September 2021
Geoff Huston

IAB Workshop on Measuring Network Quality for End Users

The telephone network had a remarkably clear overriding service objective: It had to sustain a human conversation. Now this must be able to carry a signal which is a human voice. To be discernible to human listeners, its necessary to carry audio frequencies of between 300 and 3,500 Hz. Most of the energy in a human voice signal is below 1Khz, and speaking would fit within a dynamic range of 50db from quiet to loud. In terms of perception, we may have 50db of dynamic range, but we find it hard to distinguish between similar amplitude levels and for an intelligible reproduction of the human voice we only need to use somewhere between 200 and 300 discrete amplitude levels. Human perception about what constitutes an interactive conversation tends to have a delay threshold of 300ms. If the delay between the two conversing parties is less than this threshold, they will not speak over each other, whereas higher delays cause us to trip over each other by interrupting because we interpret a pause as a silence gap, not a result of signal propagation delay. Out of this comes a set of parameters for a digitised voice signal, using an 8-bit sample (giving 256 discrete levels) taken every 125us (being able to encode a maximum frequency of 4,000 Hz). A voice signal can therefore be encoded as a digital voice channel of a constant rate of 64Kbps. If a digital network is precise in switching time, such that jitter and drop are minimised, and if the end-to-end delay is held to within 300ms, then such a network would meet the minimum quality levels to be able to carry clearly discernible human conversations. However, switching these synchronous circuits is challenging. The sharing technique used in the telephone network was Time Division Multiplexing, relying on highly stable and tightly synchronised clocks at the sender and receiver. This time switching element was probably the most significant cost point for digital telephone networks.

There are no such basic constraints with computers and packet-switched networks. With adaptation functionality (through transport protocols) in the computers, applications that want to communicate can be highly tolerant of varying rates of network-imposed jitter and loss. They can also tolerate varying path capacity, varying delay, and even various permutations of packet reordering.

Given that these digital packet networks are not built to support a single use, but are usable across a broad range of use profiles, then the question becomes: What constitutes a *good* network? In the telephone world a *good* network supported a human conversation where each party could clearly hear the other and the conversational interchange was within normal human perception parameters. But what is the corresponding set of performance criteria for a general-purpose digital network? Yes, it would be good to support human conversations in a manner where there are no network-induced distortions to the clarity or any impairment to impression of synchronicity in the conversation. Yes, we would like to send extended streams of high-definition video that can be played back without any form of video stutter. At the same time, we would like immediacy of small volume transactions, whether they be small web transactions of the babble arising from an internet that is replete with "things". Oh, and yes, we'd like to shift large volumes of stored data in a timely manner. We'd like applications over the cloud to have the same immediacy as applications running on a local platform. And we'd like gaming applications not to impose a delay penalty on some players. A *good* network should be able to support all of these attributes, and all at the same time.

This performance question has a number of implications, from the area of public regulation of consumer services intended ensure that consumers are well informed about the price and quality of network services through to informing designers of network equipment to ensure that their equipment is *fit for purpose* and for designers of software systems that are intending to use the network's resources efficiently for their application.

Obviously, it also applies to network operators and their use of scaling parameters where they are attempting to match the aggregate sum of service requirements from the network's customers to the capacity of the network's infrastructure. All these activities have been variously grouped under the general heading of *Quality of Service* (QoS).

This topic of QoS has now accumulated a rich history. The basic question is how to design and operate a network that can be used as a common shared substrate for a wide variety of application behaviours. In technology terms some applications are jitter, delay, and loss intolerant and tend to operate best without the intervention of active buffers on the traffic flow, but also have a relatively small traffic rate, while other applications can be relatively tolerant of these behavioural parameters, but they want to act as bandwidth scavengers and soak up all the available network resources that are not placed under contention by less tolerant traffic flows. How do we engineer a shared network that can meet all these service parameters at once?

One approach is crude, but generally effective. Just keep on adding more carriage capacity to the network. If the network is engineered to exceed the most demanding set of performance parameters, then the network is more likely to meet less demanding performance profiles. Of course, such an approach can be needlessly extravagant and wasteful, and in some cases, it simply may not be possible.

Another approach is to constrain the devices and applications that use the network, so that their individual behaviour is sufficiently conservative that the cumulative pressure on the network dies not create an impaired outcome for the application. This form of conservative self-limiting of application behaviour may result in network inefficiencies when the level of self-limitation is way below the network's actual capacity at the time. Such an approach may not really ensure that the network doesn't impair application behaviour in any case. If the sum of these self-constrained demands exceeds the network's capacity, then the result is still an impaired service outcome. So, this approach does not really work all that well either.

This leaves us with the objective of some form of coupling between application behaviour and available network capability. The TCP flow control protocol addresses this coupling requirement by using a feedback loop. By providing an ACK packet for every successfully delivered data packet back to the sender, the TCP sender is able to calculate the current round-trip time (RTT) between sending a data packet and receiving an ACK for this data packet, and the sender is also able to keep a running tally of successfully delivered in-sequence packets to detect some forms of packet loss. The basic loss-based TCP flow control algorithm is simple; for each RTT interval where there has been no packet loss then increase the sending rate, and if you have a RTT interval where there was packet loss then drop the sending rate. It's a good example of loose coupling. The sender is not trying to precisely match the network's capabilities at all times but is trying to find an acceptable equilibrium between using the network efficiently and maintaining an acceptable goodput (i.e., successfully delivered data packets). It can be a remarkably tolerant protocol, tolerating networks with various profiles of jitter, a broad range of variance of RTTs, and some level of tolerance for noisy media. However, there are some applications where TCP looks like the wrong choice, and the best example is human conversations. As long as the packets are small, then a voice application can tolerate a certain level of packet loss, but it is far less tolerant of jitter, and of course it is sensitive to latency (or high RTT conditions). The underlying issue is that voice traffic has a small margin for adaptation, and there is little flexibility to perform some form of coupling with the network's behaviour.

So, what do we do now? The answer includes many decades of work in QoS for the Internet. An early approach called for the application to establish a network path between speaker and receiver and then to inform the network that it desired a certain level of service for its packets long this path (think virtual circuit establishment, but in a packet-switched network). The central element in this Integrated Service architecture was the idea of a "flow state" with some form of reserved resources along a network path which was used for the packet flow associated with the original reservation. There were a few issues with this approach, including scalability and fragility of the network state and the commercial failure of the approach. A second effort was mounted with the Differentiated Services architecture that used a combination of packet marking and network response. It did not intend to offer absolute performance outcomes, but instead attempted to discriminate marked packets

from the usual best effort packet handling by reducing jitter, minimising queuing delay or minimising packet loss rates. Again, this has not gained much traction in the ISP marketplace was there are no service assurances here, just a relative discrimination outcome. From the customer perspective it's hard to justifying paying for an outcome that is very challenging to see and measure! We've seen further refinements to this, including signalling refinements to define the interaction between the application and the network, the use of MPLS to become the QoS universal substrate, then versions permutations of source and segment routing and even aggregated QoS as an amalgam of Intserv and Diffserv. The problem in all these approaches is that QoS cannot fix network over-subscription, poor network design, poor business plans or the basic laws of physics that impose a minimum signal propagation delay between any two points.

But the observation that we have so far failed at various forms of tighter coupling between applications and the network in the QoS space does not mean that we are not interested in the quality of the service provided by the network. Not at all. This is a continuing topic of interest. What defines a "good" access network? What metrics relating to the delivered service would allow us to differentiate between various service offerings? How accurate are the ISPs' claims about the quality of their serve offering? How can we figure out who is causing a bad service experience?

There has also been much in the way of improving the performance of our protocols, using tools such as QUIC, Multipath TCP, NBBR and TCP Fast Open. Yet the network is still plagued with poor performance. One possible explanation is that we've fixated on bandwidth as the only relevant service parameter, and not looked at other parameters, such as queue behaviour and latency underload, the stability of the service, the performance of ancillary functions, such as DNS resolution. It also does not look at the differences in outcomes for short flows and sustained flows. Improving these aspects of network quality will likely depend on measurement and exposing metrics to all involved parties, including to end users in a meaningful way. Such measurements and exposure of the right metrics will allow service providers and network operators to focus on the aspects that impacts the users' experience most and at the same time empowers users to choose the Internet service that will give them the best experience.

This was the topic of a three-day IAB workshop on "Measuring Network Quality for End Users" in September this year, which I attended. There were some 33 contributions to the workshop by researchers, network operators, equipment and service vendors, and application designers, and here I won't even try to track through every contribution. I will take a more general perspective and describe my takeaways from the workshop.

Workshop Themes

One of the workshop themes was to advocate further refinements to support a "more responsive" Internet looks in detail of the elements in the total click to screen delay, which includes DNS resolution time, TCP setup time, TLS negotiation time and 0-RTT session resumption, and TCP data flow setup. Ideally, all data flows should be adaptive and capable of exerting a pressure on the other concurrent data flows such that each flow will find a point of equilibrium with a fair share of available network capacity. Active Queue Management should be used in conjunction with Explicit Congestion Notification in order to improve the coupling between application data flow management and current network path state. It also helps if the source of the data is situation close to the consumer, because a faster and more responsive experience is the product of the number of round-trip times to set up a session to deliver the data to the consumer and the average time taken to perform this round trip.

It is also not overly helpful to consider network performance metrics in isolation from the performance achieved by applications. The example used to advance this proposition is the work on congestion control, which looked at the various abstract models of a highly simplified network (the bottleneck barbell, or the parking lot, for example) and then adding some guesstimate of so-called background traffic using a mix of elastic (adaptable) and inelastic flows and then adding the behaviour under test. While this process can generate repeatable tests in a simulated environment it does not have sufficient levels of direct correlation to the behaviour of applications working over the Internet to be useful as a measurement of the user experience with

any application. To be successful, any quality metric or measurement framework must include consideration of the behaviour of applications that are operating on networks as seen by the user of that application.

This theme was further explored in an assertion of "the futility of QoS", based on the observation that users want a better experience from the network service, or Quality of Experience (QoE), while the network is tuned to deliver a better service profile, or Quality of Service (QoS). The issue here is that the user experience is an outcome of an application operating over a network and different applications will behave in very different ways when working across various network service profiles. For example, a conversational voice application may increase its playback buffer to compensate for network jitter, while a web browsing application is jitter tolerant but delay sensitive. These days the issue of the coupling between network and application behaviour is more complex when there are active network elements. Such active elements include transport level TCP session window manipulators and TCP ACK proxies, as well as application-level proxies such as cache servers used in many Content Distribution Networks (CDNs). The implication is that network provider's QoS service level has little, if any, bearing on the resultant QoE that the application delivers to the end user.

This observation correlates with much of our experience over the past couple of decades where the efforts to bring services based on some distinguished QoS have largely failed in the market. Some networks may attempt to perform preferential queuing or varying drop responses based on some form of packet profiling, but in general these have not gained much traction in the larger market for network services. The user experience also tends to bear this out, when, like this IAB QoS workshop itself, a hundred or so participants can join a worldwide video conference and get clear audio and video without any appreciable degradation of the experience. Gone are the days of multi-second stutters or drop-outs, and most of this has been a result of adequate network dimensioning and good application engineering. So, what's the problem here that would justify the investment in some preferential differential treatment of network traffic? Or in other words, in the eyes of the end user, why is some QoS response is necessary given the functional adequacy of today's tools and today's networks?

I think part of the issue is a fluid flow dynamic problem. The network does not reserve capacity for any given flow. The application's packets sit in the network's buffers alongside the packets from all other concurrently active applications. The application can life its packet sending rate and in so doing it will exert pressure on other applications by increasing their buffer delay and loss probability. The application is counting on being able to establish a new equilibrium point with the other applications that allows it a fair share of the network's resources. One reason why this is a reasonable expectation is the assumption that there is a uniform flow control algorithm used by all these concurrently active applications, and this algorithm is in fact establishing an equilibrium point with itself! From this perspective the network is seen as a largely passive medium, in much the same manner as the role of roads in relation to vehicle traffic flow.

However, this line of thinking may be over-marginalising the role of the network in all this, and the network appears to be engineering to meet the predominate use characteristic, and as this use profile changes then the network's engineering changes. In a time when end-to-end paths have spanned both access and transit network and edge access speeds where relatively low compared with "core" speeds (the early dialup Internet is a good example here) then the compromise here is to maximise throughput overall all else. This can be achieved with large buffers and a loss-based congestion control algorithm with a potential overhang of around two times the bottleneck capacity. But when edge speeds increase and the transit distances collapse, such as is the case with fibre-based access networks and CDN's delivering the majority of traffic, then we can operate networks with far smaller buffers and use something like a BBR flow control algorithm in order to achieve a more precise coupling between the application and the network. If we think about this short-hop Internet where the source is typically so close to the destination that we are shrinking it all down to an extreme case of a 2- or 3-hop Internet, then the internal design of this network necessarily changes, and the application control algorithms necessarily change as well.

There is an interaction between a number of activity domains in changing the overall nature of the Internet and its capability to improve its service delivery capabilities:

- the refinements in the infrastructure of the network by improving its infrastructure in terms of speed, in terms of queue capacity and queue management, and most recently in terms of active marking,
- the refinements in terms of content architectures by adoption of widespread content distribution networks to bring content far closer to users and reduce or even eliminate the dependence on the performance of long-haul Internet paths, and
- the refinements of the behaviour of the application by adjusting its flow control algorithms to perform an efficient coupling with the network and other concurrent flows.

The relative costs of these refinement differ markedly across these three domains. Physical infrastructure changes, such as found in networks, are constrained by capital equipment investment cycles, and these cycles typically span up to a decade or even more. Content infrastructure is also a relatively long-term investment proposition and fundamental changes to this model again span cycles of decades. This means that it is left to the applications, which use a much faster cycle of software upgrades to optimise their behaviour against the slower-changing infrastructure. Financially, these are not vertically integrated markets, and each of these sectors operate independently from each other. What this implies is that making changes to this compound system is challenging and while we might all want a ubiquitous high speed digital services for all customers, it is challenging to orchestrate such changes to the infrastructure of this system and equally challenging to orchestrate how the customer income, that ultimately fuels this system, is distributed through the system in a way that motivates all these sectors to work in concert to achieve this service outcome. It is perhaps even more challenging when you add the condition that there is no central agent here that is empowered to orchestrate this environment, and we are relying on a common theme in economics that efficient outcomes are achieved when individual motivations are aligned with a common beneficial outcome.

In the QoS part of the Internet that's an exceptionally tough ask! However, we have reached a number of hypotheses which may have already been well established in other domains. The first is that investment in rationing the consumption of a resource is a lot less efficient than using the same investment sum to increase the volume of the common resource. This is a common theme over the past couple of decades in the QoS world where the simple answer to quality issues is to "add more bandwidth for all" in preference to complex mechanisms that attempt to selectively ration out the existing pool of insufficient bandwidth. Secondly, it's an exceptionally fine balance between under- and over-selling marginal improvements in network response. A congested express lane is still a congested lane and over-subscribing an elevated service results in the same degraded response as an over-subscribed baseline service. And if a network is undersubscribed then there is no marginal improvement to be had were one to subscribe to an elevated quality service.

Whether this leads to QoS being an "illusion of choice" or whether the applications are being pitted against each other in a "zero-sum game" (where a marginal improvement in the service level achieved by one application only comes at the marginal decrease in the service level achieved by other applications) is open to debate, but the basic premise is that QoS does not and cannot create more network resources in the first place.

Fairness among applications has an odd dynamic. The BitTorrent framework consistently achieves superior throughout by exploiting parallelism. The data blocks are widely distributed and transferred across potentially diverse network paths in parallel. If fairness between transport sessions is based on an equilibration between concurrent flows, then if one application commands more than one flow to conduct the data transfer then under such a framework then it can receive a disproportionate amount of network resources. However, BitTorrent has not been shifted into the mainstream of technology adoption, even in the application space of content distribution. Perhaps BitTorrent is an example of transforming fairness into areas which are unfair to other concurrent non-BitTorrent data flows. A similar reticence exists with multi-path TCP, where a data transfer can be split into two or more streams between the same source and destination, with each stream managed as a separate congestion-controlled flow.

The topic of the quality of the experience of the end user has implications in the regulatory space when the Internet is seen as a consumer product. Are consumers able to make informed choices relating to the competing services on offer? In many markets the competitive differentiation is based on some notion of speed and

possibly data caps, but it's clear that the useability of the service also includes some less obvious attributes, such as jitter, load characteristics, buffering and traffic shaping.

It is impractical to expect each consumer to conduct such tests for themselves, and there is a regulatory role to ensure that consumers are informed of the results of fair, impartial, comparable and independent measurements of the user experience from the various retail service offerings. In a number of environments, the local regulator has stepped in with a measurement program that fulfils this role. For example, in the US the FCC undertook the program of Measuring Broadband America in conjunction with SamKnows, which allows consumers (and service providers) to compare their service offerings. The metrics collected by such programs tend to look at speed, delay, and loss, but these are attributes of the service, rather than reliable indications of the user's quality of experience when using the service. Here, there is little to help the regulator in terms of adopted industry standards for such measurements.

Of course, it's not just a framework of what to measure and how to measure it, but an understanding of what is being measured. If we want to shift focus from the quality of the service being offered by the access network to the consumer's premises to the quality of the user's experience on a connected device, then the internal detail of the home network comes into play. The most commonly observed issues in this space concerns the conditions within the home network. The quality of home WiFi connections vary considerably. There may be a high local noise level in the selected WiFi band, a high codec delay in the WiFi equipment. There may be other active applications at the time of a quality measurement that impinges on the results. The result is often that there is a difference between the quality of the service as delivered to the premises and the quality of the experience as experienced by an application running on a user's device.

Can an ISP sell a retail service with associated Quality of Experience parameters in an environment that is "defendable" as a fair and honest representation of the service outcomes, and one where the consumer is aware of what they are buying?

Part of the issue with quality measurements is to understand the distinction between the baseline of a service and exceptional events. Is this my access service that is the problem? Or is it applications that I may be running at the time, or exceptional events at the other end of the application's connections. It makes some sense to obtain a set of baseline metrics using long term passive monitoring and associating these measurements with a signature of the running applications at the time. At times of recorded changes in packet loss rates or latency it would be helpful to understand what applications were active and the way in which they influenced the overall user experience.

Measurements and Metrics

There have been some recent developments in the area of "responsiveness" metrics, which are intended to complement existing measurements of bandwidth and ping latency. This responsiveness metric can be as simple as looking at passive collection of transport headers from TCP sessions and measuring the time taken by TCP for a "round trip" between sender and receiver. Moving from passive to active measurement would entail some TCP-based monitor application that would performs its own time measurement of this "round trip transaction time" at the application layer. The metric being proposed here is a simple inversion of the round-trip time, into a value terms "RPMs", or round-trips per minute. Obviously the higher this number the better.

There are some interesting longitudinal measurements using the Google Measurement Lab data showing the monthly average RPM values spanning more than a decade for a set of the larger retail ISPS in the US, showing a steady improvement in this responsiveness metric over the period.

There is an interesting metric that looks at mean responsiveness by TCP flow algorithm (Figure 1). It illustrates the observation that such average measurements make sense in the context of the transport protocols that form the dominant component of the ecosystem. If you change the protocol that you may need to alter the measurement or threshold value of the metric to reflect the properties of the changed protocol.

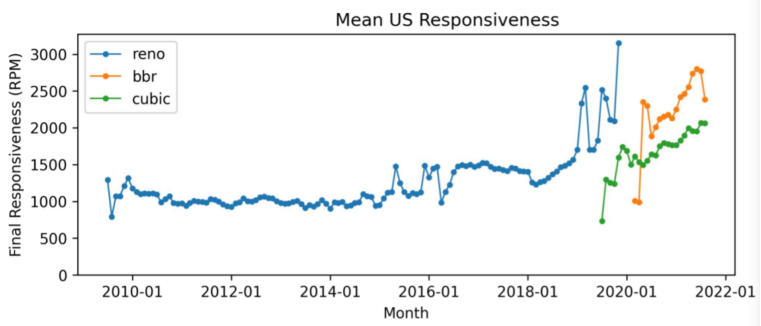

Figure 1 – Average US Responsiveness – Measurement Lab Data, Matt Mathis

There is a strong correlation between this responsiveness measure and page load times, as it has been observed that most web transactions run out of data while the transport protocol is still in slow start mode. For such small content packages, the limiting performance factor is the number of round-trip intervals, not the peak data rate that is available once the transport protocol comes out of slow start probing. For such transactions the dominant metric that is indicative of the user experience is the number of round-trip times, or the RPM metric. This insight leads to a different approach to measurement, and rather than relay on the ICMP packet exchange, as is the case with *ping*, it is feasible to push this measurement into an application content and embed HTTP/2 echo frames into the HTTP protocol exchange between client and a measurement server, which more closely aligns to the RPM metric.

However, the treatment of the initial window size in TCP on the server side appears to vary considerably. The measurement exercise saw a significant number of clients with 5 seconds of queued data after only 10 seconds elapsed time in the session, which appears to be indicative of a large initial window size compounded by slow start's window doubling per RTT behaviour. This may well have a bearing on the related issues of performance degradation due to buffer bloat.

This leads to a view of a metric of a *network propagation delay* which is the sum of transmission, switching and coding delays, plus the factor of persistent network queues. This leads to a metric of the unit of *responsiveness* which is the unit of the loop between two communicating applications which encompasses the sum of the transport protocol delay (which itself is largely dominated by network propagation delay) and application delay. If we take this responsiveness metric and add loss and congestion control signals, then we can arrive at a *goodput* metric, which expresses the effective data transfer rate in in-sequence acknowledged data arriving at the remote end of a connection. The overall intent here is not to look at loss and retransmission as isolated metrics, but to look at the impact of these behaviours on the achievable data transfer rate.

These metrics are still *network-centric*. The user experience is somewhat different and includes human perception factors. Access speeds head into the area of diminishing returns as they increase. The perceived difference between a 1G and a 5G access service, for example would be largely imperceptible in all but the most demanding of customer environments, and this is perhaps part of the explanation that in terms of speed alone there is little in the way of visible discrimination between 4G and 5G mobile services. As speeds increase and average round trip times decrease the impacts of loss and jitter also head into a largely imperceptible realm, as the application is easily capable of absorbing such variations in the network service. So should we look at the experience behaviours that are visible to the customer. In a video streaming app can the streamer maintain high resolution without freezing? Will the application react to skipping without a very visible pause? Some user experience measurements show a similar speed reading across a range of access ISPs but show a visible variation in streaming performance. Similarly, the experience for game applications, where low latency and low jitter are important, show marked variation even though there may be relatively similar speed measurements. In many cases this may be the result of differences in the access technology, such as the differing characteristics between mobile Internet systems and wires systems, but in some cases the differences are seen when a common access medium is used by a set of retail ISPs, where the differences lie not in the access network but in the ISP's back-end network.

When the traffic profiles are examined using a high speed programable network chip there are significant differences in the behaviour of the application which, in turn, exercise various aspects of the network. Figure 2 shows a trace of the "Netflix Pulse" where the streamer refills the playback buffer in discrete pulses every few seconds (many video streamers use a similar pulsing technique – it is not just Netflix), while the second is a trace of the multiplayer first person shooter game Counter-Strike: Global Offensive, which uses a relative steady traffic flow. Queueing behaviour could absorb the Netflix pulse without much visible impact, queuing behaviour in a steady flow situation could cause standing queues, as there is no quiescent period that would allow a queue to drain, causing the queue to act as a delay buffer.

Figure 2 – Application Pulse Profiles: Vijay Sivaraman

What happens to the service profile when there is a mix of traffic that places load on the access network? The anticipated scenario is that the impressed load will fill the buffers in the access network and the delays to pass through this standing queue will add latency. Active Queue Management (AQM) is one approach to mitigate this issue, and in one large US retail network A/B comparison measurement data reveals significantly better upstream latency under load performance when AQM is used. In this experiment most of the Latency Under Load (LUL) tests resulted in 15-30 milliseconds of latency when the access device, a DOCSIS modem, was configured to use AQM. In comparison, the device without AQM averaged roughly 250 milliseconds of latency, between 8-16 times higher, a highly significant difference to the end user quality of experience. These large-scale measurement comparisons should provide additional data to justify accelerated deployment of AQM in DOCSIS and other Internet Service Provider networks and user-purchased home network equipment. In general, the LUL values will be higher in conjunction with lower access speeds, as LUL likely related to the ratio of the size of the buffer to the speed of the access service.

What is evident here is that improvements in queue management in the edge access devices (Consumer Premises Equipment, or CPE) can have dramatic impacts on the delivered quality of experience. Where those devices are provisioned and managed by the ISP, then it is possible to make significant improvements in the overall quality of the service delivered to the customer without undertaking extensive upgrades to the internal network's feeder infrastructure, as evidenced by the AQM exercise. On the other hand, where the CPE is the responsibility of the end user then a poor choice of CPE may have a significant impact on the perception of the quality of the ISP's service without the ISP being able to correct the issue. The issue of the quality of CPE is a perennial issue in the ISP retail space, plagued by poor implementations of DNS stub resolution functionality, extremely poor NAT performance, poor IPv6 performance, poor WiFi implementations and poor internal queue management. This is coupled with poor user documentation which can lead the end user to make configuration changes that further exacerbate poor performance rather than improve it. Trying to lift the both the performance of the access networks and the performance of the device platforms and applications

in order to compensate for these CPE issues seems to be taking the long way around the basic problem of poorly implemented CPE!

The topic about poor handling of WiFi being an issue with CPE implementations also invokes the broader topic of WiFi and the quality of WiFi connections in general. Measurement studies report a consistent gwoth in the volume of traffic delivered over a WiFi last hop, doubling in both the 2.4G and 5G bands every three years in Europe and North America. At the same time the speeds achieved over WiFi are consistently slower than broadband more than half the time in the 2.4G band and a quarter of the time in the 5G band. This leads to the interesting conclusion that the WiFi spectrum is under pressure, with one study reporting that spectrum need in WiFi grows by some 25% annually, yet the spectrum allocation remains fixed. This topic is tangential to this particular workshop in a sense, but it's a topic of the tensions between the licensed and unlicensed radio spectrum allocations will continue to absorb the attention of the spectrum regulators for the foreseeable future.

Signalling and Coupling

The issue of QoS responses in a network is bound up in signalling by applications. The thinking here is that an application signals to the network that it has some inelastic service requirements and the network presumably has the choice to accept or deny this request. If accepted then presumably it is up to the application to ensure that it's behaviour remains within its signalled service requirements, and presumably it's an option for the network to police this service profile against the applications actual traffic. If this looks like a pretty accurate representation of the old voice telephone network than its probably not a coincidence. It this also looks to have all the inefficiencies of the old voice telephone network than that too would not be a coincidence. There was an effort in the IETF to improve service signalling at one point, and the NSIS (Next Steps in Signalling) Working Group was chartered by the IETF in 2001 to work on a replacement for the RSVP protocol that explicitly included a QoS signalling protocol component. Many documents were produced, including 18 RFCs, over the ensuing 10 years. It is unclear to what extent this work influenced subsequent thinking about the need for, and precise role of, such forms of signal interaction between applications and networks, but the observation that the topic continues to be raised at workshops such as this one probably indicates that NSIS work did not constitute the final word on this topic!

So, it appears that we have a number of elements in the network stack that are all part of the framework to improve the delivered outcomes for the end user. For many years we have worked in a "loosely coupled" environment where each element worked in isolation of the others. Network designers and vendors of network equipment made design trade-offs in their networks that optimised what they perceived to be the more critical or most valuable use profile. In optimising network utilisation to support large scale data transfer most effectively in a bandwidth constrained environment then large buffers (or at buffers designed to have a comparable size of the delay bandwidth product of the link that they were driving) appeared to make sense from the network operator's perspective. From the perspective of the end host platform's transport protocol design the task was to optimise the flow control algorithm so that it stabilised around steady rate inflation, hence the widespread adoption of *cubic* as the preferred flow control application in many platforms. And applications simply had to put up with the outcomes. Improvements in this model were designed to augment the coupling between network and transport, hence the ECN approach which allowed the transport to receive explicit notification of congestion-based queue formation in advance of queue saturation and packet loss. At the same time the application environment has been breaking out of this model and using an approach of folding the transport protocol into the application itself. The combination of QUIC and BBR is a good example of this approach where the application is attempting to optimize its use of the network to the point of onset of queue formation within the network path, thereby avoiding the need for ECN and similar signalling given that the BBR protocol is attempting to avoid this condition in the first place. Hence the discussion of whether the use of QUIC with its encrypted transport controls hampers the network's efforts to assist efficient transport, or whether it's simply not that a useful additional signal of the network path state in the context of BBR-like protocols.

What is the "appropriate" signal of network state? And is this a question that is independent of the choice of transport protocols? For example, when one major application changed its behaviour to use a combination of

QUIC and BBR within the application itself, then the observed average rate of packet loss experienced by the application increased, which would normally be seen as a signal of service degradation. However, the BBR flow control algorithm performs loss recovery without collapsing the sending window, so their experience was that goodput increased for their users at the same time. A measurement of path packet loss in isolation of other metrics is not all that useful. If there is a large standing queue build-up then loss is more impactful to the service than isolated loss events, for example.

Where to From Here?

As we increase bandwidth of network links and increase the path diversity within the network, we are increasing its capacity, and this means that the technique of periodic time-based average measurements of loss, capacity and jitter loose much of their relevance. A case can be made to reduce the time base for such quality measurements and move to very fine-grained packet measurements, using tools such as P4 in the Tofino chip set. Such tools can provide clearer insights of application performance across network paths, but it is still early days for the analysis of such data and we have yet to develop an analysis toolset that can be applied more generally outside of individual case studies. It's fine to ask for agreed standard metrics, but it does appear that we are overly impatient to develop such standardised metrics before spending further time to explore the space to understand the interactions between application behaviours, end-to-end protocol behaviours and the parameters of network behaviour.

This points to the elusive nature of QoS in the public Internet. It's been the subject of bursts of intense study for more than three decades, and the collective response has remained relatively unchanged over this period. The Type of Service field in the IPv4 packet header was defined in RFC791 in 1981 with a proposed simple binary preference vector relating to the attributes of delay, throughput and loss. We progressively refined this in various ways with Intserv and Diffserv frameworks representing two of the more comprehensive approaches, but in the public Internet they have not managed to break though the barriers to widespread deployment. The public Internet remains in a state of deployment of best effort networks with a preference for over-provisioning of both bandwidth and buffers over a preference for deploying any form of QoS mechanisms. Perhaps it's a problem of a lack of clear incentives for deployment of QoS mechanisms in networks and a lack of incentives for platforms and applications to adjust their behaviours to mark their traffic with clear signals. The two groups of actors are coupled in a mutually dependent relationship, as any movement by one without the other makes little sense. Early moves by one party, either to equip their platform or application with QoS signalling mechanisms, or by networks to respond to QoS signalling with some distinguished service response, to force the other to shift have not been overly successful in the past, and the future prospects for QoS deployment appear to be largely unchanged.

It appears that part of the problem in this space is that the incentives to deploy QoS/QoE innovations across networks, platforms and applications are rarely, if ever, commonly aligned.

Network providers were looking for a framework that would permit them to sell some of their capacity at a premium price that related to some premium service parameters. The problem is that for most of the time their networks are not overcommitted and when the network is not overcommitted it is possible to provide the same service response to all traffic. Using an analogy of roads, an express lane is only useful at those times when the other lanes are congested. So QoS mechanisms tend to have very limited windows of effectiveness in networks. Of course, when the network is heavily overcommitted then the premium service is also at risk, and to maintain a consistent service response it needs to deny service requests to service more traffic when the premium service is under contention.

Having the network manage resource allocation for its users was one of the reasons why the telephone network was inefficient and expensive. Pushing the resource allocation function back to the applications using the network to reach their own point of meta-equilibrium relieves the network of a very significant cost element and has a significant impact on overall network efficiency. This means that each individual service can be provided at less cost, giving such networks a competitive edge over their resource-managed competitors.

If platform and applications can mark their traffic for premium handling from the network with no cost premium that the end user must pay leads directly to the same outcomes as if the option to perform such traffic marking was not there in any case. If an application can mark its traffic for low delay, low jitter and high resilience with no downside, financial or otherwise, then there is no reason why it would not mark all of its traffic for such handling. And if every application has the same response, then the net result is all traffic is handled the same way and we are back to the starting condition without QoS features in the first place. If the end user has a payment option, then how can they be assured that they are getting value for money. How can they detect that the resultant experience is "better" than the hypothetical case of not paying for this premium experience? And how can they quantify this experience differential? Would paying twice as much yield a doubled goodput? Doubled against what base benchmark? One of the reasons why commuters justify a decision to pay a toll to enter an express lane is that they can visibly see the difference between their express lane and "normal" traffic. But if the user cannot see the alternative, then how can the user justify the additional expenditure?

It's not all hopeless here, and a number of initiatives have made a huge difference to the Internet experience.

- Firstly, the most significant difference comes from the enthusiastic adoption of Content Distribution Networks for content and service publication. Pushing the "remote" end of the network transaction closer to the location of the user removes delay, increases the accuracy of the congestion control algorithm, and increases the efficiency of the use of the access network. Applications are vastly more "responsive". It could be argued that the adoption of CDNs has been so enthusiastic that the Internet these days is no more than the aggregate union of a small collection of CDN services, and the entire component of transit is now largely an historical and commercially unimportant aspect of any ISP's operation.

- Secondly, we've made significant progress in understanding server-side flow control algorithms. Changing the flow algorithm from the rather brutal and coarse signal of packet drop to view that is driven by the observation of the onset of standing queue formation has had a major impact on the expectations on service quality for data-flow applications. BBR is, for me, a truly significant innovation in this space and its enthusiastic adoption by many application servers and CDN platforms attests to the significance of this technology.

- Thirdly we are slowly getting around to the view that internal buffer provisioning in network devices show not continue to follow the bandwidth delay product rule of thumb. High performance networks don't buffer to any significant degree and eliminating this expensive fast memory from the network's infrastructure further reduces the cost associated with high-speed network infrastructure. Better performance at lower cost. Win / Win, as they say.

I must admit to coming out of this workshop with a renewed appreciation that further improvements in the quality of the Internet experience is not going to be achieved by selective rationing of premium treatment of traffic for a few at the expense of the whole. I don't think the problem is a lack of a rich signalling mechanism between the state of internal element of a network path and the flow control points located in the devices and the network's edge. I don't think we need a QoS architecture to raise the quality of the delivered service on the network. It's still the case that more bandwidth for all users helps every part of the Internet experience. Bringing content and services closer to its users and thereby reducing delay makes perhaps the most significant change to the users' perceptions of a responsive and capable Internet experience. Using better flow control algorithms help with this. Oddly enough, more network buffering doesn't necessarily help!

After three days of an intensive dive onto the current state of understanding of QoS and QoE in this IAB workshop I have to admit that for me the conclusion was that the effort to understand the components of what contributes to an effective user experience is useful work, but the development of further signalling mechanisms between network and application, or further refinements to a range of network response, which common components of a QoS framework, are still not looking all that promising. It's a conclusion that was evident

decades ago when we asked ourselves why the IPv4 TOS bits were not being used, and its much the same conclusion I have today!

The Workshop's presentations and papers have been assembled online at https://www.iab.org/activities/workshops/network-quality/. I would like to thank the IAB and the workshop co-chairs, Wes Hardaker, Eugeny Khorov, and Omer Shapira for inviting me to participate in the workshop and to all the workshop contributors for such a richly diverse and stimulating set of papers and conversations about this fascinating topic!

Disclaimer

The above views do not necessarily represent the views or positions of the Asia Pacific Network Information Centre.

Author

Geoff Huston AM, B.Sc., M.Sc., is the Chief Scientist at APNIC, the Regional Internet Registry serving the Asia Pacific region.

www.potaroo.net