November 2022
Geoff Huston

Some Random Notes from IETF 115

The IETF held its 115th meeting in London in November 2022. This was another in the set of hybrid meetings with specific support for online attendees in addition to the normal face-to-face meetings for the week. In no particular order, here are a few of my impressions from the IETF meeting.

More QUIC

It took some time, but a few years back the IETF published a stable specification of Multipath TCP. This extension to TCP allowed a host to use multiple local interfaces (and associated network paths) to communicate with a remote host simultaneously over both paths. Specifically, this mechanism could support improved resilience in certain mobile environments when a device could use both WiFi and cellular radio connections simultaneously. This multipath concept is being imported into QUIC to enable the simultaneous utilization of different paths to exchange QUIC frames for a single connection. The chosen mechanism to support this is similar to QUIC's path agility, where an active QUIC connection can migrate to a new path, but instead of closing off the old path, Multipath QUIC keeps both paths active. The major design decision in this approach is to use per-path packet numbers to simplify QUIC's use of the loss recovery and congestion control mechanisms on a per-path basis.

The QUIC Working Group is also looking at the load balancing issue for QUIC. Conventionally, a network recognises packets as part of a single stream if the packets share a common 5-tuple (protocol number, source and destination addresses and source and destination ports). QUIC path agility allows clients to change their IP address while maintaining the connection state. If there are load balancers on the path this may disrupt the session as a conventional load balancer is unaware of QUIC's path agility. Preferably, QUIC-aware load balancers should be able to correctly follow QUIC's path agility and ensure that the load balancing forwarding state remains constant in spite of this form of address migration. The draft specification for QUIC Load Balancing (https://datatracker.ietf.org/doc/html/draft-ietf-quic-load-balancers) proposes using the connection ID in place of the conventional use of the 5-tuple. It provides common algorithms for encoding the server mapping in a connection ID given some shared parameters. The mapping is generally only discoverable by observers that have the parameters, preserving unlinkability by unrelated third parties as much as possible. While the working draft describes a small set of configuration parameters to make the server mapping intelligible, the means of distributing these parameters between load balancers, servers, and other trusted intermediaries is out of its scope. It strikes me that this missing item is a key part of making QUIC Load Balancing generally useful, as this approach needs to draw the load balancers into the same protected domain as the servers, which may be challenging to define in a suitably generalized manner.

QUIC Measurement and Analysis

Hypergiant Deployment via QUIC Backscatter Analysis

QUIC is an odd protocol to measure. Because Chrome implements a QUIC detection mechanism that enables QUIC on second use, many Chrome clients do not switch to use QUIC because they simply do not access the same site again in that critical time interval interval which is outside HTTP/2 session reuse and inside the HTTP/3 capability local cache lifetime. So QUIC remains a protocol that is far greater in potential use than actual observed use.

In any case, QUIC is a UDP-based protocol, and suffers from the same UDP attack profile as, say, the DNS. An attacker can generate a QUIC initial packet using a spoofed source address which will trigger a QUIC server to respond to that address with an initial response as part of the QUIC handshake. While the response is not greater in size as compared to the initial packet, it will be repeated after a retransmission timeout, so there is a small amplification gain if QUIC is used as an attack vector in a source spoofed DDOS attack.

This might sound like a rather esoteric form of DOS attack on the server, but such 'orphan' initial QUIC connection packets are common enough for a research group to listen for backscatter QUIC packets on a large (/9) IPv4 passive listening space (https://datatracker.ietf.org/meeting/115/materials/slides-115-maprg-on-the-opportunities-of-passive-measurements-to-understand-quic-deployments). In this case they are looking at the retransmission timers used by the major QUIC service platforms (Google, Facebook and Cloudflare). Facebook appears to be the most persistent in terms of retransmit, sending between 3 to 9 further packets, spaced at exponentially increasing intervals. Google appear to use a similar exponential backoff strategy, with between three to four retries, and Cloudflare use just two retries, spaced one second apart.

In looking at the responding server connection ID the group observed that Facebook and Cloudflare both used structured Connection ID values, identifying the responding host and process ID in the server connection ID value. The likely reason for this structured field is to allow the server front end load balancer to use the server's host ID rather than the client IP address and Port, as in QUIC a client can migrate to a new connection while preserving the QUIC connection (there is a draft on this approach, draft-ietf-quic-load-balancers). From this point they use active scanning to send QUIC packets to all host ID values, collecting the server connections and host ID. There are some 37K different Host IDs in Facebook's deployment, and the approach reveals some 122 distinct server clusters, each using a /24 IPv4 address.

It's a neat piece of analysis in attempting to reverse engineer Facebook's server network from the clues buried in the server connection ID. But neat as it is, I must admit I find myself asking "so what?"

Evaluating DNS over QUIC

Measuring the DNS for performance is challenging. Firstly, what are we trying to measure? If it's the stub resolver's wall clock time to resolve a name, then the results will vary greatly depending on whether the query name is held in a recursive resolver cache or not. Then there is the issue of the number of labels in the query name, and again whether the name servers themselves are already cached or if they have to be dynamically discovered. And then there is the question of whether the query name is DNSSEC-signed and whether the recursive resolver is performing DNSSEC validation. And finally, there is the issue of how to instrument the measurement client. Is it possible to capture and measure the elapsed wall clock time from the initial DNS query to a usable response? Or must the client use an indirect measurement such as the wall clock time for the first web response?

So, measuring the DNS is a challenge, but even so I found this particular measurement exercise even more challenging to interpret (https://datatracker.ietf.org/meeting/115/materials/slides-115-maprg-dns-privacy-with-speed-evaluating-dns-over-quic-and-its-impact-on-web-performance). Now they could've set up their own DNS recursive resolver that supported a number of DNS encapsulation protocols (UDP/TCP, HTTP/2, HTTP/3, TLS AND QUIC) and explicitly loading the local cache with the DNS response and then get the client set to use each of the DNS transport protocols to resolve the name, preferably with multiple queries to see if the stub resolver was taking advantage of session reuse. Instead, the measurement exercise used recursive resolvers that they discovered by scanning the IPv4 Internet and scripted as collection of Amazon EC2 clients to perform the tests. They did not appear to look at DNS resolution times per se, but used an elapsed time that included the DNS resolution time and some parts of the time for the web transaction. Because I was not sure of what their measurement was trying to achieve, I'm not sure what to make of the results they presented. Yes web load for DNS over UDP seemed to be slightly faster than DNS over QUIC, which itself is slightly faster than DNS over HTTPS (DoH), but there are so many unconstrained variables in this measurement that I find it difficult to ascribe all that much significance to this result.

If you are looking for a good analysis of the performance of DNS over HTTPS/3 (i.e., DNS over HTTP framing with QUIC transport) relative to DNS over TLS over TCP (DoT) then you might find the analysis in this report (https://security.googleblog.com/2022/07/dns-over-http3-in-android.html) far more helpful.

DNS Topics

For a while there the DNS was at the centre of a lot of activity in the IETF.

There was work on trying to make DNS servers more capable of fending off DOS attacks, work on adding privacy to the DNS, and the continual effort of refining DNSSEC support. It appears to me that the intensity of this DNS innovation activity has not been sustainable in the long run, and the efforts in DNSOP, DPRIVE and other DNS-related areas of IETF activity have recently headed into filling in the details rather than furthering the larger fundamental topics of the role of the DNS and the manner of name resolution. This DNS work is still a significant agenda, as there is a massive amount of detail to work through, but we are now seeing efforts in defining nomenclature for the DNS, defining some of the more esoteric details of DNSSEC validation, adding detail and colour to DNS error reporting and similar topics.

The topic of alternate (non-DNS) name spaces and its intersection with the public DNS name space continues to vex the IETF. The earlier efforts to corral a number of 'reserved' top level domains in a 'special use' name registry (RFC 6761) excited some level of contention in the IETF when the IETF added the top level label `.onion` to the special use names registry. While the level of support for undertaking another round of grappling with this issue is by no means universal, the working group is looking at a slightly different approach this time around. This proposal (https://datatracker.ietf.org/doc/draft-ietf-dnsop-alt-tld/) is for the IETF to reserve the `.alt` top level label for use as an unmanaged pseudo-TLD namespace. The `.alt` label can be used in any domain name as a pseudo-TLD to signify that this is an alternative (non-DNS) namespace, and should not be looked up in a DNS context. Personally, I'm not convinced if it makes the entire issue of managing colliding name spaces between DNS and non-DNS name spaces any easier or harder!

DNSSEC is also a topic that just never goes away. The operational complexities of synchronising information between the parent and child across a delegation boundary continue to provoke various forms of solutions that attempt to automate much of the process, where the solutions in turn lead to further issues of operational complexity. For me the elephant in the DNSSEC room is that so few stub resolvers perform DNSSEC validation in any case. The end client of the entire DNS infrastructure is still left in the position of forced trust in the functions of others and has no means to directly validate that the resolution outcomes that they are present with are indeed authentic. It's a rather unsatisfying situation.

Trouble at the BGP Mill

Prognostications of the imminent, or perhaps not-so-imminent, demise of the inter domain routing system in the form of BGP data back to the first days of BGP. Back in 1991 or so the IETF ROAD group was formed, where "ROAD" stood for Routing and Addressing, to look at the dire predictions of the exhaustion of class B IPv4 addresses and the imminent demise of BGP. At the time the answer for BGP was the switch to BGP-4 and dropping implicit Address Class assumptions in the routing protocol.

And in a 1998 IAB workshop we heard predictions that the rate of expansion of the BGP routing table was exceeding the router vendors' ability to keep pace. Out of this workshop came the tunnelling protocol LISP, which has gone down much the same ill-fated path as all other tunnelling protocols.

This time Pawel Foremski reminded us (https://datatracker.ietf.org/meeting/115/materials/slides-115-maprg-kirin-hitting-the-internet-with-millions-of-distributed-ipv6-announcements) that the BGP forwarding table in the public Internet is approaching another milestone size of 1M IPv4 entries. It is currently at some 970,000 entries according to RouteViews (https://bgp.potaroo.net/as6447/). These forwarding tables, or FIBs, are stored on

the unit's line cards, and need to have a lookup performance that is of the same order as the packet processing rate of the unit. This means that these FIBS are typically stored on high-speed content-addressable memory or highly customized ASICs. Either way they are very pricey, and as the routing table grows the require size for these memory units continues to grow.

In theory 1M FIB entries is a milestone point where a collection of deployed routers may encounter FIB overflow issues, but if that's truly the case then we passed this milestone point over a year ago. The thing is that most of the core of the Internet is dual stack already, and a FIB implementation needs to carry some 960,000 4-byte entries and a further 180,000 16-byte entries for the IPv6 routing table. That's a grand total of 1.7M 32-bit entries. So, if a FIB size of 1M was a threshold point for some parts of the default-free core of the Internet, then we passed this point some time ago.

How big can this FIB get? We need to make some assumptions to answer this., In IPv4 it's generally observed that operators filter out prefixes that are smaller than a /24. Today some 60% of the routed IPv4 prefixes are a /24. What if the ongoing aftermarket in IPv4 address prefixes continued to fragment the IPv4 prefix space so that every prefix was a /24? There would be some 17M such prefixes, which is a number that is far in excess of the capacity of the routers in today's default-free core.

But if you think that's not so good, then spare a second to think about IPv6. Some 50% of the advertised prefixes in the IPv6 routing table are a /48 (currently we see some 80,000 /48 prefix advertisements). A further 2% of prefixes are a /56, and there are just under 1% of prefixes that are a /64 (https://bgp.potaroo.net/v6/as6447/). What would happen to the routers' FIBs if a network with a /32 IPv6 address allocation decided to advertise its entire address prefix as a collection of /48s? Well that's not so bad, as there are only some 65,000 /48's in a /32. What if they decided to announce /56's or even /64's? Now that coule become a real problem, as a /32 contains some 4B /64s! Now "that will never happen!" is an odd claim on the Internet. Just look at AS7713 (https://www.cidr-report.org/cgi-bin/as-report?as=AS7713&view=2.0&v=6) to seer the counterfactual, where a single /32 is sliced and diced into 489 announcements, 306 of which are /48's and 138 are /64's.

While there is general consensus among network operators that a /24 as the minimum advertised prefix size in IPv4, no such common consensus exists in IPv6, and the scale of IPv6 is such that any individual address allocation can be sliced up into a set of more specific prefix advertisements that will overwhelm today's BGP routing infrastructure. Maybe we should make a token effort and look at /48s as a minimum advertised prefix size in IPv6. Maybe we should take route aggregation more seriously and filter out the widespread practice of advertising more specifics IPv6. Or maybe we should completely ignore this vulnerability in the routed environment and pretend that it will never be exploited, accidently or deliberately!

Starlink Performance

When you look at the performance of adaptive transport protocols, such as TCP, one of the most critical factors is distance between the two parties. Strictly speaking, it's not the distance per se, but the amount of time it takes for packets to pass from the sender to the receiver and back. Because adaptive protocols rely on some form of feedback from the receiver to the sender, the longer the delay between the two parties the harder it is for the protocol to optimise its performance and adapt to the characteristics of the network because the feedback signal is lagging in time.

"Traditional" satellite services were a classic example of a high delay path. Geostationary satellites orbit at a distance of 32,786km from the earth's equator, and 42,644km from the poles. A typical round trip time for a geostationary satellite service was 650ms, far higher than the 30ms to 160ms experienced in terrestrial systems. However, with the launching of a new generation of low earth orbiting spacecraft from SpaceX and WebOne, the satellite situation has changed dramatically. These LEO spacecraft orbit at an altitude of 500km - 1,200km, and the round trip time for signal propagation from the surface to the spacecraft and back is between 7 and 15ms. This should have a dramatic impact on protocol performance when using these LEO services.

The work (https://datatracker.ietf.org/meeting/115/materials/slides-115-maprg-a-first-look-at-starlink-performance) used a simple analysis looking at the total page load time for the top 120 web sites using a terrestrial service, a geo-stationary satellite service and the Starlink LEO service. Starlink performed in a manner that was very similar to the terrestrial service, which was significantly faster than the geo-stationary satellite service. Their latency measurements show a 50ms median delay, with a variance of +/-10ms. This latency extended when the service was placed under load, showing some characteristics of overly generous queues on the network path. The loss characteristics were generally in short bursts rather than extended loss events. The overall performance was of a comparable level to a terrestrial service. I would've liked to see a more detailed analysis of small scale jitter in the service, as well as an analysis of buffer behaviour, and how this relates to the performance of loss-based and delay-bounded congestion control algorithms

Congestion Control Research

The area of adaptive flow control algorithms has been an active area of research, with the work on delay-bounded congestion control algorithms now competing directly with the traditional loss-based congestion control approaches. These delay-bound algorithms, such as Vegas, FAST, BBR and PCP are variously triggered by queuing delay, the receive rate and adaptive learning. All these approaches lead over time to delay convergence where the delay variation of individual packets is small, which is exactly what the algorithm is intending to achieve. However, this state is also often associated with starvation, where one flow will take the majority of network resources and starves all other concurrent flows. Work at MIT's CSAIL (https://datatracker.ietf.org/meeting/115/materials/slides-115-iccrg-starvation-in-end-to-end-congestion-control) looks at non-congestion delay variation due to traffic clustering in WiFi controllers, mobile network or even Ethernet.

They found that different delay-bound flows will make different estimates of delay because they cannot generally distinguish between congestive delay and non-congestive delay. The implication is that even a small amount of non-congestive delay in a path can lead to starvation conditions.

Does this mean that delay-bound algorithms are always going to be flawed in terms of risk of starvation? Possibly. Could we mitigate this risk? Possibly. Potential mitigations include adding sensitivity to ECN signals., or perhaps deliberately oscillating the delay estimate. The larger lesson is that we still have much to learn about the relationship between network behaviours, queues and protocol behaviours, and the entire area of congestion control remains a rich area of study.

IETF 115 Proceedings

This is just a very small selection of material from a rather full week of activity. The full agenda of the IETF 115 meeting, together with the presented material and session recordings can be found at https://datatracker.ietf.org/meeting/115/agenda/.

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