Internet: Fairer is Faster

A complete overhaul of how we share Internet capacity is in progress

By Bob Briscoe, Networks Research Centre, BT

A reduced version of this article appeared as "A Fairer, Faster Internet Protocol" in IEEE Spectrum magazine, Dec 2008

The Internet is founded on a very simple premise: shared communications links are more efficient than dedicated channels that lie idle much of the time. We share local area networks at work and neighbourhood links from home. Indeed, a multi-gigabit backbone cable is shared among thousands of folks surfing the Web, downloading videos, and talking on Internet phones.

But there's a profound flaw in the protocol that governs how people share the Internet's capacity. The protocol allows you to *seem* to be polite, even as you take far more resources than others.

Network providers like Verizon or BT either throw capacity at the problem or patch over it with homebrewed attempts to penalize so-called bandwidth hogs or the software they tend to use. From the start it needs to be crystal clear that those with an appetite for huge volumes of data are *not* the problem. There is no need to stop them downloading vast amounts of material, *if they can do so without starving others*.

But no network provider can solve this on their own. At the Internet standards body, work has started on fixing the deeply entrenched underlying problem. A leading proposal claims to have found a way to deploy a tweak to the Internet protocol itself—the Internet's 'genetic material'. The intent is to encourage a profound shift in the incentives that drive capacity sharing. It should also make investment in Internet capacity safer, driving growth in both the developed and the developing world. Not only should it make the Internet simpler, but much faster too.

You might be shocked to learn that the designers of the Internet intended that your share of Internet capacity would be determined by what your own software considered fair. They gave network operators no mediating role between the conflicting demands of the Internet's hosts—now over a billion personal computers, mobile devices, and servers.

The Internet's primary sharing algorithm is built in to the Transmission Control Protocol, a routine on your own computer that most programs run—although they don't have to. It is one of the twin pillars of the Internet, the other being the Internet Protocol, which delivers packets of data to particular addresses. The two together are often called 'TCP/IP'.

Your TCP routine constantly increases your transmission rate until packets fail to get through some pipe on the way—a tell-tale sign of congestion. Then TCP very politely halves your bit rate. The billions of other TCP routines around the Internet behave in just the same way, in a cycle of taking then giving that fills the pipes while sharing them equally. It's an amazing global outpouring of self-denial, like the 'after you' protocol two people use as they approach a door at the same time—but paradoxically, the Internet version happens between complete strangers, even fierce business rivals, in all billions of times every second.

The commercial stakes could hardly be higher. Services like YouTube, eBay, Skype, and iTunes are all judged by how much Internet capacity they can grab for you, as are the Internet phone and TV services provided by the carriers themselves. Some of these companies are opting out of TCP's sharing regime, but most still allow TCP to control how much they get—about 90 percent of the 200 000 terabytes that cross the Internet each day.

This extraordinary spirit of global cooperation stems from the Internet's early history. In October 1986, Internet traffic persistently overran available capacity—the first of a series of what were called congestion collapses. The TCP software of the day continued to try to retransmit, aggravating the problem and causing everyone's throughput to plummet for hours on end. By mid-1987 Van Jacobson, then a researcher at Lawrence Berkeley National Laboratory, had coded a set of elegant algorithms in a patch to TCP.

Jacobson's congestion control accorded well with the defining design principle of the Internet: traffic control is consigned to the computers around the edges of the Internet (using TCP), while network equipment only routes and forwards packets of data (using IP).

The combination of near-universal usage and academic endorsement has gradually elevated TCP's way of sharing capacity to the moral high ground, altering the very language engineers use. From the beginning, equal rates were not just 'equal', they were 'fair'. Even if you don't use TCP, your protocol is considered suspect if it's not 'TCP-friendly'—a cosy-sounding idea meaning it consumes about the same data rate as TCP would.

Only recently has it started to be accepted that an equal bit rate for each data flow is likely to be extremely *unfair*, by any realistic definition. It's like insisting that boxes of food rations must all be the same size, no matter how often people return for more or how many boxes they take each time.

Consider a neighbourhood network with 100 customers, each of whom has a 2 megabit per second broadband access line connected to the Internet by a single shared 10 Mb/s regional link. The network provider can get away with such a thin shared pipe because most of the customers—let's say 80 of the 100—don't use it continuously, even over the peak period. These people might think they are constantly clicking at their browsers and getting new e-mail, but their data transfers might be active perhaps only 5 percent of the time. However, there are also 20 heavy users who download continuously—perhaps using file-sharing programs that run unattended. So at any one moment, data is flowing to about 24 users—all 20 heavy users, and 4 of the 80 light ones.

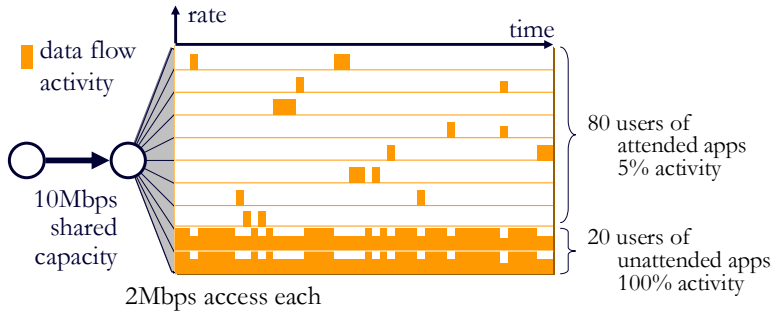

Figure 1—TCP's First Overlooked Factor: Activity over time. The plot visualises the shares of the ISP's 10Mbps link taken by 10 sample users. Those active at any one moment take equal rates, regardless of how active each user is over time.

TCP gives 20 shares of the bottleneck capacity to the heavy users and only 4 to the light ones (Figure 1). In a few moments, the 4 light users will have stepped aside and another 4 will take over their shares. However, the 20 heavy users will still be there to claim their next 20 shares. They might as well have dedicated circuits!

It gets even worse. Any programmer can just run the TCP routine multiple times to get multiple shares. It's much like getting around a food-rationing system by duplicating ration coupons. This trick has always been recognized as a way to sidestep TCP's rules—the first Web browsers opened four TCP connections. Therefore, it would have been remarkable if this ploy had *not* become more common.

A number of such strategies evolved through innocent experimentation. Take peer-to-peer file sharing—a common way to exchange movies over the Internet, one that accounts for a large portion of all traffic; it involves downloading a file from several peers at once. This parallel scheme, sometimes known as swarming, was invented in 1999; it had become routine by 2001, built into such protocols as BitTorrent.

The networking community didn't immediately view connecting with many machines as a circumvention of the TCP-friendliness rule. After all, each transfer used TCP, so each data flow 'correctly' got one share of any bottleneck it encountered. But using parallel connections to multiple machines was a new degree of freedom that hadn't been thought of when the rules were first written. Fairness should be defined as a relation between people, not data flows.

Peer-to-peer file sharing exposed both of TCP's failings. First, as I've noted, a file-sharing program might be active 20 times as often as your Web browser, and second, it uses many more TCP connections, typically 5 or even 50 times as many. Over time, peer-to-peer thus takes 100 or 1000 times as many shares of Internet bottlenecks as a browser does (Figure 2).

Returning to our 100 broadband customers: if they were all just browsing the Web and exchanging e-mail, each would get nearly the full benefit of a 2 Mb/s access pipe—if 5 customers were active at a time, they'd just squeeze into the 10Mb/s shared pipe. But if even 20 users started continuous parallel downloading, the TCP algorithm would send everyone else's bit rate plummeting to an anaemic 20 kilobits per second—worse than dial-up!

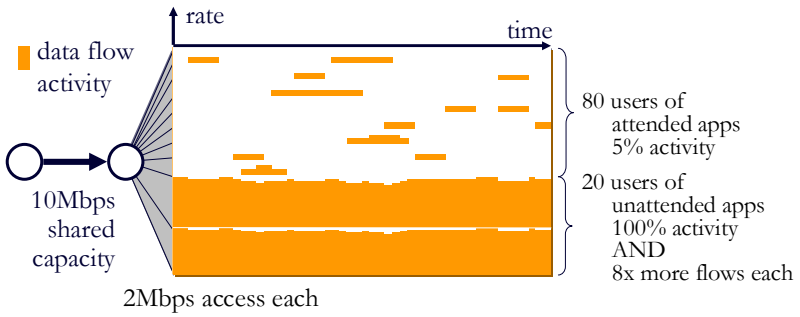

Figure 2—TCP's Second Overlooked Factor: Multiple TCP Instances.

The TCP algorithms of all 100 users take shares of the ISP's 10Mbps link. But software can take say 8 times more shares by just running the TCP algorithm 8 times in parallel.

Compounded with 20 times higher activity over time, the unattended applications shown use the shared link with 160 (=20x8) times greater intensity. But don't blame the software, blame the TCP-sharing regime.

Why can't the network provider simply upgrade that stingy 10 Mb/s shared pipe? Of course, sometimes upgrades are necessary. But lack of capacity, like the emergence of peer-to-peer protocols, is merely a symptom, not the root cause of the problem. Without fixing TCP's sharing rules, adding capacity is as futile as throwing water uphill.

Imagine two competing Internet service providers, both with this 80:20 mix of light and heavy users. One provider quadruples its capacity; the other doesn't. But TCP still shares out the upgrader's capacity in the same way. So the light users, who used to have a measly 20 kb/s share, now get a measly 80 kb/s—still barely better than dial-up. But the 80 light users must now pay substantially more for four times the long-distance capacity, which they hardly get to use. No rational network operator would upgrade under these conditions—it would lose most of its customers.

Although capacity investment makes no sense when TCP shares it out so badly, work-rounds have been found. Many Far Eastern governments have subsidised capacity growth. Weak broadband competition in the US has allowed Internet providers to fund continued investment through higher fees without the risk of losing customers to competitors.

But in competitive markets, common in Europe, some ISPs have tried to attack the deeper issue: the way their capacity is shared. They often don't allow TCP to give all the new capacity straight to the heavy users. Instead they impose their own sharing regime on their customers, thus overriding the worst effects of TCP's misguided sharing system. Some partition up the pipe to prevent heavy users encroaching on lighter ones, losing much of the benefit of sharing. Others limit, or 'throttle', the peak-time bit rate of peer-to-peer customers.

In the most cost-sensitive markets, such as the UK or New Zealand, restrictions against certain programmes and permits for others have already been embedded into the Internet. This practice is likely to spread as cost pressures grow, whether due to competition or increasing volumes of video data. The pace of innovation will be dragged down as it becomes near-impossible for new uses of the Internet to find a clear path through all the blocks and throttles.

There's a far better solution than fighting against TCP. It would allow light browsing to go blisteringly fast, but hardly prolong heavy downloads at all. The solution comes in two parts. Ironically, it begins by making it *easier* for programmers to run TCP multiple times—a deliberate break from TCP-friendliness. Programmers who use this new protocol to transfer data will be able to say "Behave like 12 TCP flows" or "Behave like 0.25 of a TCP flow." They set a new parameter—a weight—so that whenever your data comes up against others all trying to get through the same bottleneck, you'll get, say, 12 shares, or a quarter of a share. Remember, this isn't network-based prioritisation. These weights are used by the new TCP routine in your own computer to control the number of shares it takes from the network.

At this point in my argument, people generally ask why everyone won't always just demand the highest weight possible. The answer to the question involves a trick that gives everyone good reason to use the weights sparingly—a trick I'll get to in a moment. But first, let's check how this scheme ensures the lightning-fast browsing rates I just promised.

The key is to set the weights high for light interactive usage, like surfing the Web, and low for heavy usage, such as movie downloading. Whenever these uses conflict, flows with the higher weighting—those from the light users—will go much faster, which means they will also finish much sooner. Then the heavy flows can expand back to a higher bit rate sooner than otherwise. This is why the heavy flows will hardly take any longer to complete, as shown in Figure 3(3). It's like a supermarket dedicating a few staff to checkouts for customers buying 6 items or less.

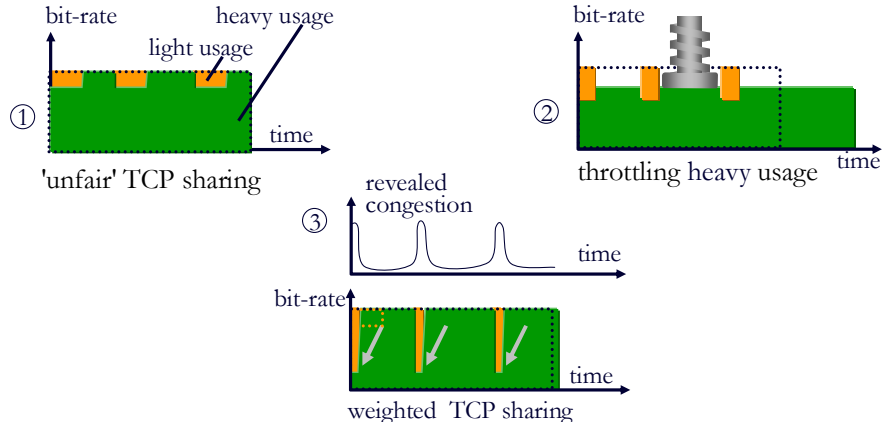

Figure 3—Throttling (2) makes light usage a little faster by slowing down heavy users. Whereas weighted TCP sharing (3) allows light usage to go much faster, hardly affecting heavy users.

But on today's Internet the balance of the weights is the wrong way around. That brings us to the second part of the problem: how can we encourage everyone to flip the weights? This is a 'tragedy of the commons' where one person's share of Internet capacity precludes others benefiting, but they cannot be prevented from consuming the shared goods. It is similar to global warming, where everyone happily pursues what's best for them—leaving lights on, driving a big car—despite the effect this may have on everyone else through the build-up of carbon dioxide and other greenhouse gases.

As with CO_2, the benefit of the Internet to all would greatly improve if we set limits. The metric for those limits should be the effect one customer has on others—the incremental cost of their usage. That isn't how many gigabytes you download but how many you download when everyone else is trying to do the same. Or, more precisely, it's the volume you download weighted by the prevailing level of congestion. This is called your congestion-volume, measured in bytes. Think of it as your carbon footprint for the Internet.

Imagine a world where some Internet service providers offer a deal for a flat price as they do today, but with a monthly congestion-volume allowance. The ISP could continuously feed this allowance into a bucket (Figure 4), which the customer's packets could draw on as required. Note that this allowance doesn't limit downloads as such, it only limits those that persist during congestion. If you used a peer-to-peer program like BitTorrent to download 10 videos continuously, you wouldn't bust your allowance so long as your TCP weight was set low enough. Your downloads would draw back during the brief moments when flows came along with higher weights. But in the end, your video downloads would finish hardly later than they do today.

On the other hand, your Web browser would set its weight high for all its data transfers, because most browsing comes in intense flurries, and so it wouldn't use up much of your allowance. Of course, server farms or heavy users could buy bigger congestion quotas, and light users might get Internet access with a tiny congestion allowance—for a lower flat fee.

But there's a snag. Today, Internet service providers can't set congestion limits because congestion can easily be hidden from them. As we've said, the Internet is designed so that congestion can be detected and managed solely by the computers at the edge—not by Internet service providers in the middle. Certainly, the receiver does send feedback messages about congestion back to the sender, which the network could intercept. But that would just encourage

the receiver to lie or to hide the feedback—no-one has to reveal anything that may be used in evidence against them.

Of course a network provider *does* know about its own congestion—packets it has had to drop itself. But recording each packet dropped by each router against the appropriate customer account would lead to a nightmare of accounting complexity. And, having destroyed the evidence—the dropped packet—it becomes somewhat tricky to hold anyone responsible. Worse, most Internet traffic passes through multiple network providers, and one network cannot reliably detect when another network drops a packet.

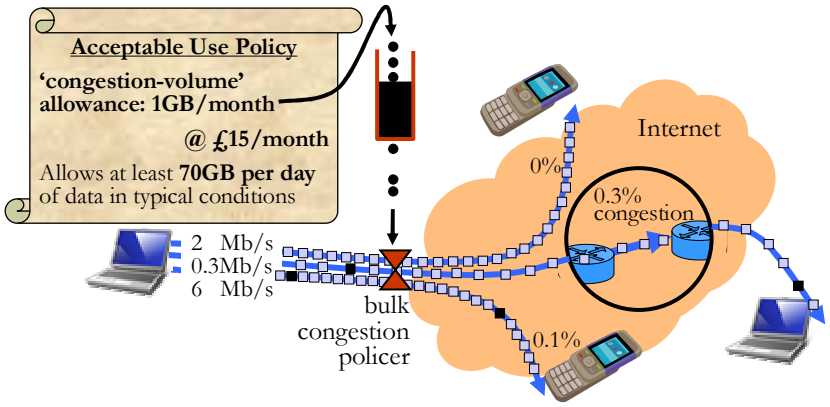

Figure 4—Flat fee congestion policing
If congestion were visible as the proportion of marked packets (shown black), ISPs could encourage software to shift traffic around to everyone's benefit, simply by limiting the overall volume of black packets each customer transferred. Flat fee pricing would suffice and ISPs could remove all other limits on volume or bit-rate. They would not need to identify different flows of packets.

Because Internet service providers can't see congestion-volume, some limit what they can see: the straight volume, in gigabytes, that each customer can transfer in a month. Straight volume is a poor proxy for congestion-volume, because congestion varies rapidly over a wide range. As cost-pressures increase, ISPs will have less wriggle-room to subsidise the costs of some users with the payments of others. Therefore, it will become increasingly important to set limits using the right metric—congestion-volume. Limiting total volume can help to balance things a little, but in times of low congestion it unnecessarily limits everyone. If ISPs could limit congestion-volume rather than volume it would be far better, encouraging extremely fast connections for light usage at little real cost to the heavy users.

Making congestion visible promotes a healthy supply of bandwidth, where market demand most wants it. This should be particularly welcome in the developing world, where everyone wants the efficient way the Internet shares capacity, but no-one wants to invest in expensive capacity if they can't recover the cost from those who take the biggest shares.

Here's how congestion visibility fixes the supply side of the market. If a network is able to see congestion ahead in other networks, it can reroute your data round them. And whenever a network can't reduce congestion ahead by rerouting, it will know that continued congestion represents real demand for more capacity—as customers must be drawing down their limited allowances.

Those networks in most demand will levy a premium against the neighbouring networks that dump congestion-causing traffic into them. But they will have to spend this cash on more capacity to alleviate the congestion. If they don't, but a competing network does, they'll lose the business altogether. Back in the mid-1990s this competitive process was explained by Hal Varian, now Chief Economist at Google. He explained that congestion becomes a commodity with the same price as the cost of the capacity needed to alleviate it.

David Clark, Chief Protocol Architect of the Internet from 1981-89, once famously said, "We figured out how to route bits, but we never figured out how to route money." That's because we never figured out how to reveal which bits were routed towards congestion.

My colleagues and I at BT have figured out a simple way to reveal congestion. We call it 're-feedback'. We believe it's the primary step needed to nudge the Internet onto a much healthier evolutionary path. Once ISPs can encourage their customers' computers to reveal congestion, we expect they will start to enforce congestion limits to bring down costs and

improve quality of customer experience. Then, in turn, their customers will choose software that sets sensible weights for its TCP sharing. Note that only the mechanism for revealing congestion will need to be standardised, not how ISPs might use it. The flat-fee congestion policer discussed earlier (Figure 4) is merely an example of how ISPs might use congestion information.

Here's how re-feedback works. Recall that today the computers at each end of an exchange of packets see congestion, but the networks between them can't. So we built on a technique called Explicit Congestion Notification—the most recent change to the TCP/IP standard, made in 2001. Network equipment that implements that change marks packets as congestion rises rather than doing nothing until it is forced to drop packets. The marks—just a change in a single bit visible to the ISP—let the network see congestion directly, rather than having to dip into packets to look for gaps in the sequence of data, which anyway can be hidden from the network. It's also particularly neat to be able to limit congestion before anyone suffers any real impairment.

Although the 2001 reform reveals congestion, it is only visible downstream of any bottleneck, as packets leave the network. Our re-feedback scheme makes congestion visible at the entrance to the Internet, where it can be limited.

Re-feedback introduces a second type of one-bit packet marking—think of it as a credit and the original congestion markings as a debit. The sender must add sufficient credits to packets entering the network to cover the debit marks introduced as packets squeeze through congested Internet pipes. If any subsequent network node detects insufficient credits relative to debits, it can discard packets from the offending data stream.

To keep out of such trouble, every time the receiver gets a congestion (debit) mark it returns feedback to the sender (Figure 5). Then the sender marks the next packet with a credit. This reinserted feedback (or 're-feedback') can then be used at the entrance to the Internet to limit congestion—you *do* have to reveal everything that may be used in evidence against you.

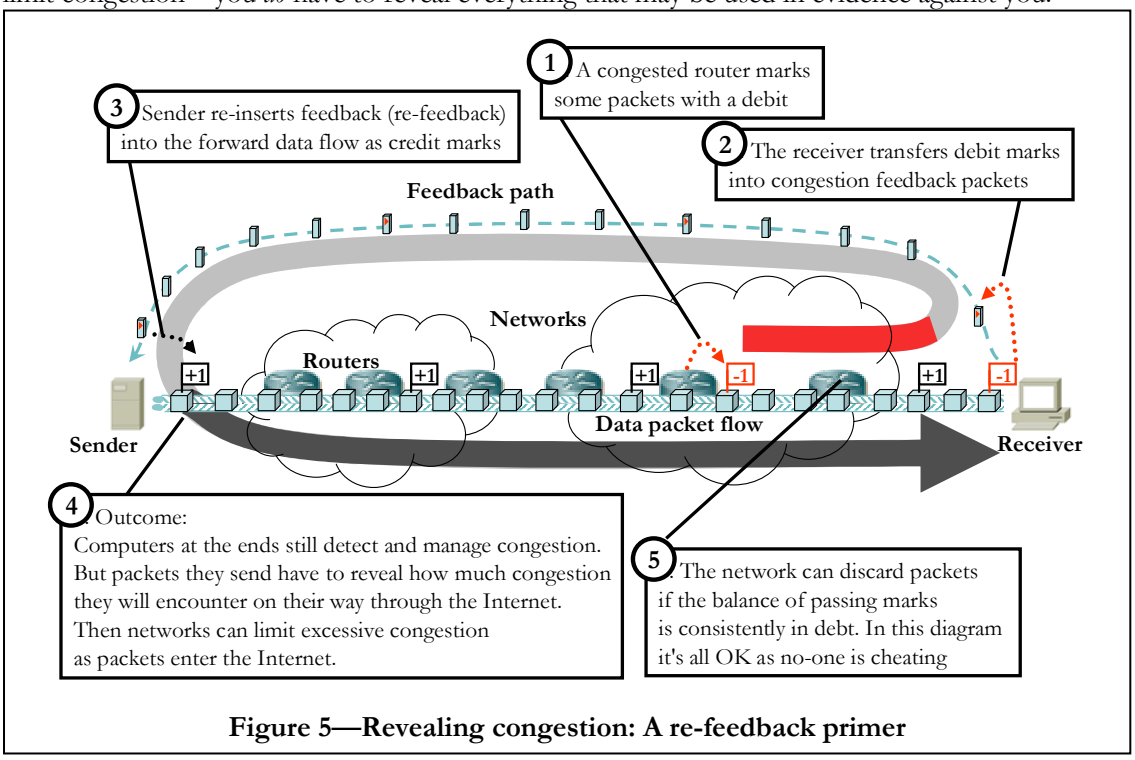

Figure 5—Revealing congestion: A re-feedback primer

Re-feedback sticks to the Internet principle that the computers on the edge of the network detect and manage congestion. But it makes congestion visible to the networks in the middle and enables them to punish anyone who provides misinformation.

The limits and checks on congestion at the borders of the Internet would be trivial for a network operator to add. Beyond that, the re-feedback scheme requires no new code in network equipment; all it needs is for the original standard congestion notification to be turned on. But

packets need somewhere to carry the second mark. That means space needs to be found in the highly constricted preamble at the start of each packet of data—the Internet protocol (IP) header.

In 2005, we prepared a proposal documenting all the technical details of how to make these second marks, and presented it to the Internet Engineering Task Force (IETF), the body that oversees Internet standards. We knew from previous experience that a change to the Internet protocol itself would require bullet-proof justification and, above all, staying power. Three years of argument later, in a non-binding straw poll at the relevant IETF plenary in March 2009, we achieved unanimous consensus that the TCP-friendly sharing regime is not a way forward. Also, at the previous meeting, the IETF had created a research task force design team to propose a new framework for sharing Internet capacity. It is gradually becoming accepted that what matters is not bit rate or volume but *congestion*-volume—the CO_2 of the Internet.

The immediate task is now to build support for a standards working group at the IETF to reveal the Internet's hidden congestion. The chosen mechanism may be re-feedback, but if something better emerges so much the better, as long as the Internet can be as simple and as fast.

The implications of revealing congestion are subtle but deep. For instance, a congestion allowance acts as a trivially simple mechanism to enable differentiated quality of service; because software that wants to go faster can just do so, simply by using up its congestion allowance more rapidly. Therefore an enterprise could simply buy a greater congestion allowance for use on the public Internet instead of buying a virtual private network (VPN) with specially enhanced quality of service. This could work whenever and wherever in the world employees attached to the public Internet, not just over networks explicitly configured to support the enterprise's VPN.

Therefore, the various stakeholders in the Internet will need to start assessing the likely impact of this new capacity sharing approach. That includes companies whose business relies on Internet delivery, and government bodies and public interest groups who have an interest in the Internet as an enabler for wider economic well-being and progress. Before the IETF changes the protocol that has become a cornerstone of the world's economy, it will need to hear broad support from the wider Internet community.

A list of reports, papers and specifications giving more background on Internet capacity sharing and the re-feedback mechanism can be found at
<http://www.cs.ucl.ac.uk/staff/bbriscoe/projects/refb/>
The list identifies the most useful material for different audiences, such as economists, computer networking engineers, security experts and so forth.
