RD Network Services: Differentiation through Performance Incentives*

Maxim Podlesny
podlesny@arl.wustl.edu

Sergey Gorinsky
gorinsky@arl.wustl.edu

Applied Research Laboratory
Department of Computer Science and Engineering
Washington University in St. Louis
One Brookings Drive, Campus Box 1045
St. Louis, Missouri 63130-4899, USA

ABSTRACT

With the Internet offering a single best-effort service, there have been numerous proposals of diversified network services that align better with the divergent needs of different distributed applications. The failure of these innovative architectures to gain wide deployment is primarily due to economic and legacy issues, rather than technical shortcomings. We propose a new paradigm for network service differentiation where design principles account 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. The proposed RD (Rate-Delay) services enable a user to choose between a higher transmission rate or low queuing delay at a congested network link. An RD router supports the two services by maintaining two queues per output link and achieves the intended rate-delay differentiation through simple link scheduling and dynamic buffer sizing. After analytically deriving specific rules for RD router operation, we conduct extensive simulations that confirm effectiveness of the RD services geared for incremental deployment in the Internet.

Categories and Subject Descriptors

C.2.6 [**Computer-Communication Networks**]: Internetworking; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design

General Terms

Algorithms, Design, Economics, Performance

*This work was sponsored in part by U.S. National Science Foundation (NSF) grants CNS-0626661 and REC-0632580.

SIGCOMM'08, August 17–22, 2008, Seattle, Washington, USA.

Keywords

Service Differentiation, Transmission Rate, Queuing Delay, Incremental Deployment, Legacy Traffic, Legacy Infrastructure, Performance Incentive

1. INTRODUCTION

The mismatch between the single best-effort service of the current Internet and diverse communication needs of different distributed applications has led to numerous proposals of alternative architectures with diversified network services. A prominent representative of the architectural innovations, IntServ (Integrated Services) [4] offers users a rich choice of services, including end-to-end rate and delay guarantees provided to packet flows by means of admission control and link scheduling mechanisms such as WFQ (Weighted Fair Queuing) [9] or EDF (Earliest Deadline First) [10]. 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. The proposal of DiffServ (Differentiated Services) [3] 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. Despite the technically simpler design, DiffServ also failed to deploy widely.

The IntServ and DiffServ experiences reveal that technical merits of an innovative architecture are neither the only nor the most important factor in determining its success. Economic and legacy issues become a crucial consideration because the Internet of today is a loose confederation of infrastructures owned by numerous commercial entities, governments, and private individuals [7]. 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. Hence, a successful architecture should provide ISPs (Internet Service Providers) and users with incentives to adopt the new service despite the partial deployment.

In this paper, we investigate a novel paradigm for network service differentiation that makes deployability the primary design concern. We explicitly postulate 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. Besides, we demand that the benefits of the service diversification should not come at the expense of legacy traffic. The imposed constraints are potent. In particular, they imply that the new architecture may not assume even for the Internet edges that most ISPs will support admission control, traffic shaping, metering, billing, or any other mechanism added by the architecture.

The above design principles lead us to the key idea of making performance itself an incentive for network service differentiation. While prior studies have established a fundamental trade-off between link utilization and queuing delay [22, 33], the Internet practice favors full utilization of bottleneck links at the price of high queuing delay. Unfortunately, delay-sensitive applications suffer dearly from the long queues created by throughput-greedy applications at shared bottleneck links. Our proposal of RD (Rate-Delay) services resolves this tension 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. The user chooses the R or D service by marking a bit in the headers of transmitted packets.

We view the interest of users in the D service 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. We also envision an RD certification program championed by early adopters. The RD certification will serve as a catalyst for virulent deployment of the RD architecture because being RD-certified will give an ISP a differentiation advantage over legacy ISPs when competing with them for users and provider peering agreements.

To support the RD services on an output link, the router maintains two FIFO (First-In First-Out) queues and achieves the intended rate-delay differentiation through simple link scheduling and dynamic buffer sizing. The simplicity makes the RD design amenable to easy implementation even at high-capacity links. RD routers treat legacy traffic as belonging to the R class. After analytically deriving algorithms for RD router operation, we report extensive simulation results that confirm effectiveness of the RD services and their fitness for incremental deployment in the Internet.

Both services of the proposed RD architecture are still best-effort and do not promise any rate or loss guarantees. The proposal modifies forwarding but not routing. 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, and a malicious ISP can disrupt the rate and delay characteristics of transient RD traffic. While security is not the main focus of this study, we believe that the RD services do not introduce any fundamentally new vulnerabilities. For example, although a user can mark some packets as R-class and other packets as D-class to increase throughput, such behavior is essentially the same as the well-known Internet technique of running multiple flows in parallel. Moreover, the two-queue RD design alleviates some existing threats, e.g., if a D flow transmits excessively to create heavy losses for other flows at the shared bottleneck link, the RD router limits the damage from the denial-of-service attack to the D class and preserves the high transmission rates of concurrent legacy and R flows. Nevertheless, new behavioral patterns induced by the RD architecture and their security aspects clearly deserve thorough separate investigation. It is possible that design for incremental deployment is intrinsically less robust, and some security concerns in such architectures have to be addressed legally rather than through purely technical means.

The rest of the paper has the following structure. Section 2 clarifies our design principles. Section 3 outlines the conceptual framework of the RD services. Section 4 lays analytical foundations for RD router operation. Section 5 presents details of our design. Section 6 reports the extensive performance evaluation. Section 7 discusses related work. Section 8 suggests directions for future work. Finally, Section 9 concludes the paper with a summary of its contributions.

2. MODEL AND PRINCIPLES

We model the Internet as an interconnection of network domains owned and operated by numerous independent ISPs. ISPs generate revenue by selling network services to their direct customers. Users are the customers whose applications run at end hosts and send flows of packets over the Internet. In general, a network path that connects the end hosts of a distributed application traverses an infrastructure that belongs to multiple ISPs.

While different applications have different communication needs, the single best-effort service of the current Internet matches the interests of the users imperfectly. In response to this tension, various architectures with diversified network services have been proposed. Although technically brilliant, even the best of the proposals failed to gain wide deployment. We attribute the failures to ignoring the serious economic challenges of deploying a new service in a confederated infrastructure governed by numerous independent stakeholders. Instead of treating the deployment as an afterthought, we base our design on principles that explicitly acknowledge the multiplicity of Internet parties and their economic rationale in deciding whether to adopt new services.

First, we explicitly recognize that partial deployment is an unavoidable and potentially long-term condition for any newly adopted service. Hence, the new design should be attractive for early adopters even if other ISPs or users refuse to embrace the innovation:

PRINCIPLE 1. *A new service should incorporate incentives for both ISPs and end users to adopt the service despite the continued presence of legacy traffic or other ISPs that do not espouse the new service.*

The above principle has a more specific but nevertheless important implication that the new design should not worsen the service provided to legacy Internet users. Doing otherwise is against the economic interests of ISPs due to the danger of losing a large number of current customers who keep communicating via legacy technologies. This consideration leads us to the following principle:

PRINCIPLE 2. *Adoption of a new service should not penalize legacy traffic.*

3. CONCEPTUAL DESIGN

Below, we apply the principles from Section 2 to derive a conceptual design for Rate-Delay (RD) services, our solution to the problem of network service differentiation. As the name reflects, the RD services enable a user to choose between a higher transmission rate or low queuing delay at a congested network link.

Our Principle 1 prescribes providing both end users and ISPs with incentives for early adoption of the RD services. 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 direct 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 link utilization and queuing delay [22, 33], different applications desire different resolutions to the tension between the two components of the performance. Hence, the RD services consist of two classes:

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

Each of the two services is neither better nor worse per se but is merely different, and its relative utility for a user is determined by whether the user's application favors 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.

An ISP finds the RD services attractive due to the potential to boost revenue by adding customers who are interested in the D service. We envisage an RD certification program championed by a nucleus of early adopters. The RD certification will serve as a catalyst for virulent deployment of the RD architecture because being RD-certified will give an ISP a differentiation advantage over legacy ISPs when competing with them for users and provider peering agreements.

To support the RD services on an output link, the 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 refer to per-flow forwarding rates for packet flows from the R and D class respectively.

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) [25] 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 low 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 subsequent experimental study reveals suggested values for parameters k and d.

As per our Principle 2, 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. Our paper reports experiments with TCP NewReno [15], Paced TCP [1], and TFRC (TCP-Friendly Rate Control) [14] as end-to-end transport protocols for D flows. While losses at the smaller D buffer are expectedly higher, a separate investigation is needed to clarify how much the D service can benefit from new TCP-compatible transport protocols that address the higher losses by employing alternative mechanisms for congestion control or reliability [5, 18, 21, 29].

4. ANALYTICAL FOUNDATION

While Section 3 outlined the conceptual design of the RD services, we now present an analytical foundation for our specific implementation of RD routers.

4.1 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. We use n_R and n_D to represent the number of flows from the R and D class respectively. Since the router treats legacy traffic as belonging to the R class, we have

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

For analytical purposes, we assume that both R and D queues are continuously backlogged and hence

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

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

$$R_R = n_R r_R \text{ and } R_D = n_D r_D. \tag{4}$$

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

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

Parameter	Semantics
k	ratio of per-flow rates for R and D flows
d	upper limit on queuing delay of D packets
b	timestamp vector size
T	update period
E	flow expiration period

Figure 2: Parameters of the RD router algorithms.

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.

4.2 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

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

To ensure that queuing delay for any packet forwarded from the D queue does not exceed d, the buffer allocation for the queue should be bounded from above as follows:

$$B_D \leq R_D d. \tag{6}$$

Taking the second of Equations 5 into account, we establish the following buffer allocation for the D queue:

$$B_D = \frac{n_D C d}{n_D + kn_R}. \tag{7}$$

In practice, we expect B_D to be much smaller than overall buffer B that the router has for the link. Manufacturers equip current Internet routers with substantial memory so that router operators could configure the link buffer to a high value B_{max}, chosen to support throughput-greedy TCP traffic effectively [37]. 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):

$$B_R = \min\left\{B_{max};\ B - \frac{n_D C d}{n_D + kn_R}\right\}. \tag{8}$$

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

Figure 3: Router operation upon receiving a packet destined to the RD link.

5. DESIGN DETAILS

5.1 End hosts

As per our discussion at the end of Section 3, the RD services restrict end-to-end transmission control protocols to being compatible with TCP. The only extra 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 the currently unused bit 7 in the TOS (Type of Service) field of the IP (Internet Protocol) datagram header [32]. To choose the D service, the bit is set to 1. The default value of 0 corresponds to the R service. Thus, the RD services preserve the IP datagram format.

5.2 Routers

The main challenge for transforming the analytical insights of Section 4 into specific algorithms for RD router operation lies in the dynamic nature of Internet traffic. In particular, while Equations 5, 7, and 8 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 5.2.1, 5.2.2, and 5.2.3 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.

5.2.1 Handling 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 the seventh TOS bit in 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. Besides, if the enqueued packet belongs to the D class, the router remembers the arrival time of the packet until the packet reaches the head of the queue. Since the D buffer is typically small, storing the arrival times does not require significant memory.

5.2.2 Serving the R and D queues

The arrival times of enqueued D packets are used by the algorithm that serves the queues. The algorithm uses the times to ensure that queuing delay of forwarded D pack-

```
\* select the queue to transmit from *\
if q_R > 0 and q_D > 0
    if k n_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 x = D
    \* enforce the delay constraint of the D service *\
    while current time - t_p > d and q_D > 0
        discard p;
        q_D ← q_D − S;
        p ← first packet in the D 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;
    else \* one of the R and D buffers is empty *\
        L_R ← 0; L_D ← 0;
    transmit p into the link;
    q_x ← q_x − S
```

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

ets does not exceed upper bound d. More specifically, if the packet at the head of the D queue has been queued for longer, the router discards the packet. The situation might arise due to the dynamic nature of Internet traffic: since the population of flows changes, the service rate for the D queue might decrease after the packet arrives. Our initial version of this algorithm did not include the last-moment enforcement of the queuing delay constraint. Experimental results for the initial version were similar to those reported in Section 6: while the loss rates did not differ much, the maximum observed queuing delay exceeded d by about 0.5 ms. It remains to be seen whether the strict enforcement of the queuing delay constraint is worth the price of tracking the arrival times of the enqueued D packets.

Figure 4 reports further 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.

We derived the above algorithm from the assumption that all flows within a class transmit at the same fair rate, r_R

```
update n_R and n_D as per Section 5.2.3;
update B_R and B_D as per Section 5.2.3;
L_R ← 0; L_D ← 0;
if q_D > B_D
    discard all packets from the D queue;
    q_D ← 0;
else while q_R > B_R
        p ← last packet in the R queue;
        S ← size of p;
        discard p;
        q_R ← q_R − S
```

Figure 5: Update of the RD algorithmic variables upon timeout.

or r_D. While the assumption is clearly unrealistic, one specific problematic scenario occurs when the total transmission rate of the D flows is much less than $n_D r_D$, the maximum service rate for the D queue. Then, a throughput-greedy flow has an incentive to mark its packets as D packets and thereby achieve a much higher forwarding rate than the one offered by the intended R service. Although this scenario has not surfaced in our extensive simulations, and the unintended selection of the D service by the throughput-greedy flow does not disrupt the D service, this issue deserves close consideration. Our future study will explore in detail the implications of the diversity in flow rates and user behaviors (including deliberate denial-of-service attacks) for the RD services.

5.2.3 Updating the algorithmic variables

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 principles allow a possibility that many users 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 apply the timestamp-vector algorithm [27] separately to the R and D classes. 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 does not register any packets of the flow during last period E. Following the guidelines in [12] 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 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

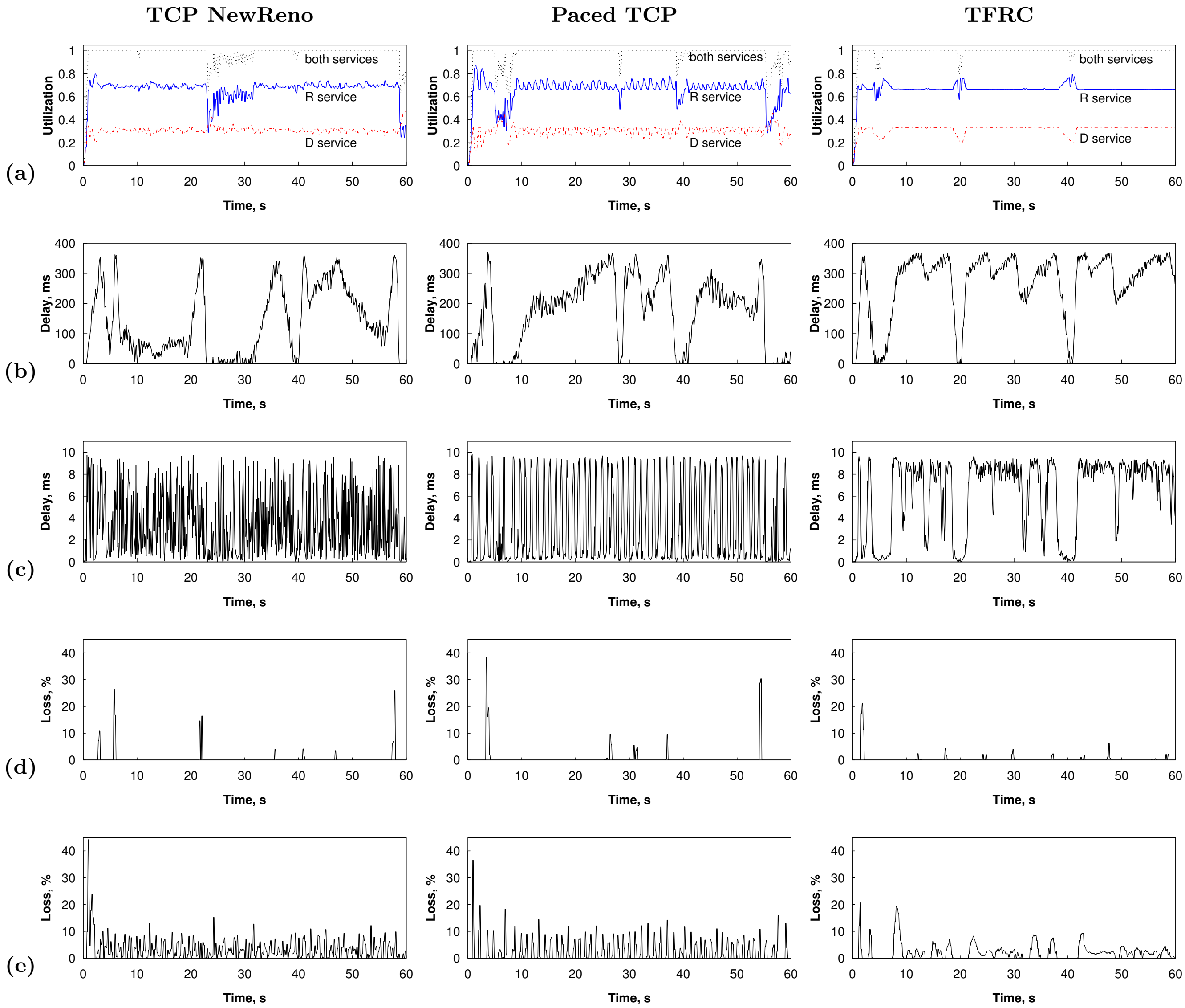


Figure 6: Using TCP NewReno, Paced TCP, or TFRC for D flows: (a) bottleneck link utilization; (b) queuing delay for R packets; (c) queuing delay for D packets; (d) loss rate for R flows; (e) loss rate for D flows.

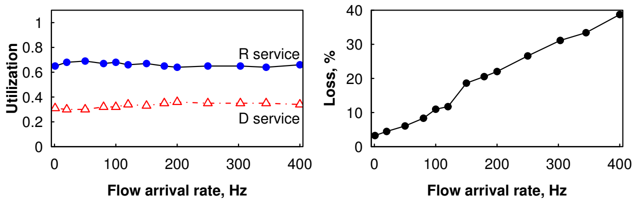
allocations to $B_D = \frac{4Cd}{4+k}$ and $B_R = \min\{B_{max};\ B - B_D\}$, which correspond to the 1:4 ratio between the numbers of flows from the R and D classes. If both n_R and n_D are positive, the router updates the buffer allocations according to Equations 7 and 8.

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 the R queue until q_R becomes at most B_R. The discards ensure that the D service receives the intended buffer allocation. If B_D is decreased below q_D, the router flushes all packets from the D queue. Emptying the D buffer assures that neither of the packets will be queued for longer than d and thus need to be discarded after reaching the head of the queue. The longer queueing might occur otherwise because the decrease of B_D also proportionally reduces the service rate for the D queue. 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. While our experiments show acceptably low loss rates with this implementation of the algorithm, we will explore more subtle discard policies in our future work.

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.

6. PERFORMANCE EVALUATION

In this section, we evaluate performance of the RD services through simulations using version 2.29 of ns-2 [30].



(a) Link utilization **(b) Loss rate for D flows**

Figure 7: Impact of the web-like traffic.

Unless explicitly stated otherwise, all flows employ TCP NewReno [15] 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. 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 five seconds. While queuing delay for the D service is at most d by design, all our experiments confirm that maximum delay of D packets satisfies and closely approximates this upper limit.

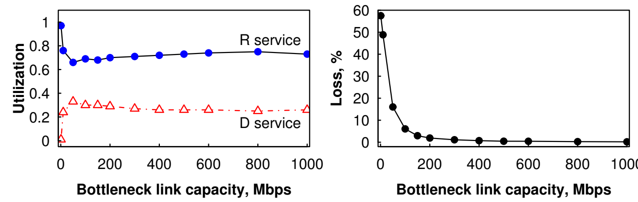
Section 6.1 evaluates the RD services in a wide variety of scenarios that include different transport protocols for D flows, both long-lived and short-lived traffic, diverse bottleneck link capacities, various settings for the delay constraint of the D service, Exponential and Pareto-distributed flow interarrival times, and sudden changes in the numbers of R and D flows. Section 6.2 continues the assessment in multi-ISP topologies and, in particular, examines whether the RD services are deployable despite the continued presence of legacy ISPs and without penalizing legacy traffic.

6.1 Basic properties

To understand basic properties of the RD services, 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.

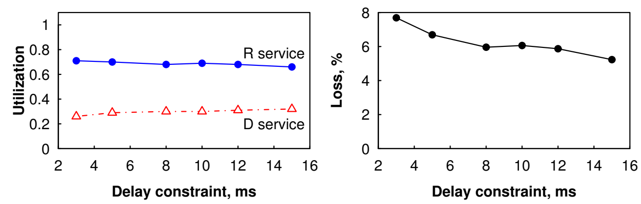
6.1.1 Various transport protocols for D flows

While the RD services restrict end-to-end transmission control to being compatible with TCP, we illustrate how the RD design performs when the D flows employ TCP NewReno [15], Paced TCP [1, 8], or TFRC [14]. All flows stay throughout the experiment. 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 and also plots queuing delay and loss rates for both services. 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. Due to slower detection of congestion and higher loss synchronization, Paced TCP yields larger losses than TCP NewReno. Among the three evaluated protocols, TFRC supports the smallest loss rate and most balanced rate differentiation. These superior properties make TFRC an attractive option for transmission control of D flows.



(a) Link utilization **(b) Loss rate for D flows**

Figure 8: Scalability of the RD services with respect to the bottleneck link capacity.



(a) Link utilization **(b) Loss rate for D flows**

Figure 9: Sensitivity to the delay constraint of the D service.

6.1.2 Short-lived flows and their intensity

To see how short-lived flows affect the RD services, we enhance the traffic mix on the bottleneck link in this and subsequent three experimental series 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. In the experiments of this section, the average arrival rate varies from 1 Hz to 400 Hz. 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 7 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.

6.1.3 Link capacity scalability

In this series of experiments, we vary the bottleneck link capacity from 1 Mbps to 1 Gbps while keeping the access link capacities twice as large. The average arrival rate for the web-like flows in this and next sections stays at 50 Hz. Figure 8 shows that the rates of the R and D flows deviate from the intended 2:1 ratio significantly only for the lowest examined capacities close to 1 Mbps. The deviation occurs due to the extremely small buffering available for D packets in those settings. In particular, satisfying the 10-ms delay constraint at the 1-Mbps bottleneck link reduces the D buffer to about one packet, and the minimal buffering causes heavy losses and effectively shuts down the D service. As the bottleneck link capacity grows, the loss rate for the D flows decreases exponentially.

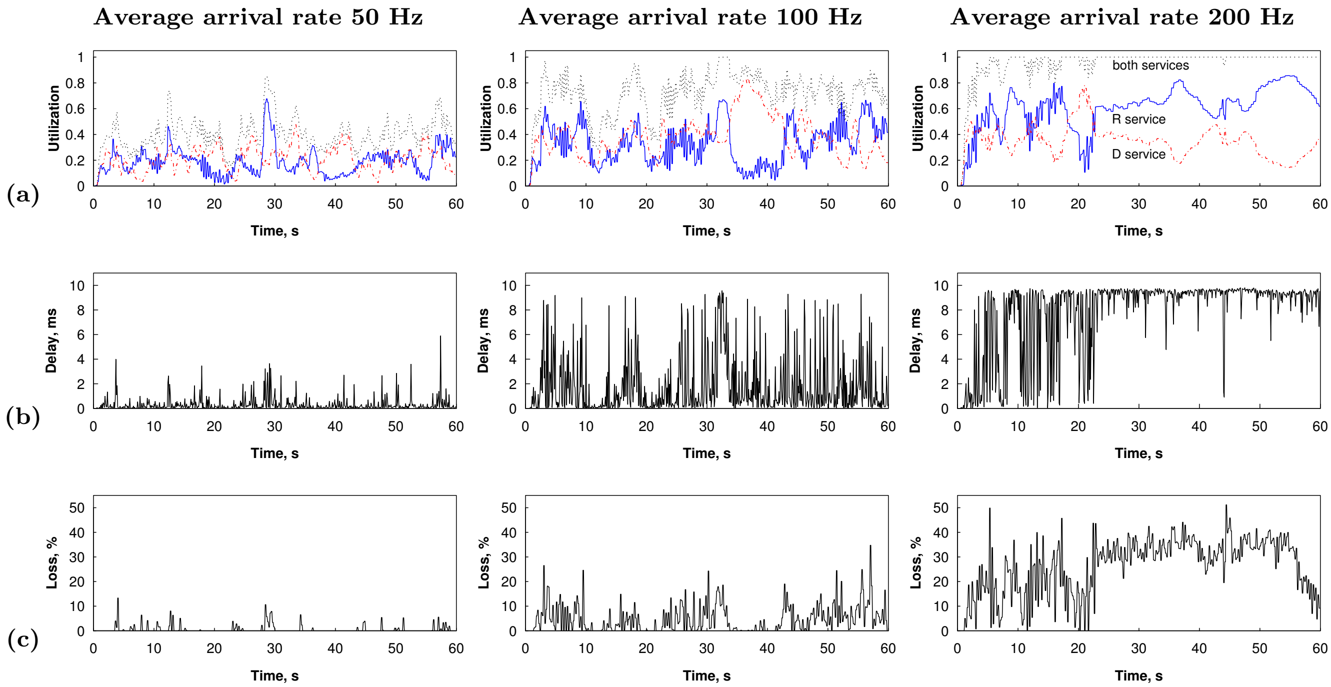
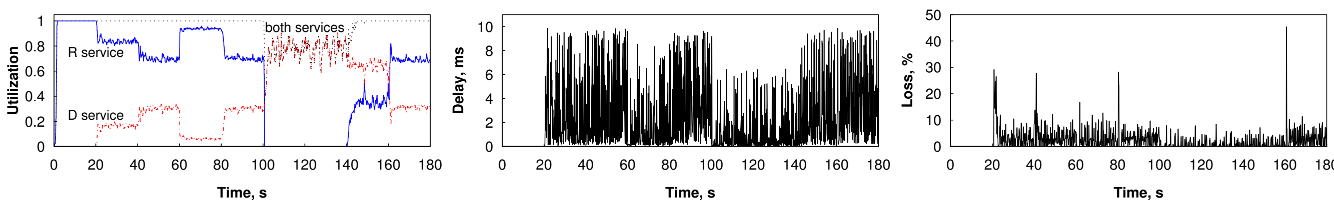


Figure 10: Performance of the RD services with the Pareto distribution for the interarrival times of the web-like flows: (a) link utilization, the legend of the rightmost graph applies to all three utilization graphs; (b) queuing delay for D packets; (c) loss rate for D flows.



(a) Link utilization **(b) Queuing delay for D packets** **(c) Loss rate for D flows**

Figure 11: Reaction to sudden changes in the numbers of R and D flows.

6.1.4 Sensitivity to the delay constraint

To examine sensitivity of the RD services to d, we vary the delay constraint of the D service from 3 ms to 15 ms. Figure 9 demonstrates that the per-flow rate ratio for the R and D flows stays close to the intended 2:1. As d increases, the loss rate for the D service decreases from about 8% to about 5% due to the increasing size of the D buffer.

6.1.5 Heavy-tailed flow interarrival times

While Section 6.1.2 experiments with the web-like traffic where the flow interarrival times adhere to the Exponential distribution, we now modify that arrangement to the Pareto distribution with the shape index of 1.1. The only other traffic besides the web-like flows comes from 50 R flows and 50 D flows that traverse the reverse direction of the bottleneck link throughout the experiment. The access links for the web-like flows have capacity 1 Gbps. The Pareto interarrival times make the traffic bursty and highly dynamic. Figure 10 reflects the high dynamism of the R and D flow counts by showing the widely fluctuating utilization of the bottleneck link by either R or D service. When the flows arrive at average rate 50 Hz, their average cumulative load is low, and they rarely congest the bottleneck link. Arrival rate 100 Hz makes the congestion instances more frequent and intense. Increasing the average arrival rate to 200 Hz creates persistent overload of the bottleneck link. Together with the burstiness of the arrival process, the persistent overload causes heavy losses for the D service.

6.1.6 Sudden changes in the numbers of flows

To investigate how the RD services react to sudden changes in the numbers of R and D flows, we experiment with the following traffic. 100 R flows start at time 0. 50 D flows join them 20 s later. 50 additional D flows arrive at time 40 s and thereby equalize the flow counts for the two services at 100. At time 60 s, 80 D flows finish. 80 other D flows arrive at time 80 s. All R flows leave at time 100 s but 20 new R flows start 40 s later. Finally, 80 extra R flows arrive at time 160 s and reestablish the parity in the numbers of R and D flows. Figure 11 shows that the RD design responds to the changes promptly and appropriately: reflecting the current ratio of the flow counts, the per-flow rate ratio for

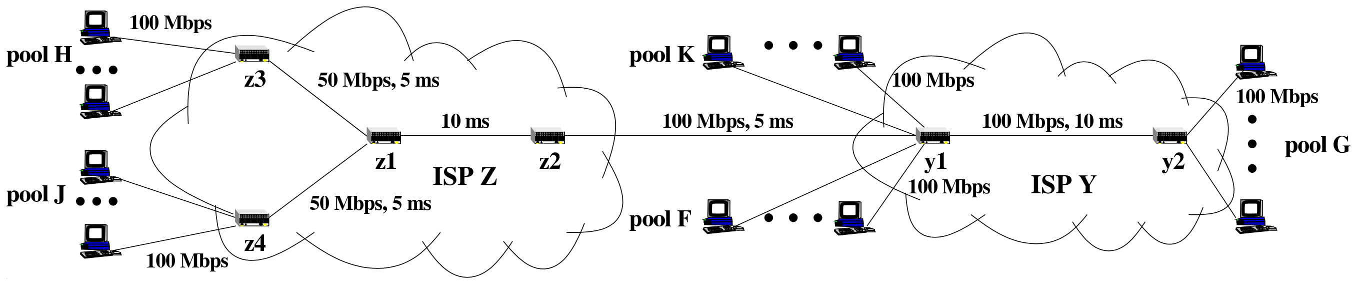


Figure 12: Shared settings of the multi-ISP topologies.

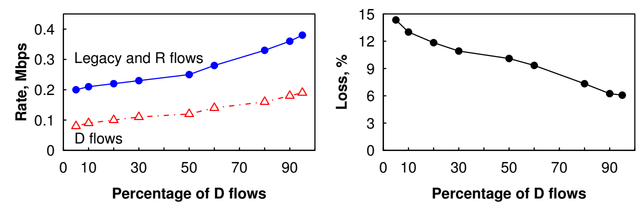
R and D flows becomes 4:1 at time 20 s, reduces to 2:1 at time 40 s, grows to 10:1 at time 60 s, and returns to 2:1 at times 80 s and 160 s, at the latter time by reverting from 1:2. The sudden changes in the flow counts cause sharp spikes in the loss rate for the D service but the spikes are short, and the losses decrease quickly to under 10%. The RD services utilize the bottleneck link fully except between 100 s and 140 s. During that interval, the link carries only D flows and is underutilized due to the small size of the D buffer.

6.2 Performance in multi-ISP topologies

Our investigation of the RD services proceeds by examining their incremental deployability and other properties in topologies where multiple ISPs own the infrastructure. Figure 12 depicts the settings shared by the multi-ISP topologies. The network core belongs to 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. Every user from pools H, J, K, and F transmits a long-lived flow to a separate user in pool G. Hence, while the flows from K and F traverse the infrastructure that belongs only to ISP Y, both ISPs serve the flows from pools H and J. We choose propagation delays for the access links so that propagation RTT for the flows is uniformly distributed between 64 ms and M. In particular, propagation delay for both access links of each flow from pool H or J is chosen between 1 ms and $\frac{M}{4} - 15$ ms, and both access-link propagation delays for a flow from pool K or F are selected between 11 ms and $\frac{M}{4} - 5$ ms. The default setting for the maximum propagation RTT is $M = 300$ ms. The flows arrive according to a Poisson process. The average arrival rate is set by default to 100 Hz for creating a confident expectation that all the flows arrive before the measurement stage of the experiment.

6.2.1 *Legacy traffic and incremental deployment*

Our design principles in Section 2 prescribe that a new service should attract adopters despite continued presence of legacy ISPs and without penalizing legacy traffic. This section experimentally verifies whether the RD services fulfill these design aspirations. Unlike ISP Y, ISP Z does not support the RD services and treats all traffic with the legacy service. 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. Link z1-z2 has capacity 55 Mbps making link y1-y2 a bottleneck for all the flows. We vary ρ, the percentage of D flows. The other $1 - \rho$ flows are either legacy or R flows. 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 indicate their preference for the D service, and the rest of the traffic consists of legacy and R flows.



(a) Per-flow rates (b) Loss rate for D flows

Figure 13: Impact of the incremental deployment on the performance at link y1-y2 of ISP Y.

Figure 13a plots the per-flow rates achieved by the legacy and R flows and D flows at link y1-y2 of ISP Y. As those legacy flows that are interested in low delay opt for the D service and thereby increase the percentage of D flows, the per-flow rate for the remaining legacy flows consistently improves even though some of them enter the network through the legacy ISP Z. Hence, the legacy traffic not only avoids being penalized by the adopters of the D service in accordance with Principle 2 but also benefits itself by becoming able to communicate at higher rates. Besides, Figure 13 reveals that adoption of the RD services yields a win-win outcome for all users: as ρ grows, the per-flow rate increases for the D flows as well, and the increasing size of the D buffer reduces the loss rate of the D service. Therefore, whereas a user opts for the D service to acquire low delay, future adoptions of the D service by other legacy users make the service even more valuable, facilitating the virulent deployment of the RD services.

6.2.2 *Impact of propagation RTT*

From now on, we consider topologies where both ISPs espouse the RD services. ISP Z configures all its four routers to offer the rate-delay differentiation with $k = 2$ and $d = 10$ ms. The long-lived traffic includes 25 R flows and 25 D flows from pool H, 25 R flows and 25 D flows from pool J, 50 R flows from pool K, and 50 D flows from pool F. For each of the flows, the reverse direction of its path carries another long-lived flow of the same class. Also, two sources in pool H transmit web-like R and D flows to pool G. The web-like traffic has the same characteristics as in Sections 6.1.3 and 6.1.4. The capacity of link z1-z2 is set to 100 Mbps. Thus, the network contains two bottleneck links: z1-z2 and y1-y2.

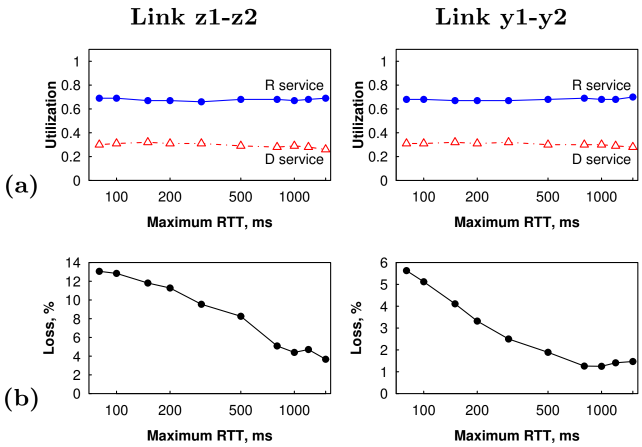


Figure 14: Impact of the propagation RTT diversity: (a) link utilization; (b) loss rate for D flows.

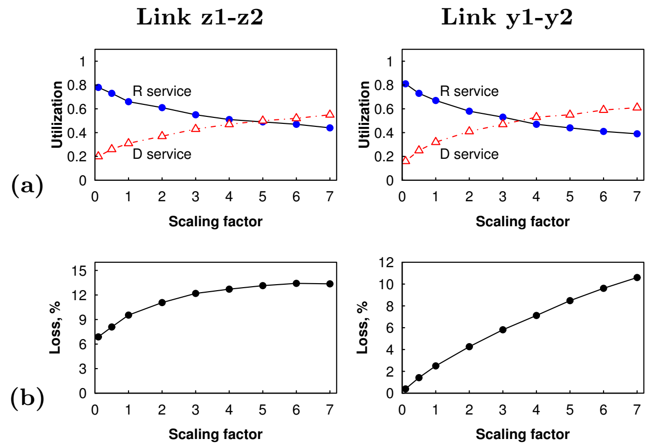


Figure 15: Scalability of the RD services with respect to the number of long-lived D flows: (a) link utilization; (b) loss rate for D flows.

To study the impact of propagation RTT on the RD services, we vary M from 80 ms to 1.5 s. As the maximum propagation RTT grows, the per-flow amount of packets inside the network increases. Consequently, the TCP flows enjoy lower loss rates. Figure 14 confirms this expectation and also shows that the RD services consistently support the intended 2:1 per-flow rate ratio for the R and D flows.

6.2.3 *Population scalability of the RD services*

We also explore population scalability of the RD services, i.e., examine how their performance scales when the numbers of R and D flows change. First, we use a scaling factor σ to modify the traffic mix as follows: the population of the long-lived flows includes 25 R flows and $\lceil 25\sigma \rceil$ D flows from pool H, 25 R flows and $\lceil 25\sigma \rceil$ D flows from pool J, 50 R flows from pool K, and $2 \cdot \lceil 25\sigma \rceil$ D flows from pool F. To preserve the expectation that all the long-lived flows arrive before the measurement stage of the experiment, we reduce average interarrival time to 3 ms for $\sigma > 3$. The long-lived traffic in the reverse direction mirrors again the forward-direction arrangement.

For either of bottleneck links z1-z2 and y1-y2, Figure 15 shows that increasing the number of long-lived D flows redistributes some of the link capacity from the R service to the D service. Due to the presence of the web-like flows, the redistribution depends on σ non-linearly. Also, since links z1-z2 and y1-y2 serve different numbers of flows, the D service gains parity with the R service in utilizing link z1-z2 with a larger scaling factor than for link y1-y2. As σ grows, the per-flow rates of the R and D flows decrease, and the loss rates of the services increase accordingly.

Finally, we conduct a similar study for scalability of the RD services with respect to the number of R flows. Once again, the long-lived traffic arrangement is symmetrical in the forward and reverse directions. In the forward direction, the long-lived traffic includes $\lceil 25\sigma \rceil$ R flows and 25 D flows from pool H, $\lceil 25\sigma \rceil$ R flows and 25 D flows from pool J, $2 \cdot \lceil 25\sigma \rceil$ R flows from pool K, and 50 D flows from pool F. Figure 16 plots utilization and loss rates for links z1-z2 and y1-y2. The analytical rationale for the observed performance profiles is the same as the above explanations for the scaling of the D population.

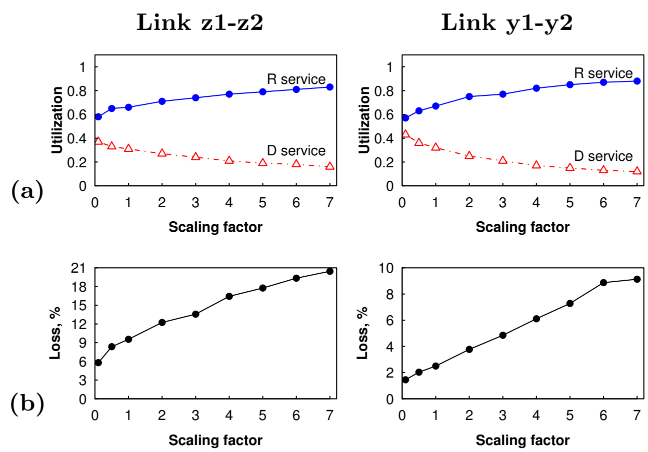


Figure 16: Scalability of the RD services with respect to the number of long-lived R flows: (a) link utilization; (b) loss rate for D flows.

7. RELATED WORK

Network service differentiation has been a topic of extensive research, with the IntServ [4] and DiffServ [3] 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 such complicated link scheduling algorithms as WFQ (Weighted Fair Queuing) [9], WF^2Q (Worst-case Fair Weighted Fair Queueing) [2], Start-time Fair Queueing (SFQ) [20], Virtual Clock (VC) [40], or Earliest Deadline First (EDF) [10] and restrict network access with distributed admission control [17, 28]. 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 such as Core-Stateless Fair Queuing (CSFQ) [35] or Core Jitter Virtual Clock (CJVC) [36]. 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 continued the above trend of focusing on scalability rather than incremental deployment. DiffServ distinguishes services not at the flow granularity but at a much coarser granularity of traffic classes [16]. Various DiffServ designs support either absolute guarantees or relative differentiation between the few traffic classes by employing such algorithmic frameworks as Expedited Forwarding (EF) [26], Assured Forwarding (AF) [6, 23], or Class Selector (CS) [11, 31]. The DiffServ schemes that offer absolute performance guarantees require admission control, e.g., the Premium service of the DiffServ EF designs assures low queuing delay only if the upstream ISPs enforce the maximum rate negotiated for the service [26]. 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. The added D service is neither better nor worse than the R service but is merely different, and the RD architecture gives each user complete freedom to select a higher rate or low queuing delay.

Among other proposals for service differentiation, Alternative Best Effort (ABE) [24] 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) [13] are similar to ABE and suffer from similar limitations.

8. FUTURE WORK

While the design principles from Section 2 led us to the elegant effective solution for network service differentiation, we believe that design for deployability holds great promise for solving 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.

On the level of implementation, the presented RD design neither is the simplest nor minimizes packet loss rates. Our implementation enforces the delay constraint via both sizing of the D buffer and tracking of the arrival times for enqueued D packets. We will study possibilities for enforcing the constraint through buffer sizing alone. Also, since the discard of all packets from the D queue upon downsizing the D buffer is likely to be excessively harsh, we plan to explore more subtle discard policies.

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 will induce 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. While securing the RD design is clearly an important area for future investigation, prior simple performance-based [19, 34] and other [38, 39] security proposals constitute promising starting points.

9. CONCLUSION

This paper revisited the problem of aligning network services with application needs. Based on the principles that a new service should incorporate incentives for users and ISPs to adopt the service despite partial deployment and without penalizing legacy traffic, we designed and implemented the RD services that empower each user to choose between a higher rate or low queuing delay. The RD design does not require changes in end-to-end transport protocols, preserves the IP datagram format, and relies on simple router algorithms that are easy to implement even for high-capacity links. Our extensive evaluation of the RD services confirmed their effective rate-delay differentiation as well as their fitness for incremental virulent deployment.

10. REFERENCES

[1] A. Aggarwal, S. Savage, and T. Anderson. Understanding the Performance of TCP Pacing. In *Proceedings IEEE INFOCOM 2000*, March 2000.

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

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

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

[5] J. W. Byers, M. Luby, and M. Mitzenmacher. A Digital Fountain Approach to Asynchronous Reliable Multicast. *IEEE Journal on Selected Areas in Communications*, 20(8):1528 – 1540, October 2002.

[6] D. D. Clark and W. Fang. Explicit Allocation of Best Effort Packet Delivery Service. *IEEE/ACM Transactions on Networking*, 6(4):362–373, August 1998.

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

[8] D. Wei. *A TCP Pacing Implementation for NS2. http://netlab.caltech.edu/projects/ns2tcplinux/ns2pacing/*.

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

[10] M. Dertouzos. Control Robotics: The Procedural Control of Physical Processes. In *Proceedings IFIP Congress 1974*, August 1974.

[11] C. Dovrolis, D. Stiliadis, and P. Ramanathan. Proportional Differentiated Services: Delay Differentiation and Packet Scheduling. In *Proceedings ACM SIGCOMM 1999*, September 1999.

[12] C. Estan, G. Varghese, and M. Fisk. Bitmap Algorithms for Counting Active Flows on High Speed Links. *IEEE/ACM Transactions on Networking*, 14(5):925–937, October 2006.

[13] V. Firoiu, X. Zhang, and Y. Guo. Best Effort Differentiated Services: Tradeoff Service Differentiation for Elastic Applications. In *Proceedings IEEE ICT 2001*, June 2001.

[14] S. Floyd, M. Handley, J. Padhye, and J. Widmer. Equation-Based Congestion Control for Unicast Applications. In *Proceedings ACM SIGCOMM 2000*, August 2000.

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

[16] S. Floyd and V. Jacobson. Link-Sharing and Resource Management Models for Packet Networks. *IEEE/ACM Transactions on Networking*, 3(4):365–386, August 1995.

[17] 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.

[18] S. Gorinsky, M. Georg, M. Podlesny, and C. Jechlitschek. A Theory of Load Adjustments and its Implications for Congestion Control. *Journal of Internet Engineering*, 1(2):82–93, October 2007.

[19] S. Gorinsky, S. Jain, H. Vin, and Y. Zhang. Design of Multicast Protocols Robust against Inflated Subscription. *IEEE/ACM Transactions on Networking*, 14(2):249–262, April 2006.

[20] P. Goyal, H. M. Vin, and H. Cheng. Start-time Fair Queueing: A Scheduling Algorithm for Integrated Services Packet Switching Networks. In *Proceedings ACM SIGCOMM 1996*, August 1996.

[21] Y. Gu, D. Towsley, C. Hollot, and H. Zhang. Congestion Control for Small Buffer High Speed Networks. In *Proceedings IEEE INFOCOM 2007*, May 2007.

[22] R. Guerin, S. Kamat, V. Peris, and R. Rajan. Scalable QoS Provision Through Buffer Management. In *Proceedings ACM SIGCOMM 1998*, September 1998.

[23] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. Assured Forwarding PHB Group. RFC 2597, June 1999.

[24] P. Hurley, J.-Y. Le Boudec, P. Thiran, and M. Kara. ABE: Providing a Low-Delay Service within Best Effort. *IEEE Network*, 15(3):60–69, May/June 2001.

[25] V. Jacobson. Congestion Avoidance and Control. In *Proceedings ACM SIGCOMM 1988*, August 1988.

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

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

[28] J. Liebeherr, D. E. Wrege, and D. Ferrari. Exact Admission Control for Networks with a Bounded Delay Service. *IEEE/ACM Transactions on Networking*, 4(6):885–901, December 1996.

[29] D. Loguinov and H. Radha. Increase-Decrease Congestion Control for Real-time Streaming: Scalability. In *Proceedings IEEE INFOCOM 2002*, June 2002.

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

[31] K. Nichols, S. Blake, F. Baker, and D. Black. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. RFC 2474, December 1998.

[32] J. Postel. Internet Protocol. RFC 791, September 1981.

[33] 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.

[34] S. Savage, N. Cardwell, D. Wetherall, and T. Anderson. TCP Congestion Control with a Misbehaving Receiver. *ACM Computer Communication Review*, 29(5):71–78, October 1999.

[35] 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.

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

[37] C. Villamizar and C. Song. High Performance TCP in ANSNET. *ACM SIGCOMM Computer Communication Review*, 24(5):45–60, October 1994.

[38] D. Wetherall, U. Legedza, and J. Guttag. Introducing New Internet Services: Why and How. *IEEE Network*, 12(3):12–19, May-June 1998.

[39] X. Yang, D. Wetherall, and T. Anderson. A DOS-limiting Network Architecture. In *Proceedings ACM SIGCOMM 2005*, August 2005.

[40] L. Zhang. A New Traffic Control Algorithm for Packet Switching Networks. In *Proceedings ACM SIGCOMM 1990*, August 1990.