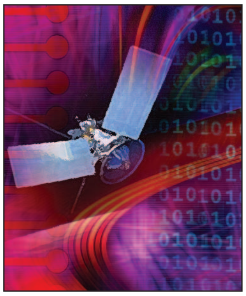

Deploying Diffserv in Backbone Networks for Tight SLA Control

Clarence Filsfils and John Evans • *Cisco Systems*

The Differentiated Services Architecture (Diffserv) enables service providers to offer tighter, more comprehensive service-level agreements (SLAs) for IP service performance. One way it does this is by letting designers engineer IP backbone networks to assure that SLA parameters are met on a per-class basis. This review covers best practices for designing, validating, deploying, and operating Diffserv in the network backbone. It also presents new results from router-based testing that demonstrate how, with Diffserv, high-performance backbone routers can achieve tight SLA capabilities.

The more competitive a particular service's market, the more comprehensive and stringent, or *tight*, the commitments or service-level agreements (SLAs) offered for the service. For Internet Protocol service providers (IP SPs), increasing competition combined with IP applications' heightened importance to business operations has led to an increased demand for – and, consequently, supply of – IP services with tighter SLAs for IP performance. Such SLA commitments are based on delay, jitter (delay variation), packet-loss rate, throughput, availability, and per-flow sequence preservation.

In the backbone, where traffic is aggregated, SLA requirements for a traffic class can be translated into the appropriate bandwidth requirements, and the problem of SLA assurance can effectively be reduced to that of bandwidth provisioning. The Differentiated Services Architecture (Diffserv)[1] helps SPs contend with the demands for tight SLA services by letting designers engineer IP backbone networks to assure that they meet SLA parameters on a per-class basis.

Diffserv deployments consist of both edge and core components. We previously described in *IEEE Internet Computing*[2,3] the considerations when deploying Diffserv at the network's edge. Here, we review best practices for designing and deploying Diffserv in the core of IP and multiprotocol label-switching (MPLS) networks. We also offer results from our recent router-based testing, which demonstrates how, using Diffserv, high-performance backbone routers can achieve tight-SLA capabilities.

Problem Overview

In designing an IP backbone network to support different traffic classes, SPs have three key design objectives:

- Commit to the different SLA requirements for each class.
- Optimize available bandwidth.
- Keep the design as simple as possible.

Empirical evidence suggests that designing an IP backbone network for low delay, low jitter, and low loss can be relatively easy: simply overprovision the bandwidth compared to the actual load.[4-6] Steven Casner and his colleagues showed, for example, that for a best-effort IP backbone with a single service class, worst-case jitter was measured at less than 1 millisecond, with zero loss for probes sent at 1 Mbit per second during a seven-day period between a US tier-one ISP's East and West Coast points-of-presence (POPs).[5] These

Published by the IEEE Computer Society 1089-7801/05/$20.00 © 2005 IEEE

measurements demonstrate the excellent performance an IP backbone can achieve when it's designed with both high-speed links, from optical carrier 12 (OC-12) to OC-192, and conservative capacity-planning rules that upgrade links when utilization reaches 40 to 50 percent, with the goal of ensuring at least twice as much capacity as average load.

Provisioning twice the capacity as the peak aggregate load is a simple design rule that could help SPs achieve tight SLAs for delay, jitter, and loss. Let's say, for example, that an SP has 10 Mbps of voice-over-IP (VoIP) traffic and 100 Mbps of standard data traffic between two POPs. Given the above rule, it would need twice the sum of the VoIP and data traffic loads — or 220 Mbps — to assure low delay, low jitter, and low loss for the VoIP class. Aggregate overprovisioning of bandwidth, however, represents an expensive option for the SP and can be difficult to ensure in all cases, such as in the presence of denial-of-service attacks and network failures. In such cases, all traffic classes share the same fate, and if unforeseen congestion occurs, it affects all classes.

Diffserv provides a solution to this problem, in that it lets SPs build virtual backbones for each class on a single physical backbone. This gives SPs the flexibility to have different under- or overprovisioning ratios (the ratio of offered load to available capacity) for each service class. Given our previous example, the SP could thus overprovision the VoIP class capacity by a factor of two, hence ensuring that the class receives low-delay, low-jitter, and low-loss service, while overprovisioning the data class capacity by a lower factor, such as 1.2 (a realistic figure that still offers good service). This would result in 140 Mbps of total required bandwidth, providing a savings of 80 Mbps, or 36 percent.

We offer this example to highlight two key points:

- Diffserv's backbone deployment is conceptually simple.
- Diffserv's concepts have already been proven by deployed, best-effort IP network experience.

Diffserv deployment is simple in that it lets an SP support different service levels merely by using different under- or overprovisioning ratios per class. The higher the available capacity compared to the offered load, the tighter the SLA the SP can support. The concepts are proven in that empirical evidence from high-speed best-effort (single service class) SP IP/MPLS backbones indicates that extremely good jitter and loss targets can be achieved using simple bandwidth overprovisioning rules. Thomas Telkamp, for example, recently studied the bandwidth overprovisioning level needed at different link rates to achieve defined delay and loss commitments.[7] Diffserv simply extends the concept of overprovisioning to multiple classes to offer more efficient network capacity use.

Diffserv Architecture

Diffserv is currently the preferred technology for large-scale IP quality-of-service deployments, such as SP backbone networks. It achieves scalability by performing complex QoS functions, including traffic classification, marking, and conditioning at the network's edges. Diffserv uses the Diffserv Code Point (DSCP) in the IP packet header to mark and classify traffic into a limited number of traffic aggregates or classes. Within the network's core, it applies scheduling and queuing-control mechanisms to traffic classes based on the DSCP marking, and intelligently handles all network-layer traffic conditioning and dropping using Diffserv IP QoS mechanisms. Diffserv is not prescriptive in defining the scheduling and queuing control algorithms that should be implemented at each core router; rather, it uses a level of abstraction to define externally observable forwarding behaviors — called *per-hop behaviors* (PHBs) — that can be applied to traffic at each router. Currently, Diffserv defines three PHBs:

> **Aggregate overprovisioning of bandwidth represents an expensive option for the service provider and can be difficult to ensure in all cases.**

- The *expedited forwarding* PHB[8] supports traffic with low delay, low jitter, low loss, and assured bandwidth requirements, such as VoIP. Router vendors typically implement the EF PHB using a strict priority-queue mechanism to ensure priority scheduling and isolation from the other queues.

- The *assured forwarding* PHB[9] supports data traffic with assured bandwidth requirements, such as absolute- or relative-minimum bandwidth guarantees with a work-conserving property. Router vendors typically implement the AF PHB with a weighted-fair-queuing-like scheduling mechanism.
- The *default* PHB represents Diffserv's default forwarding behavior for packets not identified as belonging to another class; this represents a "best effort" forwarding behavior with no guaranteed SLA assurances.

These PHBs support the different class SLAs that SPs offer.

Core Service Classes and SLA specification

Common among QoS-enabled IP services that SPs offer today is support for service classes designed to meet the needs of three aggregate traffic types.

- *Real-time*. This class targets applications such as VoIP and video. SPs define service for this class in terms of low delay and jitter (typically less than 5 ms within the backbone), and close to zero loss. The class might include a commitment for per-flow sequence preservation.
- *Business data*. This class represents business-critical interactive applications such as IBM's System Network Architecture; Systems, Applications, and Programming Facilities' real-time system, version three (SAP R/3); Telnet; and possibly intranet Web applications. SPs define service for this class in terms of defined delay and close to zero loss. The class might include a commitment for per-flow sequence preservation.
- *Standard*. This class represents all traffic not classified as real-time or business. SPs define service for this class in terms of a loss rate; it might also include a commitment for per-flow sequence preservation. Because delay and jitter are unimportant for this service, they are not defined. (We intentionally don't use the term "best effort" in relation to this class as doing so would imply no SLA commitments.)

We now consider how to practically apply Diffserv PHBs to support these QoS classes.

Prioritized Diffserv Core Model

Most backbone Diffserv deployments have adopted a prioritized differentiated services model. To illustrate the model's underlying concepts, consider a typical design consisting of three queues on each backbone interface to support the three traffic types: an EF class queue for real-time traffic, and two AF class queues, one for business data and another for standard data.

For real-time traffic, the SP uses a strict priority-queuing behavior to ensure the lowest delay and jitter service. Once this class is served, the SP allocates the remaining bandwidth, with a minimum assurance of 90 percent going to the business data class and 10 percent to the standard class. Because the real-time and business data loads are expected to be less than their available class capacity, these classes effectively experience zero loss. A holistic per-class capacity-planning process is essential to ensure that this is actually the case. The capacity-planning process might take into account single or multiple network component (link and node) failures, depending on the SP's particular goals. Assuming the routers use a work-conserving scheduler, the standard class can reuse all unallocated or unused interface capacity once the real-time and business data classes have been serviced.

A characteristic of this design is that when congestion occurs (assuming the per-class capacity-planning process is operating correctly), the loss is restricted to the standard class, thereby assuring the SLAs for the real-time and business classes. Further, because this design significantly overprovisions the bandwidth available to the real-time and business classes, SPs don't need to reconfigure the core router schedulers as per-class loads increase over time. When load thresholds relative to available class bandwidth are exceeded, the problem is one of capacity provision; it is solved by increasing the link bandwidth.

Core Design Details

As an example, Figure 1 shows the Diffserv policy that the SP would apply to all backbone links to achieve our example SLA specification. To simplify the design description, we use a metalanguage (see the "Diffserv Metalanguage" sidebar).

Real-Time Traffic

The SP classifies the real-time class as traffic that matches the defined DSCP `Dr`; we assume the traffic has already been marked, either at the source host or by an edge Diffserv policy applied on ingress to the network. The SP assures the

Diffserv Metalanguage

To ease the Diffserv design's description, we use the following metalanguage:

- `Policy <policy_name>` defines a Diffserv policy that can be applied to a particular interface.
- `Class <class_name>` refers to a traffic aggregate that matches the classification profile `<class_name>`.
- `EF(r,b)` indicates that the class must receive an expedited fowarding (EF) per-hop behavior (PHB) with an assured minimum rate of `r` percent of the link speed and a burst `b`.
- `AF(m,p)` indicates that the class must receive an assured forwarding (AF) PHB with an assured minimum absolute rate of `m` percent of the link speed and a relative assurance of `p` percent of any bandwidth unused by or unallocated to other classes.
- `match {not} DSCP (D)` refers to a class's Diffserve code point (DSCP) classification criteria.
- `police(r,b) conform <action> exceed <action>` refers to a single-rate, two-color conditioner definition. Possible actions are "drop" and "transmit."
- `RED` implements random early detection as a congestion-avoidance technique.[1]

Figure 1 in the main text uses this Diffserv metalanguage to illustrate how underlying Diffserv behaviors can be defined and combined in a design to meet per-class SLA commitments.

Reference

1. S. Floyd and V. Jacobson, "Random Early Detection gateways for Congestion Avoidance," *IEEE/ACM Trans. Networking*, vol. 1, no. 4, 1993, pp. 397–413.

per-hop latency commitment by defining the class as `EF`, requesting the lowest latency from the scheduler. It allocates the class's forwarding resources to ensure that, given the expected load, there will be no congestion and the class will receive good deterministic service with low delay, jitter, and loss.

The SP applies a policer to the real-time class to limit the maximum rate, `Rr`. This policer ensures that this class can't starve the other classes of bandwidth. In normal operation, and with correct core network-capacity planning, the SP would typically know the real-time traffic demands and provision capacity accordingly. The policer therefore serves as a safety precaution, offering the other classes protection in the case of unexpected failures that lead to adversely high real-time load. The SP typically sets the real-time policer to the maximum acceptable EF rate under failure conditions, which is typically greater than or equal to 50 percent of the link rate. The allowed burst size, `Br`, is set to the maximum acceptable per-hop jitter target for the class.

Business Traffic

The SP classifies the business class as traffic matching DSCP `Db`. The SP assures the business class bandwidth by treating it with an AF PHB. In doing so, it provides a relative assurance of `Rb` percent of any bandwidth unused by or unallocated to the other classes; here, this is a percentage of the remaining bandwidth once the real-time class has been serviced. As we described earlier, with the prioritized Diffserv model, `Rb` is typically 90 percent. Because the SP assumes that most business class traffic is TCP/IP,[10] it uses the random early detection (RED)[11] congestion-control mechanism within the business class queue rather than a fixed queue depth with tail-drop behavior. This maximizes TCP throughput within the class when congestion occurs.

If an edge service is defined with in-contract and out-of-contract capabilities (similar to frame relay's discard eligibility concept), the SP can use weighted RED (WRED) to establish two different RED profiles. SPs can define a RED profile's characteristics by a minimum average queue depth threshold (min_{th}), a maximum threshold (max_{th}), and a discard probability (max_p) at max_{th}. WRED lets an SP support multiple RED profiles in the same queue, with separately defined min_{th}, max_{th}, and max_p per profile. This results in different drop characteristics (and consequently, different drop probabilities) per profile. For the business class queue, for example, an SP might have:

- an aggressive profile for out-of-contract traffic, and
- a gentler profile for in-contract traffic that ensures that out-of-contract packets are

```
policy core-policy
  class Real-time
    match dscp Dr
    police(Rr, Br)
      conform transmit
      exceed drop
    ef(Rr, Br)
  class Business
    match dscp Db
    AF(0, Rb)
    RED
  class Standard
    match not dscp Dr
    match not dscp Db
    AF(0, Rs)
    RED
```

Figure 1. Example Diffserv policy. The service provider would apply this policy to all backbone links to achieve the example service-level agreement specification.

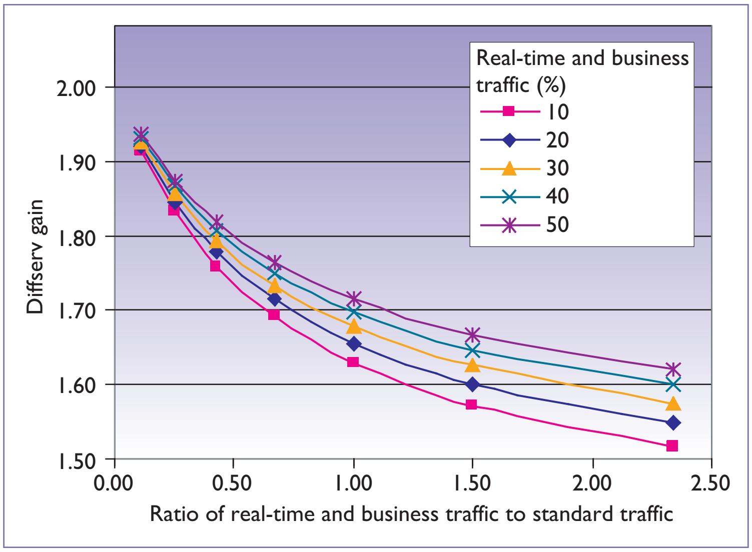

Figure 2. Diffserv bandwidth gain. The gain for different traffic loads is expressed as the bandwidth required without Diffserv divided by the bandwidth required with Diffserv to achieve the same SLAs.

dropped first if class congestion occurs.

The SP would achieve this by choosing a maximum threshold value for out-of-contract traffic that's less than the minimum threshold for in-contract traffic.

Typically, SPs create a discrete class for routing protocol and management traffic at the network's edge. They can also do this in the backbone, or alternatively, they can service such traffic from one of the existing low-loss classes (such as the business class in our example).

Standard Traffic

The standard class is classified as all traffic that does not explicitly match the real-time or business classes. SPs can provide the standard class bandwidth assurances by treating the class with an AF PHB, with a relative assurance of `Rs` percent of any remaining bandwidth unused by or unallocated to the other classes; here, this is a percentage of the remaining bandwidth once the real-time class has been serviced. With the prioritized Diffserv model, `Rs` is typically 10 percent. An SP can use RED within the class to ensure that TCP throughput is maximized when congestion occurs.

Backbone Marking Scheme

The SP can choose the actual DSCP class markings it uses in the core; using the Diffserv-recommended 6-bit EF and AF markings isn't mandatory. An SP could, for example, use class-selector code points,[12] which are functionally equivalent to IP precedence markings. Class-selector code points ease DSCP marking translation from and to technologies that support only 3-bit marking schemes, such as MPLS, 802.1Q, and 802.17 (resilient packet ring); they also offer backward compatibility with deployed designs and systems that support only IP precedence marking.

For our purposes here, we assume that SPs perform classification, conditioning, and marking functions at the network's edge. An edge design might support more service classes than the backbone. In this case, an SP might, for instance, map distinct edge VoIP and video classes into an aggregate backbone real-time class. It could offer several such classes at the network's edge and thus allow application isolation or ease the bandwidth budgeting among different customer departments, all supported with a single backbone business class. In such cases, the SP's edge router must be capable of mapping several edge classes into a single aggregate backbone class. It can realize this mapping in two ways:

- A backbone class can match several DSCPs. If, for example, a DSCP value of 40 represents VoIP and a value 32 represents video at the network's edge, the backbone aggregate real-time class would match both DSCP 40 and DSCP 32.
- When the backbone uses MPLS, the edge SP router can set the three-bit MPLS experimental (`EXP`) field as a function of the received DSCP. For example, if the SP uses the `EXP` value of 5 for the aggregate real-time backbone class, the SP's edge router will impose MPLS labels with an `EXP` value of 5 for packets received with either DSCP 40 (representing the edge VoIP class) or DSCP 32 (representing the edge video class).

There are, however, considerations other than the Diffserv policy configurations that must be factored in when contemplating a Diffserv backbone deployment.

Deployment Considerations

We obviously must consider whether the cost of deploying Diffserv outweighs its benefits. There is no generic answer here, as the benefits gained vary by deployment. In our earlier example, if the Diffserv deployment cost exceeds the cost of the additional 80 Mbps of bandwidth (and the router interfaces that would be needed to terminate the extra bandwidth) then deploying Diffserv might not make sense.

Factors that affect Diffserv's overall economic viability include its economic benefit, impact on router performance, complexity of deployment, and capacity planning and monitoring requirements. Training is also an issue: Diffserv is a new technology, and SPs must plan to train designers and operational staff to support its backbone deployment.

Economic Benefit

For most practical deployments, Diffserv offers the greatest economic benefit when the traffic requiring the highest SLA targets represents a minor proportion of the overall traffic load. As our example demonstrates, without Diffserv, the designer provisions capacity equal to the aggregate load across all classes, multiplied by the overprovisioning ratio of the tightest SLA class. Such a strategy can be extremely expensive when the tightest SLA class is a small proportion of the aggregate traffic. Conversely, using Diffserv offers no benefit when all classes require the same level of service, and hence the same overprovisioning ratio.

SPs can realize Diffserv's benefit either in terms of using less bandwidth to achieve the same SLAs, or in supporting more aggregate traffic for the same provisioned bandwidth. Figure 2 shows the Diffserv bandwidth gain for different relative loads of real-time, business data, and standard traffic with per-class bandwidth overprovisioning ratios of 2, 1.5, and 1, respectively. As the figure shows, the lower the proportion of real-time and business data traffic, the greater Diffserv's bandwidth savings. Obviously, if all traffic is real-time, Diffserv offers no benefit.

Router Performance

If applying Diffserv EF and AF forwarding behaviors impacts router forwarding performance, the router will support less aggregate throughput with Diffserv enabled and, consequently, the network's per-port cost will increase. Today's high-performance backbone routers typically implement EF/AF forwarding behaviors in ASICs, ensuring that there is no forwarding penalty associated with Diffserv functionality support.

Deployment Complexity

Typical backbone Diffserv deployments require relatively minor changes to existing router configurations, as our example policy shows. In backbone Diffserv deployments, these configurations are typically applied once, remaining static thereafter.

Furthermore, SPs can seamlessly migrate a backbone to Diffserv by undertaking the backbone configuration independently of the network edge configuration to ensure that traffic is appropriately conditioned and marked on ingress to the network. Diffserv's benefit, however, will not be realized until both edge and backbone components are complete.

Capacity Planning and Monitoring

In terms of operational process, capacity planning a Diffserv backbone is similar to a single-class standard IP network. SPs might need to enhance their backbone network-management systems and operational support systems to retrieve the bytes/packets transmitted and dropped per class rather than per interface. When per-class load thresholds are reached, the provisioning of additional network bandwidth is triggered.

Further, SPs should consider deploying an active SLA probing system to monitor (and report) delay, jitter, and loss. Some router vendors implement software agents that send and receive probes with user-definable DSCP and protocol identities (such as FTP, HTTP, and DNS). Active SLA probing results give SPs heuristics that let them tune capacity-planning thresholds by correlating per-class load with SLA probing reports on delay, jitter, and loss.

In addition to per-link statistics and active SLA monitoring systems, accurate core capacity planning depends on an SP understanding the core traffic matrix — the matrix of aggregated ingress to egress traffic demands across the core network (such as POP-to-POP or edge-router-to-edge-router) — and mapping it to the underlying network topology. The benefit of a core traffic matrix over per-link statistics is that it enables trending and "what if" scenario simulation, letting SPs analyze how the rerouting of traffic when a core network component fails might affect the (aggregate or per-class) utilization of the rest of the network. This is useful from an operational-capacity-planning perspective — SPs are increasingly considering single-element failures when they provision core networks — as well as for understanding the relative benefits of different traffic-engineering approaches.

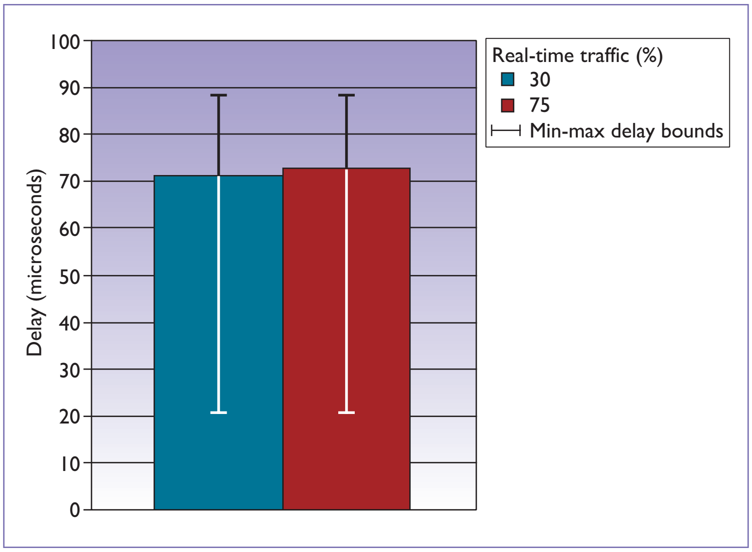

Figure 3. Expedited forwarding (EF) class latency. The delay recorded for the real-time class traffic with 200-byte IP packets through the target link. The thick bars represent the average delay measured for two traffic profiles: 30 percent real-time traffic (left) and 75 percent real-time traffic (right). Both profiles have 45 percent business traffic and 150 percent standard traffic. The thin bars show the measured minimum and maximum delay bounds. Even with 270 percent congestion, the worst-case delay of real-time packets is below 100 microseconds.

Performance Results

Results from our router-based testing illustrate the tight latency, jitter, and loss capabilities of today's router technology. In our tests, we used a Cisco 12416 router with four-port OC-48 line cards. The router's distributed architecture supports the EF and AF per-hop behaviors implemented in ASICs on each line card using a deficit round-robin scheduling algorithm[13] that Cisco modified to support a strict priority queue for EF class traffic.

We used a Diffserv configuration equivalent to our policy example in Figure 1 for all tests. For real-time traffic, we used a packet size of 200 bytes; the business data and standard traffic followed an Internet-mix packet size distribution: 58 percent were small (40 bytes), 33 percent were medium-sized (552 bytes), and 9 percent were large (1,500 bytes). The router had three ingress OC-48 ports receiving traffic from a traffic generator. It aggregated this traffic and forwarded it onto the single-hop link under test — an OC-48 packet over synchronous optical network (POS) link.

Our tests focused on two characteristics that are key to Diffserv's successful deployment: EF traffic latency and bandwidth allocation between AF queues.

EF Traffic Latency

Our first test measured the one-way delay of the real-time class in the presence of interface congestion. To verify a backbone SLA target for EF jitter of less than 5 ms, we assumed 10 router hops through the backbone and set a worst-case delay success criterion of 500 microsecond (μs) per hop to ensure that the 5-ms delay-jitter target was never exceeded, assuming additive jitter. While jitter might not be additive in practice, this scenario represents the absolute worst case, and if the deployment can achieve the worst-case per-hop budget, the 5-ms backbone budget is assured.

The results demonstrate the low delay that today's routers can achieve using an EF-compliant strict-priority queue scheduler. Figure 3 charts the recorded delay for the real-time class (EF) traffic with 200-byte IP packets through the target link, which is congested with two different traffic profiles.

As Figure 3 clearly shows, the priority queuing mechanisms give the real-time (EF) traffic low delay and low jitter service. Even under 270 percent congestion of the outbound port — with a traffic profile of 75 percent real-time traffic, 45 percent business traffic, and 150 percent standard traffic (measured as a percentage of the OC-48 link rate) — the maximum delay of real-time packets remained below 100 μs. These results are significantly better than our target of 500 μs.

Bandwidth Allocation between AF Queues

Our final test measured bandwidth-allocation accuracy for AF class traffic (business data and standard). Successful Diffserv deployment depends on it managing the relative under- and overprovisioning ratios between the classes, which in turn depends on the scheduler implementation's accuracy in allocating AF bandwidth.

We loaded the target link with bursty priority-queue traffic up to 30 percent of the OC-48 link rate, and variable ratios of business and standard class traffic to an aggregate interface load of more than 200 percent of the OC-48 link rate. We then varied the ratio of the configured bandwidth allocations between the business and standard classes, and measured the actual bandwidth allocation ratio compared to the expected result.

Figure 4 graphs the difference between the measured business bandwidth allocation and the expected bandwidth allocation as a percentage of the OC-48 link rate. As the figure shows, the router's accuracy in bandwidth allocation between different AF classes is within about 0.1 percent of the expected result in most cases.

As SP offerings become more competitive and customer requirements more demanding, SLA parameters are increasingly significant for IP service performance. Our analysis shows that Diffserv offers bandwidth savings, relatively simple backbone deployment, and, as our test results show, tight SLA capabilities. Several other technologies – including Fast Interior Gateway Routing Protocol (IGP) convergence, MPLS traffic engineering, MPLS traffic engineering fast reroute, and IGP metric-based traffic engineering – complement Diffserv and further enable IP networks to support tight SLAs.

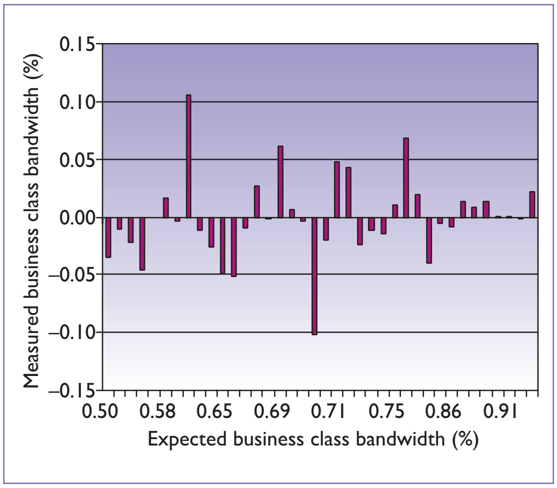

Figure 4. Bandwidth accuracy for assured forwarding (AF) queues. The measured business bandwidth allocations, as a percentage of the OC-48 link rate, are within roughly 0.1 percent of the expected allocation in most cases.

Acknowledgments

We thank Peter De Vriendt and Kris Michielsen for their work, which has contributed to this article.

References

1. S. Blake et al., *An Architecture for Differentiated Service*, IETF RFC 2475, Dec. 1998; www.ietf.org/rfc/rfc2475.
2. J. Evans and C. Filsfils, "Deploying Diffserv at the Network Edge for Tight SLAs, Part I" *IEEE Internet Computing*, vol. 8, no. 1, 2004, pp. 61–65.
3. J. Evans and C. Filsfils, "Deploying Diffserv at the Network Edge for Tight SLAs, Part II" *IEEE Internet Computing*, vol. 8, no. 2, 2004, pp. 61–69.
4. T. Bonald, A. Proutiere, and J. Roberts, "Statistical Guarantees for Streaming Flows Using Expedited Forwarding," *Proc. IEEE Conf. Computer Comm.* (INFOCOM), IEEE CS Press, 2001, pp. 1104–1112.
5. S. Casner, C. Alaettinoglu, and C.-C. Kuan, "A Fine-Grained View of High-Performance Networking," North American Network Operators' Group, May 2001; www.nanog.org/mtg-0105/ppt/casner.
6. A. Charny and J.-Y. Le Boudec, "Delay Bounds in a Network with Aggregate Scheduling," *Proc. 1st Int'l Workshop Quality of Future Internet Services* (QOFIS), LNCS 1922, Springer-Verlag, 2000, pp. 1–13.
7. T. Telkamp, "Traffic Characteristics and Network Planning," *Proc. Internet Statistics and Metrics Analysis Workshop*, North American Network Operators' Group, 2002; www.caida.org/outreach/isma/0210/talks/thomas.pdf.
8. B. Davie et al., *An Expedited Forwarding PHB*, IETF RFC 3246, Mar. 2002; www.ietf.org/rfc/rfc3246.
9. J. Heinanen et al., *Assured Forwarding PHB Group*, IETF RFC 2597, June 1999; www.ietf.org/rfc/rfc2597.
10. S. McCreary and k. claffy, "Trends in Wide-Area IP Traffic Patterns – A View from Ames Internet Exchange," *Proc. 13th ITC Specialist Seminar on Internet Traffic Measurement and Modeling*, Cooperative Assoc. for Internet Data Analysis, 2000, pp. 1–11.
11. S. Floyd and V. Jacobson, "Random Early Detection gateways for Congestion Avoidance," *IEEE/ACM Trans. Networking*, vol. 1, no. 4, 1993, pp. 397–413.
12. K. Nichols et al., *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*, IETF RFC 2474, Dec. 1998; www.ietf.org/rfc/rfc2474.
13. M. Shreedhar and G. Varghese, "Efficient Fair Queuing Using Deficit Round Robin," *Proc. ACM Special Interest Group Data Comm.* (SIGCOMM), ACM Press, 1995, pp. 231–242.

Clarence Filsfils is a distinguished engineer at Cisco Systems, where he focuses on IP core routing and capacity-management design. He has an engineering degree in computer science from the University of Liege, Belgium, and a business degree from the Solvay Business School, Brussels. Contact him at cf@cisco.com.

John Evans is a consulting engineer at Cisco Systems, where he focuses on IP network design and development with a special interest in core routing and traffic management. His current focus is advanced QoS for broadband and associated work in the DSL Forum. He received a B.Eng. (Hons) in electronic engineering and an M.Sc. in communications engineering from the University of Manchester Institute of Science and Technology, UK. Contact him at joevans@cisco.com.