D queues respectively. Also, we assume that every flow within each class transmits at its respective fair rate, r_R or r_D:

$$R_R = n_R r_R, \quad R_D = n_D r_D. \tag{4}$$

Our experiments with dynamic realistic traffic including a lot of short-lived flows confirm that the above assumptions do not undermine the intended effectiveness of the RD services in practice.

We denote the sizes of the R and D queues as q_R and q_D respectively and the buffer allocations for the queues as B_R and B_D respectively. If the corresponding buffer does not have enough free space for an arriving packet, the router discards the packet.

B. Sizing and serving the R and D queues

Combining Equations 1, 3, and 4, we determine that the service rates for the R and D queues should be respectively equal to

$$R_R = \frac{kn_R C}{n_D + kn_R}, \quad R_D = \frac{n_D C}{n_D + kn_R}. \tag{5}$$

The RD router sizes the D buffer dynamically so that that queuing delay for any packet placed in the D queue never exceeds d. We determine the necessary and sufficient size of the D buffer through the worst-case delay analysis [11]. The analysis identifies the packet arrival patterns that result in the largest queuing delay for D packets under the RD forwarding algorithm. Based on this analysis, the D buffer is sized as

$$B_D = \lfloor R_D(d - w) \rfloor^+ \tag{6}$$

where

$$w = \frac{2}{C}\left(S_D^{max}\frac{kn_R}{n_D} + S_R^{max}\right) \tag{7}$$

and S_D^{max} and S_R^{max} refer to the maximum size of D packets and R packets, respectively.

Taking Equation 5 into account, we express the D buffer size as:

$$B_D = \left\lfloor \frac{n_D C(d - w)}{n_D + kn_R} \right\rfloor^+. \tag{8}$$

In practice, we expect B_D to be much smaller than overall buffer size B that the router has for the link. Manufacturers equip current Internet routers with substantial

Notation	Semantics
x	class of the service, R or D
n_x	number of flows from class x
L_x	amount of data transmitted from queue x since the last reset of L_x
B_x	buffer allocation for queue x
q_x	size of queue x
p	packet

Fig. 1. Internal variables of the RD router algorithms in Figures 3, 4, and 5.

Parameter	Semantics
d	upper bound on queuing delay experienced by a packet of class D
k	ratio of per-flow rates for classes R and D
T	update period
E	flow expiration period
b	timestamp vector size

Fig. 2. Parameters of the RD router algorithms.

memory so that router operators could configure the link buffer to a smaller (but still large) value B_{max} chosen to support throughput-greedy TCP traffic effectively [15]. Thus, we recommend to allocate the buffer for the R queue to the smallest of $B - B_D$ and B_{max} (and expect B_{max} to be the common setting in practice):

$$B_R = \min\left\{B_{max};\ B - \left\lfloor \frac{n_D C(d-w)}{n_D + kn_R} \right\rfloor^+\right\}. \quad (9)$$

IV. Design Details

A. End hosts

As per our discussion at the end of Section II, the RD services do not demand any changes in end-to-end transport protocols. The only support required from end hosts is the ability to mark a transmitted packet as belonging to the D class. We implement this requirement by employing bits 3-6 in the ToS (Type of Service) field of the IP datagram header [12]. To choose the D service, the bits are set to 1001. The default value of 0000 corresponds to the R service. Thus, the RD services preserve the IP datagram format.

B. Routers

The main challenge for transforming the analytical insights of Section III into specific algorithms for RD router operation lies in the dynamic nature of Internet traffic. In particular, while Expressions 5, 8, and 9 depend on n_R and n_D, the numbers of R and D flows change over time. Hence, the RD router periodically updates its values of n_R and n_D. Sections IV-B1, IV-B2, and IV-B3 describe our algorithms for processing a packet arrival, serving the queues, and updating the algorithmic variables at the RD router respectively. Figure 1 summarizes the internal variables of the algorithms. In addition to the internal variables, a number of parameters characterize the RD router operation. Figure 2 sums up these parameters.

```
p ← the received packet;
x ← the class of p;
S ← size of p;
if q_x + S ≤ B_x
        append p to the tail of queue x;
        q_x ← q_x + S;
 else
        discard p
```

Fig. 3. RD router operation upon receiving a packet destined to the link.

```
\* select the queue to transmit from *\
if q_R > 0 and q_D > 0
      if kn_R L_D > n_D L_R
          x ← R;
      else
          x ← D;
 else \* exactly one of the R and D buffers is empty
*\
      x ← class of the non-empty buffer;
 p ← first packet in the x queue;
 S ← size of p;
 if p != null
      \* update the L variables *\
      if q_R > 0 and q_D > 0
          L_x ← L_x + S;
          δL ← (L_R n_D)/(k n_R) − L_D;
          if δL < 0 δL ← 0;
      else \* only D buffer is empty *\
      if q_R > 0 and q_D = 0
          L_R ← 0; L_D ← 0;
      else \* only R buffer is empty *\;
            if δL > 0 δL ← δL − S;
            if δL > 0 L_D ← −δL;
            else
                L_D ← 0;
            L_R ← 0;
      transmit p into the link;
      q_x ← q_x − S
```

Fig. 4. Router operation when the RD link is idle, and the link buffer is non-empty.

1) Processing a packet arrival: Figure 3 presents our simple algorithm for dealing with packet arrivals. When the router receives a packet destined to the link, the router examines bits 3-6 in the TOS field of the packet header to determine whether the packet belongs to the R or D class. If the corresponding buffer is already full, the router discards the packet. Otherwise, the router appends the packet to the tail of the corresponding queue.

$B_D^{old} \leftarrow B_D$;
update n_R and n_D as in [16];
update B_R and B_D as per Equations 9 and 8;
if $\delta L > 0$ $L_D \leftarrow -\delta L$;
  else $L_D \leftarrow 0$;
$L_R \leftarrow 0$;
if $q_D > B_D$ **or** $B_D^{old} < B_D$
  discard all packets from the D queue;
  $q_D \leftarrow 0$;
else while $q_R > B_R$
    $p \leftarrow$ last packet in the R queue;
    $S \leftarrow$ size of p;
    discard p;
    $q_R \leftarrow q_R - S$

Fig. 5. Reset of the RD algorithmic variables upon timeout.

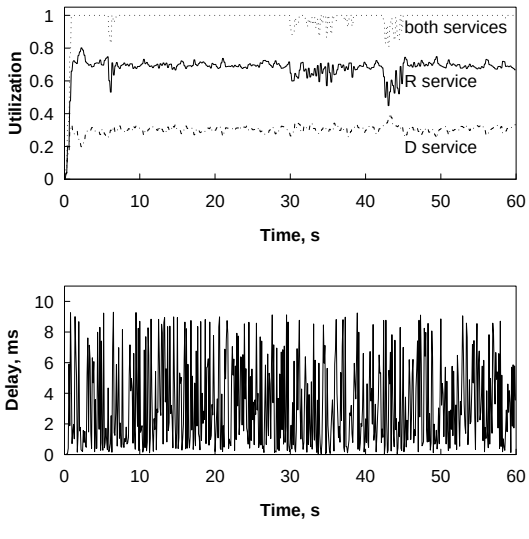


Fig. 6. Sample behavior of the RD services: (a) bottleneck link utilization; (b) queuing delay of D packets.

2) Serving the R and D queues: Figure 4 reports details of the algorithm for serving the R and D queues. While the RD services are work-conserving, the router transmits into the link whenever the link buffer is non-empty. Since the router can transmit at most one packet at a time, the intended split of link capacity C into service rates R_R and R_D can be only approximated. The router does so by:

- monitoring L_R and L_D, the amounts of data transmitted from the R and D queues respectively since the last reset of these variables;
- transmitting from such queue that $\frac{L_R}{L_D}$ approximates $\frac{R_R}{R_D} = \frac{kn_R}{n_D}$ most closely.

More specifically, when $kn_R L_D > n_D L_R$, the router transmits from the R queue; otherwise, the router selects the D queue.

3) Updating the algorithmic variables: There are two types of scenarios where the RD router resets the values of L_R and L_D to zero. First, $L_R \leftarrow 0$ and $L_D \leftarrow 0$ whenever at least one of the two buffers is empty, because the need for the differentiated forwarding arises only when both services are backlogged at the link. Second, if the backlog is mutual and continuous, the router periodically resets L_R and L_D to 0 to prevent their values from overflowing. The reset of L_R and L_D creates a possibility of violating the queuing delay constraint of the D service. To handle this issue, variable δL tracks the amount of traffic that needs to depart from the D queue in order to avoid the delay constraint violations.

Whereas n_R and n_D play important roles in the presented RD router algorithms, we compare two approaches to computing the numbers of flows: explicit notification from end hosts and independent inference by the router. Since our design handles a possibility that many hosts do not embrace the RD services, it is likely that the router serves many legacy flows and needs to do at least some implicit inference. Furthermore, since we favor solutions with minimal modification of the current infrastructure, the router in our RD implementation estimates n_R and n_D without any help from end hosts.

To estimate the numbers of flows, we independently apply the timestamp-vector algorithm [16] to each of classes R and D. Our experiments confirm the excellent performance of the algorithm. Using a hash function, the algorithm maps each received packet into an element of the array called a timestamp vector. The timestamp vector accommodates b elements. The algorithm inspects the timestamp vector with period T and considers a flow inactive if the timestamp vector did not register any packets of the flow during the last period E. Following the guidelines in [17] and assuming $E = 1$ s, 10^5 active flows, and standard deviation $\epsilon = 0.05$, we recommend $b = 18,000$ as the default setting for the timestamp vector size. The memory needs of the adopted timestamp-vector algorithm scale well, e.g., a link with 10^6 flows is effectively supported by the timestamp vector that is sized to 128,000 elements and occupies about 1 MB.

The RD router updates n_R and n_D with period T. At the same time, the router updates the buffer allocations for the R and D queues. Even if n_R or n_D is zero, the router allocates a non-zero buffer for each of the queues. Our experimental results suggest that the specific allocation split is not too important; in the reported experiments, we initialize the buffer allocations to $B_D = \frac{4Cd}{4+k}$ and $B_R = \min\{B_{max};\ B - B_D\}$, which correspond to the 1:4 ratio between the numbers of flows from classes R and D. If both n_R and n_D are positive, the router updates the buffer allocations according to Equations 8 and 9.

The update of B_R and B_D can make one of them smaller than the corresponding queue size. Figure 5 describes how the router deals with this issue. If the updated B_R is less than q_R, the router discards packets from the tail of queue R until q_R becomes at most B_R. The discards ensure that the D service receives the

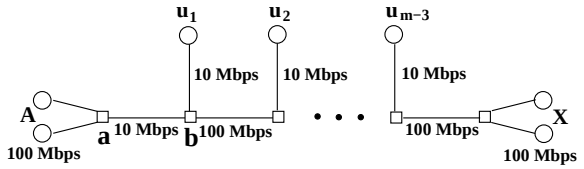


Fig. 7. Parking-lot topology for studying the impact of the hop count on end-to-end delay.

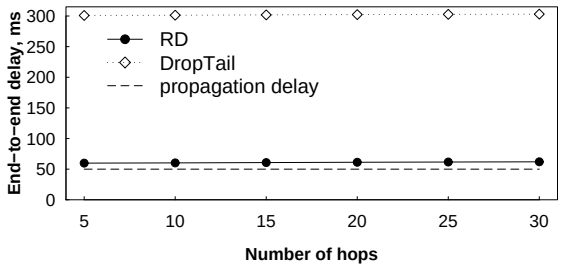


Fig. 8. Maximum end-to-end delay of D packets from pool A to pool X in the parking-lot topology.

intended buffer allocation. If the update decreases B_D, i.e., $B_D^{old} > B_D$, where B_D^{old} is the previous value of the size of the D buffer, the router flushes all packets from queue D to ensure that neither of them will be queued for longer than d. The longer queuing might occur otherwise because the decrease of B_D also proportionally reduces the service rate for queue D.

Although the D buffer is typically small, discarding the burst of packets might affect the loss rate negatively and be even unnecessary because it might be still possible to forward at least some of the discarded D packets in time despite the reduced service rate. We experimentally evaluate the packet discard policy in Section V.

To select update period T, we observe that reducing T increases the computational overhead. Also, the operation might become unstable unless T is much larger than d. However, with larger T, the design responds slower to changes in the network conditions. Our experiments show that $T = 400$ ms offers a reasonable trade-off between these factors.

V. Performance Evaluation

This section assesses performance of the RD services through simulations using version 2.29 of ns-2 [18]. Unless explicitly stated otherwise, all flows employ TCP NewReno [19] and data packets of size 1 KB. Each link buffer is configured to $B = B_{max} = C \cdot 250$ ms where C is the capacity of the link. Every experiment lasts 60 s and is repeated five times for each of the considered parameter settings. We report the average, minimum, and maximum values. The default settings include k = 2, d = 10 ms, b = 18,000, T = 400 ms, E = 1 s, T_{avg} = 200 ms, and T_q = 10 ms, where T_{avg} refers to the averaging interval for the bottleneck link utilization and loss rate, and T_q denotes the averaging interval for queuing delay. We also average the utilization and loss rate over the whole experiment with exclusion of its first 5 s.

A. Basic properties

1) Representative behavior: To illustrate how the RD services operate, this section experiments in a traditional dumbbell topology where the core bottleneck and access links have capacities 100 Mbps and 200 Mbps respectively. The bottleneck link carries 100 R flows and 100 D flows in both directions and has propagation delay 50 ms. We choose propagation delays for the access links so that propagation RTT (Round-Trip Time) for the flows is uniformly distributed between 104 ms and 300 ms. With $k = 2$ and equal numbers of R and D flows, we expect the R and D services to utilize the bottleneck link capacity fully with the 2:1 ratio. Figure 6 mostly confirms this expectation. Around 43 s into the experiment, congestion on the reverse portion of the TCP data path causes loss of acknowledgment packets and thereby lowers temporarily the throughput of the R traffic. For the R service, maximum queuing delay is about 375 ms, as expected for the link that allocates two thirds of its capacity C to the R flows and has the buffer sized to the product of C and 250 ms. Queuing delay for the D service fluctuates between 0 and d = 10 ms.

2) End-to-end delay: While the RD services strive to improve end-to-end Internet performance by providing low queuing delay on individual links, we now examine how end-to-end delay depends on the number of hops in the parking-lot topology presented in Figure 7. Pool A transmits 10 long-lived R flows and 10 long-lived D flows to pool X, which transmits the same mix of flows to pool A. The path between pools A and X contains m hops. All links on the path have capacity 100 Mbps except for link a-b which has capacity 10 Mbps. Each of the subsequent $m - 4$ links on the A-X path also serves 3-hop cross-traffic that consists of 5 long-lived R flows and 5 long-lived D flows, originates in node u_i, traverses one A-X path link, and terminates in node u_{i+1}, where $i = 1, \ldots, m - 4$. The links connecting nodes $u_1, \ldots, u_{m-3}$ to the A-X path have capacity 10 Mbps. Thus, link a-b is the bottleneck of the A-X path, and all the other $m - 1$ links on the path are utilized lightly in agreement with real Internet congestion patterns. Each link in the topology has propagation delay $\frac{50}{m}$ ms so that the A-X path has propagation delay 50 ms regardless of the number of links. This setting highlights the contribution of queuing into end-to-end delay of the path. We vary m from 5 to 30 and measure maximum end-to-end delay of D packets on the A-X path.

In the scenarios that Figure 8 denotes as DropTail, all routers use traditional forwarding, the TCP sources fill up the buffer of link a-b, and the corresponding 250-ms queuing delay raises the end-to-end delay above 300 ms. After link a-b adopts the RD forwarding with $d = 10$ ms, the maximum end-to-end delay of the D flows is at most 62 s even in the extreme stetting where the A-X path contains 30 links. For both DropTail and RD, Figure 8

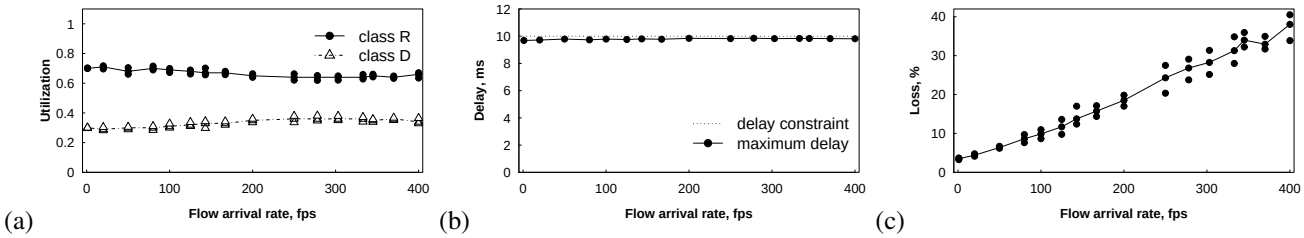


Fig. 9. Influence of the short-lived traffic on the RD services for different arrival rates of the web-like flows: (a) average utilizations for the R and D classes; (b) maximum queuing delay of D packets; (c) average loss rate for the D class.

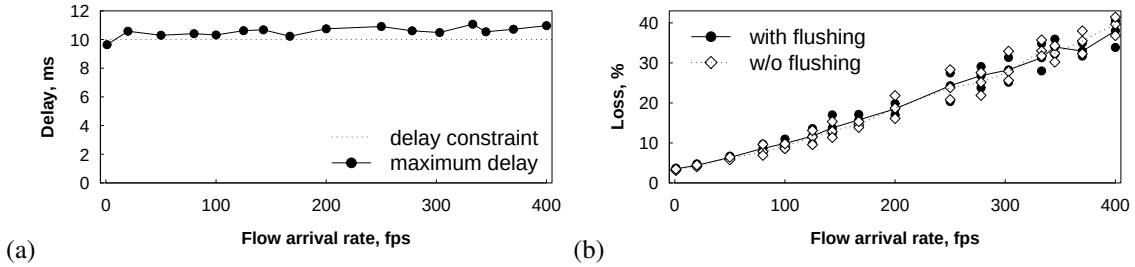


Fig. 10. Influence of the D-queue flushing on the RD services for different intensities of the web-like traffic: (a) maximum queuing delay of D packets; (b) average loss rate for the D class.

shows that all the non-bottleneck links contribute less than 3 ms of cumulative queuing delay regardless of how numerous these non-bottleneck links are. Hence, our results illustrate that the RD Services can dramatically improve end-to-end performance by being deployed only at bottleneck links.

3) Short-lived flows: We now revert to the dumbbell topology in Section V-A1 and enhance the traffic mix on the bottleneck link with web-like flows from two sources: one source generates R flows, and the other transmits D flows. The sizes of the web-like flows are Pareto-distributed with the average of 30 packets and shape index of 1.3. The flows arrive according to a Poisson process. The average arrival rate varies from 1 to 400 fps (flows per second). When the flows arrive more frequently, the traffic mix becomes burstier and imposes higher load on the bottleneck link. As expected, these factors drive up the loss rate for the D service. Figure 9 reveals that despite the increasing losses, the RD services closely maintain the intended 2:1 per-flow rate ratio for the R and D flows. The maximum queuing delay for the D packets does not exceed the delay constraint for any of the examined flow arrival rates.

4) Packet discard policy: This section evaluates how the RD services are affected by the D-buffer flushing that discards all queued D packets upon variable updates as described in Section IV-B3. Whereas the packet discard policy assures the maximum queuing delay, avoidance of the flushing might decrease the loss rate for the D service at the expense of occasional delay violations. For the same settings as in Section V-A3, Figure 10 reports maximum queuing delay and loss rate for the D flows with and without the D-queue flushing. As expected, maximum queuing delay of the D packets exceeds the upper bound consistently except for the web-flow arrival rate of 1 fps. However, the delay violations are insignificant, at most 1 ms. Also, while the loss rate for the D class remains approximately the same as with the flushing, the packet discard policy does not make a significant impact on the performance of the RD services.

B. Incremental deployment

Our RD design aspires to attract adopters despite continued presence of legacy ISPs and without penalizing legacy traffic. This section experimentally verifies whether the RD services fulfill these aspirations in the multiple-ISP topology depicted in Figure 11. The network core consists of ISP Z and ISP Y. Routers y1 and y2 of ISP Y offer the RD services with $k = 2$ and $d = 15$ ms. Backbone link z2-y1 connects the two ISPs and provides universal connectivity for all users. The users form five pools H, J, K, F, and G. Each user accesses his or her ISP through a personal link with capacity 100 Mbps. Link z1-z2 has capacity 55 Mbps making link y1-y2 a bottleneck for all the flows.

We choose propagation delays for the access links so that propagation RTT for the flows is uniformly distributed between 64 ms and 300 ms. In particular, propagation delay for both access links of each flow from pool H or J is chosen between 1 ms and 60 ms, and both access-link propagation delays for a flow from pool K or F are selected between 11 ms and 70 ms. The flows arrive according to a Poisson process. The average arrival rate is set by default to 100 fps for creating a confident expectation that all the flows arrive before the measurement stage of the experiment.

Every user from pools H, J, K, and F transmits a long-lived flow to a separate user in pool G. Hence, while

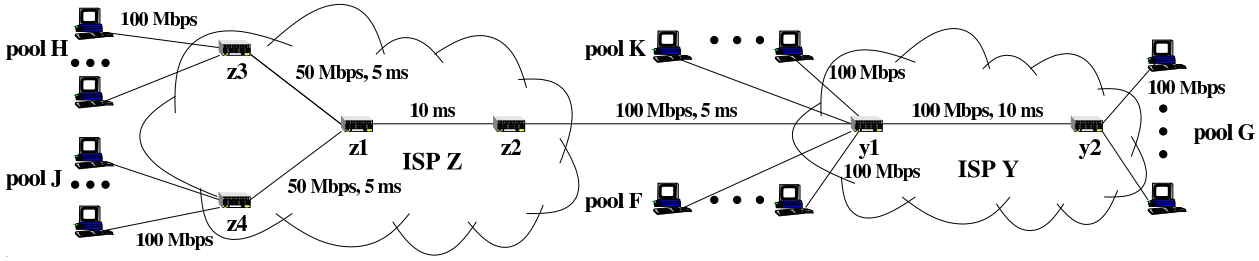


Fig. 11. Multiple-ISP topology for studying the incremental deployment of the RD services.

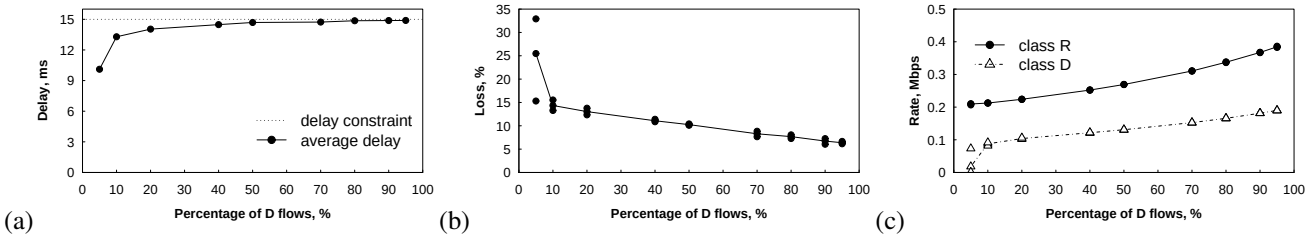


Fig. 12. Performance of the R (including legacy) and D flows on bottleneck link y1-y2 in the multiple-ISP topology during the incremental deployment of the RD services: (a) maximum queuing delay of D packets; (b) average loss rate for the D class; (c) average per-flow rates of the R and D classes.

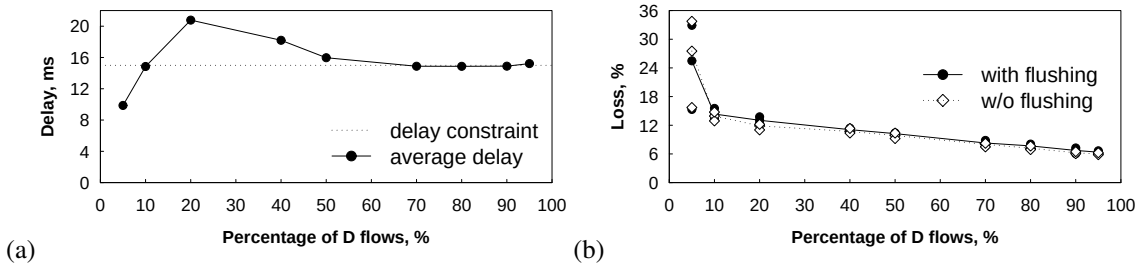


Fig. 13. Influence of the D-queue flushing on the RD services in the multiple-ISP incremental-deployment scenarios: (a) maximum queuing delay of D packets; (b) average loss rate for the D class.

the flows from pools K and F traverse the infrastructure that belongs only to ISP Y, both ISPs serve the flows from pools H and J. 500 flows traverse the network: 125 flows come from pool H, other 125 flows originate at pool J, and the remaining 250 flows enter from pools K or F. We vary fraction ρ of flows that choose the new D service. More specifically, $\lceil 125\rho \rceil$ D flows come from pool H, $\lceil 125\rho \rceil$ D flows originate at pool J, all $2 \cdot \lceil 125\rho \rceil$ flows from pool F adopt the D service, while the other $500 - 4 \cdot \lceil 125\rho \rceil$ flows are either legacy or R-class. We change ρ from 0.05 to 0.95, i.e., from 5% to 95% of the flows in the topology adopt the D service.

The multiple-ISP topology is interesting for the incremental-deployment studies because ISP Y supports the RD services but ISP Z provides only the legacy service, i.e., each network link of ISP Z performs traditional DropTail FIFO forwarding. Figure 12 plots the performance that the legacy and R flows and D flows experience at bottleneck link y1-y2. The results yield three pertinent insights.

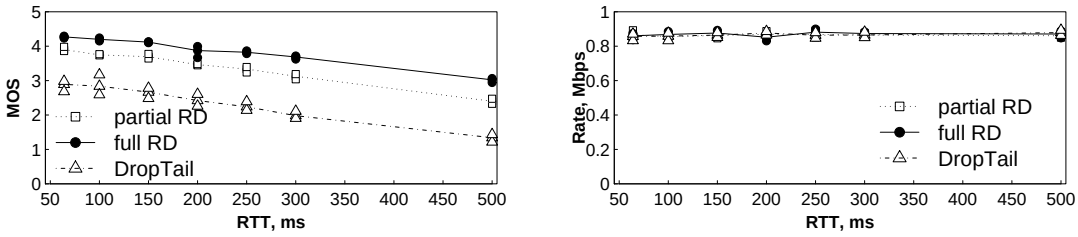
First, Figure 12a shows that the H-pool and J-pool users adopting the D service enjoy low queuing delay at the bottleneck link (and thereby low end-to-end delay) despite accessing the Internet through legacy ISP Z. Hence, even the users of legacy providers have strong incentives to adopt the RD services.

Second, Figure 12c demonstrates that the per-flow rate of the D flows increases consistently as more users adopt the D service. The result demonstrates the positive network externalities of the RD services. The more users adopt the D service, the more valuable the service becomes because of its increasing per-flow rate. Figure 12b corroborates the above assertion by plotting the loss rate for the D class: as the fraction of the D flows increases, the increasing size of the D buffer reduces the loss rate. When only 5% of the flows in the topology adopt the D service, the loss rate for the D class is high at 25% because the D buffer size accommodates less than 3 packets in this setting. ISP Y can reduce the loss rate by increasing the value of d for link y1-y2.

Third, Figure 12c also reveals that the per-flow rate of the R (including legacy) flows grows steadily as the D service gains wider adoption. Therefore, incremental deployment of the RD services produces a win-win outcome for both rate-sensitive and delay-sensitive types of users. In particular, instead of being penalized by the adopters of the D service, the legacy traffic benefits as well by becoming able to communicate at higher rates.

TABLE I
CATEGORIES OF VOICE TRANSMISSION QUALITY

R-factor range	MOS	Quality category	User satisfaction
90 - 100	4.34 - 4.50	Best	Very satisfied
80 - 90	4.03 - 4.34	High	Satisfied
70 - 80	3.60 - 4.03	Medium	Some users dissatisfied
60 - 70	3.10 - 3.60	Low	Many users dissatisfied
50 - 60	2.58 - 3.10	Poor	Nearly all users dissatisfied



(a) (b)

Fig. 14. Application-centric performance of VoIP in the three scenarios of the RD-services deployment: (a) average MOS; (b) average per-flow rate of the R class.

We now utilize the above incremental-deployment scenarios to revisit the impact of the D-queue flushing in the multiple-ISP topology. Figure 13 shows that delay-constraint violations without the flushing can be as high as 5 ms, i.e., more pronounced than in Figure 10. On the other hand, the loss rate for the D class remains about the same with and without the flushing. The flushing does not raise the loss rate significantly because emptying the D queue creates extra space for future D packets, and a future D packet is less likely to be dropped than without the flushing. Based on the quantified trade-offs between delay and loss in Figures 10 and 13, we recommend equipping the RD router design with the D-buffer flushing because this assures the maximum queuing delay without increasing the loss rate substantially.

C. Application-perceived end-to-end performance

Shifting the evaluation focus from the network-centric metrics of rate, delay, and loss to application-perceived performance, this section assesses the Internet-telephony application of VoIP with respect to the application-centric end-to-end metric of MOS (Mean Opinion Score) [20]. MOS maps voice quality into a range from 1 (unacceptable quality) to 5 (excellent quality). We utilize the E-Model [21] to estimate the subjective MOS scores based on our measurements of the network-centric metrics. The E-Model relies on an R-factor that captures all the impairments in the voice signal. The R-factor ranges from 0 (worst) to 100 (no impairments). Table I correlates the R-factor with MOS and voice quality.

We simulate VoIP conversations and measure their MOS in ns-2 by utilizing the VoIP tool in [22]. The speech is encoded with the AMR (Adaptive Multi-Rate) audio codec [23] operating at the rate of 12.2 Kbps. The tool transmits VoIP flows over UDP. All VoIP packets have the same size of 32 bytes. We conduct the simulations as in Section V-A3 but turn all the long-lived D flows into VoIP flows. The values of d and bottleneck link delay are set to 50 ms and 10 ms respectively. The web flows arrive with the intensity of 50 fps. We compute the average MOS based on the measurements starting from 10 s into the experiment.

Our first experiment lasts for 600 s. 500 VoIP D flows arrive from the very beginning at the average rate of 1 fps and stay until the end of the experiment. With the traditional DropTail FIFO forwarding, the average MOS is 2.97. With the RD services, the average MOS becomes 4.16. Thus, the RD services dramatically improve the VoIP quality in this dynamic scenario.

Finally, we experiment in the multiple-ISP topology with one bottleneck link in each of its two ISPs. Two pools of users transmit 50 R flows each over a single ISP and its bottleneck link. The other two pools transmit 50 VoIP D flows and 50 R flows that traverse both bottleneck links of the two ISPs. We consider three deployment scenarios: the traditional DropTail FIFO forwarding everywhere, adoption of the RD services by the first ISP only, and adoption of the RD services by both ISPs. Figure 14 refers to these scenarios respectively as DropTail, partial RD, and full RD. The results show that even the partial deployment of the RD services significantly improves the application-centric performance. Moreover, the full deployment of the RD design further improves the MOS scores of the VoIP conversations. Figure 14b illustrates that the improvements of the voice quality do not decrease the rates of the R flows.

VI. SECURITY CONSIDERATIONS

Whereas security of the RD architecture needs a separate future evaluation, this section experimentally examines few potential vulnerabilities of the RD design to sender misbehavior. We conduct the experiments in

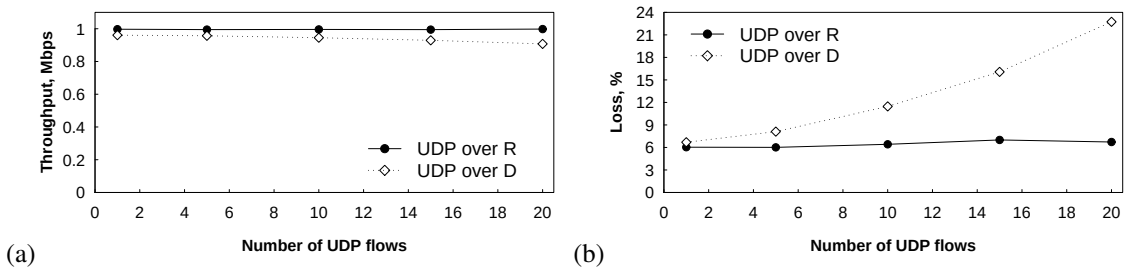


Fig. 15. Choosing the R versus D service for the throughput-greedy UDP transmission: (a) average per-flow throughput of the throughput-greedy UDP flows; (b) average loss rate for the D class.

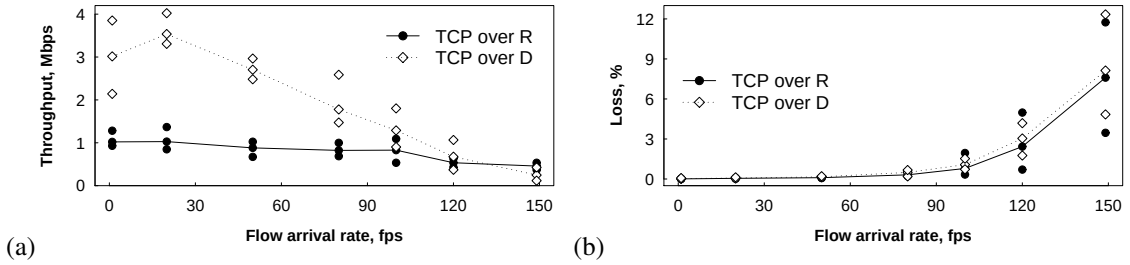


Fig. 16. Exploiting the low transmission rates of legitimate D flows by a throughput-greedy TCP flow: (a) average throughput of the throughput-greedy TCP flow; (b) average loss rate for the D class.

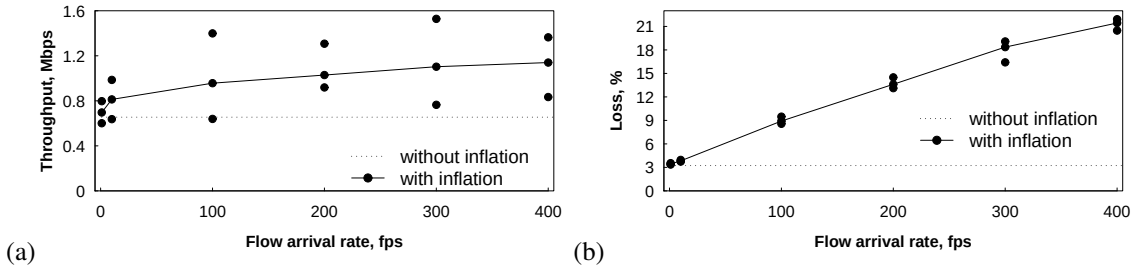


Fig. 17. Inflating the R flow count by a throughput-greedy TCP flow: (a) average throughput of the throughput-greedy TCP flow; (b) average loss rate for the D class.

the same network topology and for the same traffic pattern as in Section V-A3, except for the bottleneck link delay that we set to 10 ms.

First, we explore a scenario with throughput-greedy UDP senders where each of the UDP sources transmits at the constant rate of 1 Mbps. We vary the number of the UDP senders from 1 to 20. The intensity of the web cross traffic is 50 fps. Figure 15 shows that the loss rate for the D class grows up to 25% when the UDP sources select the D service. These UDP flows do not exercise congestion control but the D-class traffic also includes TCP-controlled web flows that reduce their transmission in response to the high losses. Despite pushing most of the damage to the TCP cross-traffic, the UDP flows consistently achieve lower throughput than in the alternative scenario where they choose the R service. Hence, in agreement with our incentive intentions, the RD design steers the throughput-greedy UDP flows to the R service, rather than to the low-delay D service where the negative impact of the excessive UDP transmission on the loss rate would be greater.

Second, we assess an attempt of a throughput-greedy TCP sender to exploit the potentially low transmission rates of delay-sensitive D flows. The throughput-greedy TCP source might increase its throughput by switching from the intended R service to the D service if the legitimate D flows underutilize their share of the bottleneck link capacity. We conduct simulations as in Section V-C by replacing all the long-lived D flows with 100 VoIP D flows. Figure 16 reveals that the throughput-greedy TCP sender is indeed able to benefit from the misbehavior and attain a significantly higher throughput by switching to the D service. The switch also raises the loss rate of the D service, although the increase is not substantial.

The success of the above attack is not certain and depends on the traffic pattern of the legitimate flows. Now, we consider an explicit attempt by a throughput-greedy R sender to manipulate the forwarding algorithm at the bottleneck link. More specifically, the throughput-greedy R sender inflates the count of R flows by generating dummy one-packet R flows. In its turn, the inflated flow count increases the bottleneck capacity share allocated to the R class, and this translates into personal throughput benefits for the misbehaving R sender. In our simulations

of this scenario, we have no web cross traffic and vary the intensity of the dummy-flow arrivals from 1 fps to 400 fps. Figure 17 confirms that the misbehaving R sender succeeds in improving its throughput substantially. Also, the flow count inflation increases the loss rate for the suppressed D service.

The presented experiments show vulnerabilities of the RD forwarding algorithm to attacks on its flow-counting implementation. The attacks enable a misbehaving sender to acquire both high throughput and low queuing delay at the bottleneck link. While the incentive mechanism of the RD services is imperfect, there exists space for future RD-like designs that assure as large throughput with an R-like service as with a D-like service and as low queuing delay with the D-like service as with the R-like service.

VII. Related Work

Network service differentiation through multiclass architectures has been a topic of extensive research, with the IntServ [1] and DiffServ [5] initiatives being prominent examples. The main feature that favorably distinguishes the RD services from the prior work is their incremental virulent deployability despite continued presence of legacy traffic and legacy service providers.

IntServ offers users an exciting possibility to receive absolute end-to-end rate and delay guarantees for individual flows. To provide the flexible but assured differentiation at the flow granularity, the best IntServ designs employ different complicated link scheduling algorithms [3], [4], and restrict network access with distributed admission control [24]. In contrast, RD routers maintain only two FIFO queues per output link and schedule the link capacity with the simple algorithm which is easy to implement even at high bitrates. Besides, the RD services exercise no admission control because the latter is ineffective under partial deployment where legacy ISPs keep providing users with unfettered access to shared bottleneck links of the network. While early retrospectives attributed IntServ deployment failures to the overhead imposed on backbone routers by per-flow storage and processing, core-stateless versions of IntServ designs moved all per-flow state and operations to the network edges and scheduled the core link capacities with simpler algorithms [25], [26]. The core-stateless IntServ designs put even more faith in access ISPs and also fail to realize the promise of guaranteed services under partial deployment.

DiffServ distinguishes services not at the flow granularity but at a much coarser granularity of traffic classes [27]. Various DiffServ designs support either absolute guarantees or relative differentiation between the few traffic classes [28], [29]. The DiffServ schemes that offer absolute performance guarantees require admission control. The DiffServ schemes that support relative performance differentiation preserve the Internet openness but serve one traffic class better than another. Such differentiation requires charging lower prices for worse services because all users would otherwise opt for the best service. Since either admission control or differentiated pricing is ineffective in the presence of legacy ISPs, incremental deployability of all the DiffServ schemes is poor as well. In comparison, the incentives for adopting the RD services are tied only to the performance itself, not the price.

Among other proposals for service differentiation, Alternative Best Effort (ABE) [30] resembles the RD services by aspiring to diversify services without distinguishing their prices. In addition to a D-like low-delay green service, ABE offers a blue service with a smaller loss rate. The storage and processing overhead of ABE is substantially larger than for our RD design. Also, while ABE considers it normal for a flow to mark some packets blue and other packets green, potential negative impact of such practices on legacy traffic raises a concern that the ABE design does not incorporate a sound strategy for incremental deployment. Most importantly, the blue service does not consistently provide a larger rate, e.g., by transmitting more aggressively, the green users can enjoy both a higher rate and lower queuing delay than those of the blue users. The lack of explicit rate-delay differentiation significantly weakens incentives for adopting ABE. Best Effort Differentiated Services (BEDS) [31] are similar to ABE and suffer from similar limitations.

While our paper leverages the rate-delay trade-off to offer a simple architectural solution for network service differentiation, the relationship between the rate and delay has a number of other important dimensions explored in earlier work. In particular, source and network coding techniques enhance the ability of networks to support applications with high communication rates but at a price of extra delay [32], [33], [34]. The research results in this area include both specific delay-sensitive coding techniques [35] and fundamental information-theoretic limits on the rate-delay relationship [36], [37]. The rate-delay trade-off is also studied from queuing-theoretic perspectives, e.g., to understand the rate of congestion notification necessary for effective congestion control [38] and to analyze preemption rates in preemption-based multiclass architectures [39].

VIII. Future Work

We believe that the approach of designing for deployability holds great promise for not only network service differentiation but also other types of networking problems. Even within the conceptual framework of rate-delay differentiation, we see numerous opportunities for further fruitful exploration. For example, whereas our strict enforcement of the delay constraint for the D service is a conscious attempt to encourage the service adoption only if the user is really interested in assuredly low queuing delay, it is worth to investigate whether delay should be allowed to spike occasionally as long as average low delay remains guaranteed.

Despite the above envisioned improvements of the RD design, a flow that opts for the D service will likely experience a larger loss rate. The significance of the heavier losses for applications is an interesting topic for future study. If the impact is tangible, we anticipate subsequent design efforts on transport protocols tailored for the D service.

A related issue is whether the RD architecture induces any unintended behavior of users who seek to improve own service or deliberately disrupt services for other users. Although the two-queue design alleviates some denial-of-service attacks, the RD architecture inherits most security problems of the Internet. Furthermore, our own limited experimental evidence indicates that the incentive mechanism of the RD services is imperfect. Thus, securing the RD design is clearly an important area for future investigation.

IX. Conclusions

We presented the RD network services, an architecture for rate-delay differentiation in a confederation of network domains owned and operated by multiple providers. Putting an emphasis on incentives for both end users and ISPs to adopt the new low-delay service despite its partial deployment, we designed and implemented the RD services that offer two best-effort services of low queuing delay or higher throughput. The RD router supports the services with two queues per output link, one queue per traffic class. The extensive evaluation revealed that the design supports the intended rate-delay differentiation in a wide variety of settings. Using VoIP as an example, we demonstrated advantages of our architecture for real applications. Besides, the design improves performance of web browsing [11] and appears a promising architectural solution for other delay-sensitive or throughput-greedy applications. Other contributions of the RD services include:

- incremental deployability within the current Internet;
- preservation of the current end-to-end transport protocols and IP datagram header structure;
- elimination of the billing and management problems of previous DiffServ designs.

Acknowledgments

This work was supported in part by iCORE (Informatics Circle of Research Excellence) in Alberta, Canada, Department-of-Education grant S2009/TIC-1468 from the Regional Government of Madrid, Ramon-y-Cajal grant RYC-2009-04660 from the Spanish Ministry of Science and Innovation, FP7-PEOPLE grant 229599 and FP7-ICT grant 258053 from the European Commission.

References

[1] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: an Overview," IETF RFC 1633, June 1994.

[2] S. Gorinsky, S. Baruah, T. Marlowe, and A. Stoyenko, "Exact and Efficient Analysis of Schedulability in Fixed-Packet Networks: A Generic Approach," in *Proceedings IEEE INFOCOM 1997*, April 1997.

[3] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queuing Algorithm," in *Proceedings ACM SIGCOMM 1989*, September 1989.

[4] J. Bennett and H. Zhang, "WF2Q: Worse-case Fair Weighted Fair Queuing," in *Proceedings IEEE INFOCOM 1996*, March 1996.

[5] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services," IETF RFC 2475, December 1998.

[6] D. Clark, J. Wroclawski, K. Sollins, and R. Braden, "Tussle in Cyberspace: Defining Tomorrow's Internet," in *Proceedings ACM SIGCOMM 2002*, August 2002.

[7] K. K. Ramakrishnan and R. Jain, "A Binary Feedback Scheme for Congestion Avoidance in Computer Networks with a Connectionless Network Layer," in *Proceedings ACM SIGCOMM 1988*, August 1988.

[8] L. Deng and A. Kuzmanovic, "Monitoring Persistently Congested Internet Links," in *Proceedings IEEE ICNP 2008*, October 2008.

[9] D. Ghita, H. Nguyen, M. Kurant, K. Argyraki, and P. Thiran, "Netscope: Practical Network Loss Tomography," in *Proceedings IEEE INFOCOM 2010*, March 2010.

[10] M. Podlesny and S. Gorinsky, "RD Network Services: Differentiation through Performance Incentives," in *Proceedings ACM SIGCOMM 2008*, August 2008.

[11] ——, "Stateless RD Network Services," in *Proceedings IFIP Networking 2010*, May 2010.

[12] J. Postel, "Internet Protocol. DARPA Internet Program. Protocol Specification." IETF RFC 791, September 1981.

[13] V. Jacobson, "Congestion Avoidance and Control," in *Proceedings ACM SIGCOMM 1988*, August 1988.

[14] J. Postel, "User Datagram Protocol," RFC 768, August 1980.

[15] C. Villamizar and C. Song, "High performance TCP in ANSNET," *ACM SIGCOMM Computer Communication Review*, vol. 24, no. 5, pp. 45–60, October 1994.

[16] H.-A. Kim and D. O'Hallaron, "Counting Network Flows in Real Time," in *Proceedings IEEE GLOBECOM 2003*, December 2003.

[17] C. Estan, G. Varghese, and M. Fisk, "Bitmap Algorithms for Counting Active Flows on High Speed Links," *IEEE/ACM Transactions on Networking*, vol. 14, no. 5, pp. 925–937, October 2006.

[18] S. McCanne and S. Floyd, *ns Network Simulator. http://www.isi.edu/nsnam/ns/.*

[19] S. Floyd and T. Henderson, "The NewReno Modification to TCP's Fast Recovery Algorithm," RFC 2582, April 1999.

[20] ITU-T, "Methods for Subjective Determination of Transmission Quality," Recommendation P.800, August 1996.

[21] J. Bergstra and C. Middelburg, "The E-model, a Computational Model for Use in Transmission Planning," ITU-T Recommendation G.107, June 2006.

[22] A. Bacioccola, C. Cicconetti, and G. Stea, "User-level Performance Evaluation of VoIP Using ns-2," in *Proceedings NSTools 2007*, October 2007.

[23] J. Sjoberg, M. Westerlund, A. Lakaniemi, and Q. Xie, "Real-Time Transport Protocol (RTP) Payload Format and File Storage Format for the Adaptive Multi-Rate (AMR) and Adaptive Multi-Rate Wideband (AMR-WB) Audio Codecs," IETF RFC 3267, June 2002.

[24] J. Liebeherr, D. Wrege, and D. Ferrari, "Exact Admission Control for Networks with a Bounded Delay Service," *IEEE/ACM Transactions on Networking*, vol. 4, no. 6, pp. 885–901, December 1996.

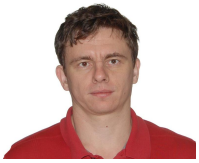
[25] I. Stoica, S. Shenker, and H. Zhang, "Core-Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks," in *Proceedings ACM SIGCOMM 1998*, September 1998.

[26] I. Stoica and H. Zhang, "Providing Guaranteed Services Without Per Flow Management," in *Proceedings ACM SIGCOMM 1999*, September 1999.

[27] S. Floyd and V. Jacobson, "Link Sharing and Resource Management Models for Packet Networks," *ACM Transactions on Database Systems*, vol. 3, no. 4, pp. 365–386, August 1995.

[28] V. Jacobson, K. Nichols, and K. Poduri, "An Expedited Forwarding PHB," IETF RFC 2598, June 1999.

[29] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured Forwarding PHB Group," IETF RFC 2597, June 1999.
[30] P. Hurley, J.-Y. L. Boudec, P. Thiran, and M. Kara, "ABE: Providing a Low-Delay Service within Best Effort," *IEEE Network*, vol. 15, no. 3, pp. 60–69, May/June 2001.
[31] V. Firoiu, X. Zhang, and Y. Guo, "Best Effort Differentiated Services: Tradeoff Service Differentiation for Elastic Applications," in *Proceedings IEEE ICT 2001*, June 2001.
[32] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, "Network Information Flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, July 2000.
[33] R. Koetter and M. Medard, "An Algebraic Approach to Network Coding," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782–795, October 2003.
[34] D. Lun, M. Medard, and M. Effros, "On Coding for Reliable Communication over Packet Networks," in *Proceedings 42nd Annual Allerton Conference on Communication, Control, and Computing*, September 2004.
[35] J. Walsh and S. Weber, "A Concatenated Network Coding Scheme for Multimedia Transmission," in *Proceedings Fourth Workshop on Network Coding, Theory, and Applications (NetCod 2008)*, January 2008.
[36] A. Eryilmaz, A. Ozdaglar, M. Medard, and E. Ahmed, "On the Delay and Throughput Gains of Coding in Unreliable Networks," *IEEE Transactions on Information Theory*, vol. 54, no. 12, December 2008.
[37] J. Walsh, S. Weber, and C. wa Maina, "Optimal Rate Delay Tradeoffs and Delay Mitigating Codes for Multipath Routed and Network Coded Networks," *IEEE Transactions on Information Theory*, vol. 55, no. 12, pp. 5491–5510, July 2009.
[38] K. Jagannathan, E. Modiano, and L. Zheng, "On the Trade-Off Between Control Rate and Congestion in Single Server Systems," in *Proceedings IEEE INFOCOM 2009*, April 2009.
[39] Z. Zhao, S. Weber, and J. de Oliveira, "Preemption Rates for a Parallel Link Loss Network," *Performance Evaluation*, vol. 66, no. 1, pp. 21–46, January 2009.



Maxim Podlesny received degrees of Bachelor of Science and Master of Science in Applied Physics and Mathematics from Moscow Institute of Physics and Technology, Russia in 2000 and 2002 respectively and the degree of Ph.D. in Computer Science from Washington University in St. Louis, USA in 2009. From 2009 Dr. Podlesny works as a Postdoctoral Fellow at the University of Calgary, Canada. His work has appeared in top conferences and journals such as ACM SIGCOMM, IEEE INFOCOM, and IEEE Journal on Selected Areas in Communications. His research interests are network congestion control, service differentiation, Quality-of-Service, and transport protocols.

Sergey Gorinsky received the Engineer degree from Moscow Institute of Electronic Technology, Zelenograd, Russia in 1994 and M.S. and Ph.D. degrees from the University of Texas at Austin, USA in 1999 and 2003 respectively. From 2003 to 2009, he served on the tenure-track faculty at Washington University in St. Louis, USA. Dr. Gorinsky currently works as a Senior Researcher at Institute IMDEA Networks, Madrid, Spain. The areas of his primary research interests are computer networking and distributed systems. His research contributions include multicast congestion control resilient to receiver misbehavior, analysis of binary adjustment algorithms, efficient fair transfer of bulk data, network service differentiation based on performance incentives, and economic perspectives on Internet routing. His work appeared at top conferences and journals such as ACM SIGCOMM, IEEE INFOCOM, IEEE/ACM Transactions on Networking, and IEEE Journal on Selected Areas in Communications. Sergey Gorinsky has served on the TPCs of INFOCOM (2006-2011), ICNP (2008, 2010), and other networking conferences. He co-chaired the TPCs of HSN 2008 (High-Speed Networks 2008, an INFOCOM 2008 workshop) and FIAP 2008 (Future Internet Architectures and Protocols 2008, an ICCCN 2008 symposium) and served as a TPC Vice-Chair of ICCCN 2009.