QUIC
Internet-Draft
Intended status: Informational
Expires: November 15, 2018

B. Trammell, Ed.
P. De Vaere
ETH Zurich
R. Even
Huawei
G. Fioccola
Telecom Italia
T. Fossati
Nokia
M. Ihlar
Ericsson
A. Morton
AT&T Labs
E. Stephan
Orange

May 14, 2018

Adding Explicit Passive Measurability of Two-Way Latency to the QUIC Transport Protocol
draft-trammell-quic-spin-03

Abstract

This document describes the addition of a "spin bit", intended for explicit measurability of end-to-end RTT, to the QUIC transport protocol. It proposes a detailed mechanism for the spin bit, as well as an additional mechanism, called the valid edge counter, to increase the fidelity of the latency signal in less than ideal network conditions. It describes how to use the latency spin signal to measure end-to-end latency, discusses corner cases and their workarounds in the measurement, describes experimental evaluation of the mechanism done to date, and examines the utility and privacy implications of the spin bit.

time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on November 15, 2018.

Table of Contents

1. Introduction

The QUIC transport protocol [QUIC-TRANS] is a UDP-encapsulated protocol integrated with Transport Layer Security (TLS) [TLS] to encrypt most of its protocol internals, beyond those handshake packets needed to establish or resume a TLS session, and information required to reassemble QUIC streams (the packet number) and to route QUIC packets to the correct machine in a load-balancing situation (the connection ID). In contrast to TCP, QUIC's wire image (see [WIRE-IMAGE]) exposes much less information about transport protocol state than TCP's wire image. Specifically, the fact that sequence and acknowledgement numbers and timestamps (available in TCP) cannot be seen by on-path observers in QUIC means that passive TCP loss and latency measurement techniques that rely on this information (e.g. [CACM-TCP], [TMA-QOF]) cannot be easily ported to work with QUIC.

This document proposes a solution to this problem by adding a "latency spin bit" to the QUIC short header. This bit is designed solely for explicit passive measurability of the protocol. It provides one RTT sample per RTT to passive observers of QUIC traffic. This document describes the mechanism, how it can be added to QUIC, and how it can be used by passive measurement facilities to generate RTT samples. It explores potential corner cases and shortcomings of the mechanism, and proposes an extention called the Valid Edge Counter (VEC) to mitigate them. It further details findings on privacy risk researched by the QUIC RTT Design Team, which was tasked by the IETF QUIC Working Group to determine the risk/utility tradeoff for the spin bit.

Appendices summarize experimental results to date with an implementation of the spin bit built atop a recent QUIC implementation, describe use cases for passive RTT measurement at the resolution provided by the spin bit, explore alternatives to the spin bit for passive latency measurement of QUIC flows, and discuss the necessity of "greasing" the spin bit.

The spin bit has low overhead, presents negligible privacy risk, and has clear utility in providing passive RTT measurability of QUIC that is far superior to QUIC's measurability without the spin bit, and equivalent to or better than TCP passive measurability.

1.1. About This Document

[QUIC-SPIN-EXP] specifies the addition of the spin bit to the QUIC transport protocol for experimental purposes. This document provides background for that specification, documents work done in the development of the spin bit proposal, and extends it with the VEC signal for loss, reordering, and delay compensation without relying on the QUIC packet number.

This document is maintained in the GitHub repository https://github.com/britram/draft-trammell-quic-spin, and the editor's copy is available online at https://britram.github.io/draft-trammell-quic-spin. Current open issues on the document can be seen at https://github.com/britram/draft-trammell-quic-spin/issues. Comments and suggestions on this document can be made by filing an issue there, or by contacting the editor.

2. The Spin Bit Mechanism

The latency spin bit enables latency monitoring from observation points on the network path. Each endpoint, client and server, maintains a spin value, 0 or 1, for each QUIC connection, and sets the spin bit on packets it sends for that connection to the appropriate value (below). It also maintains the highest packet number seen from its peer on the connection. The value is then determined at each endpoint as follows:

- The server initializes its spin value to 0. When it receives a packet from the client, if that packet has a short header and if it increments the highest packet number seen by the server from the client, it sets the spin value to the spin bit in the received packet.

- The client initializes its spin value to 0. When it receives a packet from the server, if the packet has a short header and if it increments the highest packet number seen by the client from the server, it sets the spin value to the opposite of the spin bit in the received packet.

This procedure will cause the spin bit to change value in each direction once per round trip. Observation points can estimate the network latency by observing these changes in the latency spin bit, as described in Section 3. See Section 3.2 for an illustration of this mechanism in action.

The defails of the addition of the spin bit to the QUIC short header are given in [QUIC-SPIN-EXP].

3. Using the Spin Bit for Passive RTT Measurement

When a QUIC flow is sending at full rate (i.e., neither application nor flow control limited), the latency spin bit in each direction changes value once per round-trip time (RTT). An on-path observer can observe the time difference between edges in the spin bit signal in a single direction to measure one sample of end-to-end RTT. Note that this measurement, as with passive RTT measurement for TCP, includes any transport protocol delay (e.g., delayed sending of acknowledgements) and/or application layer delay (e.g., waiting for a request to complete). It therefore provides devices on path a good instantaneous estimate of the RTT as experienced by the application. A simple linear smoothing or moving minimum filter can be applied to the stream of RTT information to get a more stable estimate.

An on-path observer that can see traffic in both directions (from client to server and from server to client) can also use the spin bit to measure "upstream" and "downstream" component RTT; i.e, the component of the end-to-end RTT attributable to the paths between the observer and the server and the observer and the client, respectively. It does this by measuring the delay between a spin edge observed in the upstream direction and that observed in the downstream direction, and vice versa.

3.1. Limitations and Workarounds

Application-limited and flow-control-limited senders can have application and transport layer delay, respectively, that are much greater than network RTT. Therefore, the spin bit provides network latency information only when the sender is neither application nor flow control limited. When the sender is application-limited by periodic application traffic, where that period is longer than the RTT, measuring the spin bit provides information about the application period, not the RTT. Simple heuristics based on the observed data rate per flow or changes in the RTT series can be used to reject bad RTT samples due to application or flow control limitation.

Since the spin bit logic at each endpoint considers only samples on packets that advance the largest packet number seen, signal generation itself is resistant to reordering. However, reordering can cause problems at an observer by causing spurious edge detection and therefore low RTT estimates, if reordering occurs across a spin bit flip in the stream. This can be probabilistically mitigated by the observer also tracking the low-order bits of the packet number, and rejecting edges that appear out-of-order [RFC4737].

All of these limitations are addressed by an enhancement to the spin bit, the Valid Edge Counter, described in detail in Section 4.

3.2. Illustration

To illustrate the operation of the spin bit, we consider a simplified model of a single path between client and server as a queue with slots for five packets, and assume that both client and server sent packets at a constant rate. If each packet moves one slot in the queue per clock tick, note that this network has a RTT of 10 ticks.

Initially, during connection establishment, no packets with a spin bit are in flight, as shown in Figure 1.

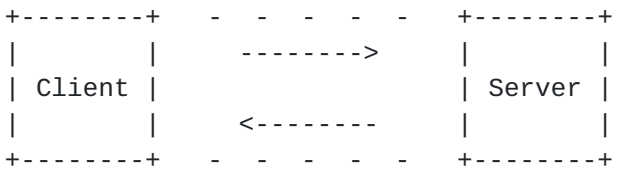

Figure 1: Initial state, no spin bit between client and server

Either the server, the client, or both can begin sending packets with short headers after connection establishment, as shown in Figure 2; here, no spin edges are yet in transit.

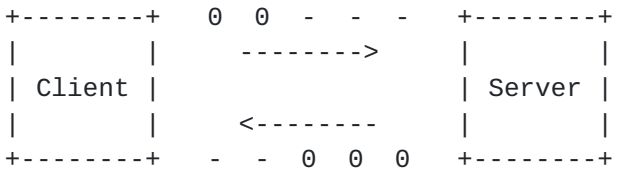

Figure 2: Client and server begin sending packets with spin 0

Once the server's first 0-marked packet arrives at the client, the client sets its spin value to 1, and begins sending packets with the spin bit set, as shown in Figure 3. The spin edge is now in transit toward the server.

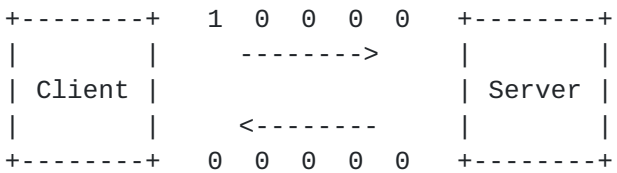

Figure 3: The bit begins spinning

Five ticks later, this packet arrives at the server, which takes its spin value from it and reflects that value back on the next packet it sends, as shown in Figure 4. The spin edge is now in transit toward the client.

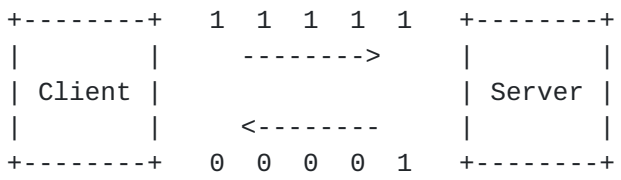

Figure 4: Server reflects the spin edge

Five ticks later, the 1-marked packet arrives at the client, which inverts its spin value and sends the inverted value on the next packet it sends, as shown in Figure 5.

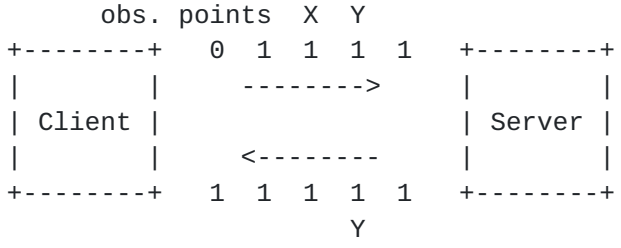

Figure 5: Client inverts the spin edge

Here we can also see how measurement works. An observer watching the signal at single observation point X in Figure 5 will see an edge every 10 ticks, i.e. once per RTT. An observer watching the signal at a symmetric observation point Y in Figure 5 will see a server-client edge 4 ticks after the client-server edge, and a client-server edge 6 ticks after the server-client edge, allowing it to compute component RTT.

Figure 6 shows how this mechanism works in the presence of reordering. Here, packet C carries the spin edge, and packet B is reordered on the way to the client. In this case, the client will begin sending spin 1 after the arrival of C, and ignore the spin bit flip to 1 on packet B, since B < C; i.e. it does not increment the highest packet number seen.

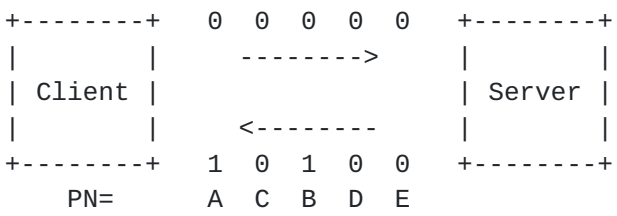

Figure 6: Handling reordering

4. The Valid Edge Counter

This mechanism is indented to provide additional information about the validity of the passively observed spin edges without using information from a cleartext packet number.

A one-bit spin signal is resistent to reordering during signal generation, since the spin value is only updated at each endpoint on a packet that advances the packet counter. However, without using the packet number, a passive observer can neither detect reordered nor lost edges, and it must use heuristics to reject delayed edges.

The Valid Edge Counter (VEC) addresses these issues with two additional bits added to each packet, encoding values from 0 to 3, indicating that an edge was considered to be valid when send out by the sender, and providing a possibility to detect invalid edges due to reordering and edge loss.

4.1. Proposed Short Header Format Including Spin Bit and VEC

As of the current editor's version of [QUIC-TRANS], this proposal specifies using bit 0x04 of the first octet in the short header for the spin bit, and the bits 0x03 for the valid edge counter. Note that these values are subject to change as the layout of the first octet is finalized.

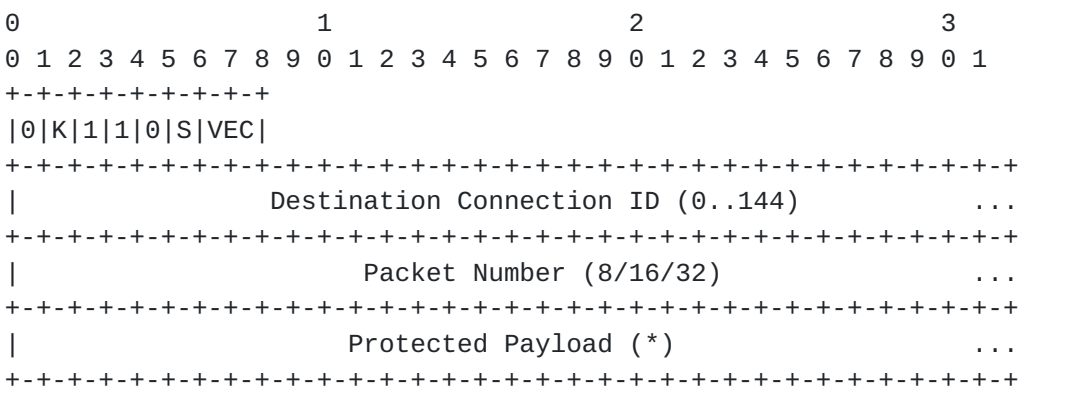

Figure 7: Short Header Format Spin Bit and VEC

S: The Spin bit is set 0 or 1 depending on the stored spin value that is updated on packet reception as explained in Section 2.

VEC: The Valid Edge Counter is set as defined in Section 4.2. If the spin bit field does not contain an edge, the VEC is set to 0.

4.2. Setting the Valid Edge Counter (VEC)

The VEC is set by each endpoint as follows; unlike the spin bit, note that there is no difference between client and server handling of the VEC:

- By default, the VEC is set to 0.

- If a packet contains an edge (transition 0->1 or 1->0) in the spin signal, and that edge is delayed (sent more than a configured delay since the edge was received, defaulting to 1ms), the VEC is set to 1.

- If a packet contains an edge in the spin signal, and that edge is not delayed, the VEC is set to the value of the VEC that accompanied the last incoming spin bit transition plus one. This counter holds at 3, instead of cycling around. In other words, an edge received with a VEC of 0 will be reflected as an edge with a VEC of 1; with a VEC of 1 as VEC of 2, and a VEC of 2 or 3 as a VEC of 3.

This mechanism allows observers to recognize spurious edges due to reordering and delayed edges due to loss, since these packets will have been sent with VEC 0: they were not edges when they were sent. In addition, it allows senders to signal that a valid edge was delayed because the sender was application-limited: these edges are

sent with the VEC set to 1 by the sender, prompting the VEC to count back up over the next RTT.

4.3. Use of the VEC by a passive observer

The VEC can be used by observers to determine whether an edge in the spin bit signal is valid or not, as follows:

- A packet containing an apparent edge in the spin signal with a VEC of 0 is not a valid edge, but may be have been caused by reordering or loss, or was marked as delayed by the sender. It should therefore be ignored.

- A packet containing an apparent edge in the spin signal with a VEC of 1 can be used as a left edge (i.e., to start measuring an RTT sample), but not as a right edge (i.e., to take an RTT sample since the last edge).

- A packet containing an apparent edge in the spin signal with a VEC of 2 can be used as a left edge, but not as a right edge. If the observation point is symmetric (i.e, it can see both upstream and downstream packets in the flow), the packet can also be used to take a component RTT sample on the segment of the path between the observation point and the direction in which the previous VEC 1 edge was seen.

- A packet containing an apparent edge in the spin signal with a VEC of 3 can be used as a left edge or right edge, and can be used to compute component RTT in either direction.

5. Privacy and Security Considerations

The privacy considerations for the latency spin bit are essentially the same as those for passive RTT measurement in general.

A concern was raised during the discussion of this feature within the QUIC working group and the QUIC RTT Design Team that high-resolution RTT information might be usable for geolocation. However, an evaluation based on RTT samples taken over 13,780 paths in the Internet from RIPE Atlas anchoring measurements [TRILAT] shows that the magnitude and uncertainty of RTT data limit the resolution of geolocation information that can be derived from Internet RTT to national- or continental-scale; i.e., less resolution than is generally available from free, open IP geolocation databases.

One reason for the inaccuracy of geolocation from network RTT is that Internet backbone transmission facilities do not follow the great-circle path between major nodes. Instead, major geographic features

and the efficiency of connecting adjacent major cities both influence the facility routing. An evaluation of ~3500 measurements on a mesh of 25 backbone nodes in the continental United States shows that 85% had RTT to great-circle error of 3ms or more, making location within US State boundaries ambiguous [CONUS].

Therefore, in the general case, when an endpoint's IP address is known, RTT information provides negligible additional information.

RTT information may be used to infer the occupancy of queues along a path; indeed, this is part of its utility for performance measurement and diagnostics. When a link on a given path has excessive buffering (on the order of hundreds of milliseconds or more), such that the difference in delay between an empty queue and a full queue dwarfs normal variance and RTT along the path, RTT variance during the lifetime of a flow can be used to infer the presence of traffic on the bottleneck link. In practice, however, this is not a concern for passive measurement of congestion-controlled traffic, since any observer in a situation to observe RTT passively need not infer the presence of the traffic, as it can observe it directly.

In addition, since RTT information contains application as well as network delay, patterns in RTT variance from minimum, and therefore application delay, can be used to infer or fingerprint application-layer behavior. However, as with the case above, this is not a concern with passive measurement, since the packet size and interarrival time sequence, which is also directly observable, carries more information than RTT variance sequence.

We therefore conclude that the high-resolution, per-flow exposure of RTT for passive measurement as provided by the spin bit poses negligible marginal risk to privacy.

As shown in Section 2, the spin bit can be implemented separately from the rest of the mechanisms of the QUIC transport protocol, as it requires no access to any state other than that observable in the QUIC packet header itself. We recommend that implementations take advantage of this property, to reduce the risk that errors in the implementation could leak private transport protocol state through the spin bit.

Since the spin bit is disconnected from transport mechanics, a QUIC endpoint implementing the spin bit that has a model of the actual network RTT and a target RTT to expose can "lie" about its spin bit transitions, by anticipating or delaying observed transitions, even without coordination with and the collusion of the other endpoint. This is not the case with TCP, which requires coordination and collusion to expose false information via its sequence and

acknowledgment numbers and its timestamp option. When passive measurement is used for purposes where one endpoint might gain a material advantage by representing a false RTT, e.g. SLA verification or enforcement of telecommunications regulations, this situation raises a question about the trustworthiness of spin bit RTT measurements.

This issue must be appreciated by users of spin bit information, but mitigation is simple, as QUIC implementations designed to lie about RTT through spin bit modification can easily be detected. A lying server can be contacted by an honest client under the control of a verifying party, and the client's RTT estimate compared with the spin-bit exposed estimate. Though in the general case, it is impossible to verify explicit path signals with two complicit endpoints (see [WIRE-IMAGE]), a lying server/client pair may be subject to dynamic analysis along paths with known RTTs. We consider the ease of verification of lying in situations where this would be prohibited by regulation or contract, combined with the consequences of violation of said regulation or contract, to be a sufficient incentive in the general case not to do it.

6. Acknowledgments

Many thanks to Christian Huitema, who originally proposed the spin bit as pull request 609 on [QUIC-TRANS]. Thanks to Tobias Buehler for feedback on the draft, and for Alexandre Ferrieux for input on the Valid Edge Counter. Special thanks to the QUIC RTT Design Team for discussions leading especially to the privacy and security considerations section.

This work is partially supported by the European Commission under Horizon 2020 grant agreement no. 688421 Measurement and Architecture for a Middleboxed Internet (MAMI), and by the Swiss State Secretariat for Education, Research, and Innovation under contract no. 15.0268. This support does not imply endorsement.

7. References

7.1. Normative References

[QUIC-SPIN-EXP]
Trammell, B. and M. Kuehlewind, "The QUIC Latency Spin Bit", draft-ietf-quic-spin-exp (work in progress).

7.2. Informative References

[ALT-MARK]
Fioccola, G., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate Marking method for passive and hybrid performance monitoring", draft-ietf-ippm-alt-mark-14 (work in progress), December 2017.

[CACM-TCP]
Strowes, S., "Passively Measuring TCP Round-Trip Times (in Communications of the ACM)", October 2013.

[CARRA-RTT]
Carra, D., Avrachenkov, K., Alouf, S., Blanc, A., Nain, P., and G. Post, "Passive Online RTT Estimation for Flow-Aware Routers Using One-Way Traffic (NETWORKING 2010, LNCS 6091, pp. 109-121)", 2010.

[CONUS] Morton, A., "Comparison of Backbone Node RTT and Great Circle Distances (https://github.com/acmacm/CONUS-RTT)", September 2017.

[IMC-CONGESTION]
Luckie, M., Dhamdhere, A., Clark, D., Huffaker, B., and k. claffy, "Challenges in Inferring Internet Interdomain Congestion (in Proc. ACM IMC 2014)", November 2014.

[IMC-TCPSIG]
Sundaresan, S., Dhamdhere, A., Allman, M., and . k claffy, "TCP Congestion Signatures (in Proc. ACM IMC 2017)", n.d..

[MINQ] Rescorla, E., "MINQ, a simple Go implementation of QUIC (https://github.com/ekr/minq)", November 2017.

[MOKUMOKUREN]
Trammell, B., "Mokumokuren, a lightweight flow meter using gopacket (https://github.com/britram/mokumokuren)", November 2017.

[NOSPIN] Morton, A., "Description of a tool chain to evaluate Unidirectional Passive RTT measurement (and results) (https://github.com/acmacm/PassiveRTT)", October 2017.

[QUIC-MGT]
Kuehlewind, M. and B. Trammell, "Manageability of the QUIC Transport Protocol", draft-ietf-quic-manageability-01 (work in progress), October 2017.

[QUIC-TRANS]
Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport-11 (work in progress), April 2018.

[RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <https://www.rfc-editor.org/info/rfc792>.

[RFC4433] Kulkarni, M., Patel, A., and K. Leung, "Mobile IPv4 Dynamic Home Agent (HA) Assignment", RFC 4433, DOI 10.17487/RFC4433, March 2006, <https://www.rfc-editor.org/info/rfc4433>.

[RFC4737] Morton, A., Ciavattone, L., Ramachandran, G., Shalunov, S., and J. Perser, "Packet Reordering Metrics", RFC 4737, DOI 10.17487/RFC4737, November 2006, <https://www.rfc-editor.org/info/rfc4737>.

[RFC5357] Hedayat, K., Krzanowski, R., Morton, A., Yum, K., and J. Babiarz, "A Two-Way Active Measurement Protocol (TWAMP)", RFC 5357, DOI 10.17487/RFC5357, October 2008, <https://www.rfc-editor.org/info/rfc5357>.

[RFC6049] Morton, A. and E. Stephan, "Spatial Composition of Metrics", RFC 6049, DOI 10.17487/RFC6049, January 2011, <https://www.rfc-editor.org/info/rfc6049>.

[RFC7799] Morton, A., "Active and Passive Metrics and Methods (with Hybrid Types In-Between)", RFC 7799, DOI 10.17487/RFC7799, May 2016, <https://www.rfc-editor.org/info/rfc7799>.

[SHBAIR] Shbair, W., Cholez, T., Francois, J., and I. Chrisment, "A multi-level framework to identify HTTPS services (in Proc. IEEE/IFIP NOMS)", April 2016.

[SPINBIT-REPORT]
De Vaere, P., "Latency Spinbit Implementation Experience (https://devae.re/f/eth/quic/spinbit_report/)", November 2017.

[TLS] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", draft-ietf-tls-tls13-28 (work in progress), March 2018.

[TMA-QOF] Trammell, B., Gugelmann, D., and N. Brownlee, "Inline Data Integrity Signals for Passive Measurement (in Proc. TMA 2014)", April 2014.

[TOKYO-PING]
Pelsser, C., Cittadini, L., Vissicchio, S., and R. Bush, "From Paris to Tokyo - On the Suitability of ping to Measure Latency (In Proc. ACM IMC 2014)", October 2014.

[TRILAT] Trammell, B., "On the Suitability of RTT Measurements for Geolocation (https://github.com/britram/trilateration/blob/paper-rev-1/paper.ipynb)", August 2017.

[WIRE-IMAGE]
Trammell, B. and M. Kuehlewind, "The Wire Image of a Network Protocol", draft-trammell-wire-image-04 (work in progress), April 2018.

[WWMM-BLOAT]
Alfredsson, S., Giudice, G., Garcia, J., Brunstrom, A., Cicco, L., and S. Mascolo, "Impact of TCP Congestion Control on Bufferbloat in Cellular Networks (in Proc. IEEE WoWMoM 2013)", June 2013.

Appendix A. Experimental Evaluation

We have evaluated the effectiveness of the spin bit in an emulated network environment. The spin bit was added to a fork of [MINQ], using the mechanism described in Section 2, but with the spin bit appearing in a measurement byte added to the header for passive measurability experiments. Spin bit measurement support was added to [MOKUMOKUREN]. Full results of these ongoing experiments are available online in [SPINBIT-REPORT], but we summarize our findings here.

First, we confirm that the spin bit works as advertised: it provides one useful RTT sample per RTT to any passive observer of the flow. This sample tracks each sender's local instantaneous estimate of RTT as well as the expected RTT (i.e., defined by the emulation) fairly well. One surprising implication of this is that the spin bit provides _more_ information than is available by local estimation to an endpoint which is mostly receiving data frames and sending mainly ACKs, and as such can also be useful in purely endpoint-local observations of the RTT evolution during the flow. The spin bit also works correctly under moderate to heavy packet loss and jitter.

Second, we confirm that the spin bit can be easily implemented without requiring deep integration into a QUIC implementation. Indeed, it could be implemented completely independently, as a shim, aside from the requirement that the spin bit value be integrity-protected along with the rest of the QUIC header.

Third, we performed experiments focused on the intermittent-sender problem described in Section 3.1. We confirm that the spin bit does not provide useful RTT samples after the handshake when packets are only sent intermittently. Simple heuristics can be used to recognize this situation, however, and to reject these RTT samples. We also find that a simple sender-side heuristic can be used to determine whether a sample will be useful. If a sender sends a packet more than a specified delay (e.g. 1ms) after the last packet received by the client, it knows that any latency spin observation of that packet will be invalid. If a second "spin valid" bit were available, the sender could then mark that packet "spin invalid". Our experiments show that this simple heuristic and spin validity bit are successful in marking all packets whose RTT samples should be rejected.

Fourth, we performed experiments focused on the reordering problem described in Section 3.1. We find that while reordering can cause spurious samples at a naive observer, two simple approaches can be used to reject spurious RTT samples due to reordering. First, a two-bit spin signal that always advances in a single direction (e.g. 00 -> 01 -> 10 -> 11) successfully rejects all reordered samples, including under amounts of reordering that render the transport itself mostly useless. However, adding a bit is not necessary: having the observer keep the least significant bits of the packet number, and rejecting samples from packets that reverse the sequence [RFC4737], as suggested in Section 3.1, is essentially as successful as a two-bit spin signal in mitigating the effects of reordering on RTT measurement.

Fifth, we performed parallel active measurements using ping, as described in Appendix C.2. In our emulated network, the ICMP packets and the QUIC packets traverse the same links with the same treatment, and share queues at each link, which mitigates most of the issues with ping. We find that while ping works as expected in measuring end-to-end RTT, it does not track the sender's estimate of RTT, and as such does not measure the RTT experienced by the application layer as well as the spin bit does.

In summary, our experiments show that the spin bit is suitable for purpose, can be implemented with minimal disruption, and that most of the identified problems can be easily mitigated. See [SPINBIT-REPORT] for more.

Appendix B. Use Cases for Passive RTT Measurement

This section describes use cases for passive RTT measurement. Most of these are currently achieved with TCP, i.e., the matching of packets based on sequence and acknowledgment numbers, or timestamps and timestamp echoes, in order to generate upstream and downstream

RTT samples which can be added to get end-to-end RTT. These use cases could be achieved with QUIC by replacing sequence/ acknowledgement and timestamp analysis with spin bit analysis, as described in Section 3.

In any case, the measurement methodology follows one of a few basic variants:

- The RTT evolution of a flow or a set of flows can be compared to baseline or expected RTT measurements for flows with the same characteristics in order to detect or localize latency issues in a specific network.

- The RTT evolution of a single flow can also be examined in detail to diagnose performance issues with that flow.

- The spin bit can be used to generate a large number of samples of RTT for a flow aggregate (e.g., all flows between two given networks) without regard to temporal evolution of the RTT, in order to examine the distribution of RTTs for a group of flows that should have similar RTT (e.g., because they should share the same path(s)).

B.1. Inter-domain Troubleshooting

Network access providers are often the first point of contact by their customers when network problems impact the performance of bandwidth-intensive and latency-sensitive applications such as video, regardless of whether the root cause lies within the access provider's network, the service provider's network, on the Internet paths between them, or within the customer's own network.

The network performance is currently measured by points of presence on-the-path which extract spatial delay and loss metrics measurements [RFC6049] from fields of the transport layer (e.g. TCP) or of application layer (e.g. RTP). The information is captured in the upper layer because neither the IP header nor the UDP layer includes fields allowing the measurement of upstream and downstream delay and loss.

Local network performance problems are detected with monitoring tools which observe the variation of upstream metrics and downstream metrics.

Inter-domain troubleshooting relies on the same metrics but is not a pro-active task. It is a recursive process which hones in on the domain and link responsible for the failure. In practice, inter-domain troubleshooting is a communication process between the Network

Operations Center (NOC) teams of the networks on the path, because the root cause of a problem is rarely located on a single network, and requires cooperation and exchange of data between the NOCs.

One example is the troubleshooting performance degradation resulting from a change of routing policy on one side of the path which increases the burden on a defective line card of a device located somewhere on the path. The card's misbehavior introduces an abnormal reordered packets only in the traffic exchanged at line rate.

Other examples are similar in terms of cooperation requirements and the need to refer to measurements. NOCs need to share the same measurement metrics and to measure these metrics on the same fields of the packet to enable a minimal level of technical cooperation.

Experimentation with the spinbit Appendix A has shown ability to replace the current RTT measurement opportunities based on clear-text transport or application header fields with a standard approach for measuring passive upstream and downstream RTT, which are a fundamental metric for this diagnostic process.

B.2. Two-Point Intradomain Measurement

The spin bit is also useful as a basic signal for instantaneous measurement of the treatment of QUIC traffic within a single network. Though the primary design goal of the spin bit signal is to enable single-observer on-path measurement of end-to-end RTT, the spin bit can also be used by two cooperating observers with access to traffic flowing in the same direction as an alternate marking signal, as described in [ALT-MARK]. The only difference from alternate marking with a generated signal is that the size of the alternation will change with the flight size each RTT. However, these changes do not affect the applicability of the method that works for each marking batch separately applied between two measurement points on the same direction. This two point measurement is an additional feature enabled "for free" by the spin bit signal.

So, with more than one observer on the same direction, it can be useful to segment the RTT and deduce the contribution to the RTT of the portion of the network between two on-path observers. This can be easily performed by calculating the delay between two or more measurement points on a single direction by applying [ALT-MARK]. In this way, packet loss, delay and delay variation can be measured for each segment of the network depending on the number and distribution of the available on-path observation points. When these observation points are applied at network borders, the alternate-marking signal can be used to measure the performance of QUIC traffic within a

network operator's own domain of responsibility. own portion of the network.

B.3. Bufferbloat Mitigation in Cellular Networks

Cellular networks consist of multiple Radio Access Networks (RAN) where mobile devices are attached to base stations. It is common that base stations from different vendors and different generations are deployed in the same cellular network.

Due to the dynamic nature of RANs, base stations have typically been provisioned with large buffers to maximize throughput despite rapid changes in capacity. As a side effect, bufferbloat has become a common issue in such networks [WWMM-BLOAT].

An effective way of mitigating bufferbloat without sacrificing too much throughput is to deploy Active Queue Management (AQM) in bottleneck routers and base stations. However, due to the variation in deployed base-stations it is not always possible to enable AQM at the bottlenecks, without massive infrastructure investments.

An alternative approach is to deploy AQM as a network function in a more centralized location than the traditional bottleneck nodes. Such an AQM monitors the RTT progression of flows and drops or marks packets when the measured latency is indicative of congestion. Such a function also has the possibility to detect misbehaving flows and reduce the negative impact they have on the network.

B.4. Locating WiFi Problems in Home Networks

Many residential networks use WiFi (802.11) on the last segment, and WiFi signal strength degradation manifests in high first-hop delay, due to the fact that the MAC layer will retransmit packets lost at that layer. Measuring the RTT between endpoints on the customer network and parts of the service provider's own infrastructure (which have predictable delay characteristics) can be used to isolate this cause of performance problems.

The network provider can measure the RTT and packet loss in the home gateway or an upstream point if there is no access to home gateway. A problem in the WiFi network is identified by seeing high delay and low packet loss.

These measurements are particularly useful for traffic which is latency sensitive, such as interactive video applications. However, since high latency is often correlated with other network-layer issues such as chronic interconnect congestion [IMC-CONGESTION], it

is useful for general troubleshooting of network layer issues in an interdomain setting.

In this case, multiple RTT samples per flow are useful less for observing intraflow behavior, and more for generating sufficient samples for a given aggregate to make a high-quality measurement.

B.5. Internet Measurement Research

As a large, distributed, engineered system with no centralized control, the Internet has emergent properties of interest to the research community not just for purely scientific curiosity, but also to provide applicable guidance to Internet engineering, Internet protocol design and development, network operations, and policy development. Latency measurements in particular are both an active area of research as well as an important tool for certain measurement studies (see, e.g. [IMC-TCPSIG], from the most recent Internet Measurement Conference). While much of this work is currently done with active measurements, the ability to generate latency samples passively or using a hybrid measurement approach (i.e., through passive observation of purpose-generated active measurement traffic; see [RFC7799]) can drastically increase the efficiency and scalability of these studies. A latency spin bit would make these techniques applicable to QUIC, as well.

Appendix C. Alternate RTT Measurement Approaches for Diagnosing QUIC flows

There are three broad alternatives to explicit signaling for passive RTT measurement of the RTT experienced by QUIC flows.

C.1. Handshake RTT measurement

The first of these is handshake RTT measurement. As described in [QUIC-MGT], the packets of the QUIC handshake are distinguishable on the wire in such a way that they can be used for one RTT measurement sample per flow: the delay between the client initial and the server cleartext packet can be used to measure "upstream" RTT (between the observer and the server), and the delay between the server cleartext packet and the next client cleartext packet can be used to measure "downstream" RTT (between the client and the observer). When RTT measurements are used in large aggregates (all flows traversing a large link, for example), a methodology based on handshake RTT could be used to generate sufficient samples for some purposes without the spin bit.

However, this methodology would rely on the assumption that the difference between handshake RTT and nominal in-flow RTT is

negligible. Specifically, (1) any additional delay required to compute any cryptographic parameters must be negligible with respect to network RTT; (2) any additional delay required to establish state along the path must be negligible with respect to network RTT; and (3) network treatment of initial packets in a flow must be identical to that of later packets in the flow. When these assumptions cannot be shown to hold, spin-bit based RTT measurement is preferable to handshake RTT measurement, even for applications for which handshake RTT measurement would otherwise be suitable.

C.2. Parallel active measurement

The second alternative is parallel active measurement: using ICMP Echo Request and Reply [RFC0792] [RFC4433], a dedicated measurement protocol like TWAMP [RFC5357], or a separate diagnostic QUIC flow to measure RTT. Regardless of protocol, the active measurement must be initiated by a client on the same network as the client of the QUIC flow(s) of interest, or a network close by in the Internet topology, toward the server. Note that there is no guarantee that ICMP flows will receive the same network treatment as the flows under study, both due to differential treatment of ICMP traffic and due to ECMP routing (see e.g. [TOKYO-PING]). TWAMP and QUIC diagnostic flows, though both use UDP, have similar issues regarding ECMP. However, in situations where the entity doing the measurement can guarantee that the active measurement traffic will traverse the subpaths of interest (e.g. residential access network measurement under a network architecture and business model where the network operator owns the CPE), active measurement can be used to generate RTT samples at the cost of at least two non-productive packets sent though the network per sample.

C.3. Frequency Analysis

The third alternative, proposed during the QUIC RTT design team process, relies on the inter-packet spacing to convey information about the RTT, and would therefore allow measurements confined to a single direction of transmission, as described in [CARRA-RTT].

We evaluated the applicability of this work to passive RTT measurement in QUIC, and found it wanting. We assembled a toolchain, as described in [NOSPIN], that allowed evaluation of a critical aspect of the [CARRA-RTT] method: extraction of inter-packet times of real packet streams and the analysis of frequencies present in the packet stream using the Lomb-Scargle Periodogram. Several streams were evaluated, as summarized below:

- It seems that Carra et al. [CARRA-RTT] took the noisy and low-confidence results of a statistical process (no RTT-related

frequency has been detected even after using very low alpha confidence) and added heuristics with sliding-window averaging to infer the fundamental frequency and RTT present in a unidirectional stream.

- There appear to be several limitations on the streams that are applicable. Streams with long RTT (~50ms) are more likely to be suitable (having a better match between packet rate and relatively low frequencies to detect).

- None of the TCP streams analysed (to date) possess a sufficient packet rate such that the measured fundamental frequency or the multiples of the fundamental are actually within the detectable range.

- "Ideal" interarrival time streams were simulated with uniform sampling and period. The Lomb-Scargle Periodogram is surprisingly unable to detect the fundamental frequency at 100 Hz from the constant 10 ms packet spacing.

- It is not clear if IETF QUIC protocol stream will possess the same inter-packet arrival time features as TCP streams. Also, Carra et al. note that their process may not work if the TCP stream encounters a bottleneck, which would be an essential circumstance for network troubleshooting. Mobile networks with time-slot service disciplines would likely cause similar issues as a bottleneck, by imposing their time-slot interval on the spacing of most packets.

- The Carra et al. [CARRA-RTT] calculation of minimum and maximum frequencies that can be detected may not be applicable when the inter-arrival times are (both) the signal being detected and govern the non-uniform sampling frequency.

Appendix D. Greasing

Routes, congestion levels and therefore latency between two fixed QUIC endpoints, as well as the shape of individual application flows, fluctuate in ways that are not totally predictable by an on path observer. In general, there is no a-priori pattern for the spin-bit distribution that will always materialise on a certain flow aggregate, even for a single user.

There has been discussion in the QUIC working group that greasing could be a strategy to counter an evil access provider that might gate access to its users on a valid spin bit signal. Let's accept for a moment this threat model and consider the practical case of a home gateway that temporarily misbehaves, for example draining its

queues slower than it would normally do while a firmware download is in progress. It would be ill-considered for an access provider (even a malicious one) to block, or otherwise interfere with, QUIC flows originating from behind that CPE solely based on the fact that RTTs are now different from "usual". In fact, providing a numerical assessment of what such "usual" RTT looks like would necessarily include many paths with different length, and considerable RTT variability within any fixed path, which is clearly beyond most ISPs' reach. But even assuming it were, there is a simple cost-benefit counterargument here that the same effect (i.e., gating traffic from or to a given user based on observed traffic patterns) could be achieved with much cheaper and effective means (e.g., [SHBAIR]).

So, the potential for ossification appears to be extremely low. Since it depends on so much external noise, the spin-bit result variability is self-greasing to an extent. In fact, implementing explicit greasing around the spin-bit might even be harmful as it would potentially erode confidence in the veracity of the signal.

However, if a greasing algorithm is really needed - for example, if we want to reuse the bit with different semantics in the future (i.e.: the spin-bit is not included in the header invariants), one very simple implementation would be as follows: each server will refuse to spin its bit on a per-flow basis with a given probability p, instead leaving it stuck to a randomly chosen value, 0 or 1. The client will then end up leaving its bit stuck to the opposite value, or could detect this condition and also pick a randomly chosen stuck value. The value chosen for p must be small enough to let the spin-bit mechanics work and large enough not to be seen as an error instead of an intentional protocol feature.

Authors' Addresses

Brian Trammell (editor)
ETH Zurich

Email: ietf@trammell.ch

Piet De Vaere
ETH Zurich

Email: piet@devae.re

Roni Even
Huawei

Email: roni.even@huawei.com

Giuseppe Fioccola
Telecom Italia

Email: giuseppe.fioccola@telecomitalia.it

Thomas Fossati
Nokia

Email: thomas.fossati@nokia.com

Marcus Ihlar
Ericsson

Email: marcus.ihlar@ericsson.com

Al Morton
AT&T Labs

Email: acmorton@att.com

Emile Stephan
Orange

Email: emile.stephan@orange.com