TSVWG
Internet-Draft
Intended status: Experimental
Expires: May 7, 2020

M. Cociglio
Telecom Italia
G. Fioccola
Huawei Technologies
F. Bulgarella
R. Sisto
Politecnico di Torino
November 4, 2019

New Spin bit enabled measurements with one or two more bits
draft-cfb-tsvwg-spinbit-new-measurements-00

Abstract

This document introduces additional measurements by using the same spin bit signal as defined in [I-D.trammell-tsvwg-spin] and [I-D.trammell-ippm-spin]. The spin bit signal alone is not enough to evaluate correctly in every network condition the RTT of a flow. In order to solve this problem, it is theorized the possibility of introducing an additional validation signal called delay bit, similar to what is done by the Valid Edge Counter (VEC), but using just one bit instead of two. An alternative with two bits is also introduced with a so called loss bit. More in general a loss signal is defined to measure packet loss and two alternatives are presented with one bit and two bits.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

This Internet-Draft will expire on May 7, 2020.

Copyright Notice

Table of Contents

1. Introduction

Both [I-D.trammell-tsvwg-spin] and [I-D.trammell-ippm-spin] define an explicit per-flow transport-layer signal for hybrid measurement of end-to-end RTT. This signal consists of three bits: a spin bit, which oscillates once per end-to-end RTT, and a two-bit Valid Edge Counter (VEC), which compensates for loss and reordering of the spin bit to increase fidelity of the signal in less than ideal network conditions.

In this document it is introduced the delay bit, that is a single bit signal that can be used together with the spin bit by passive observers to measure the RTT of a network flow, avoiding the spin bit ambiguities that arise as soon as network conditions deteriorate. Unlike the spin bit, which is actually set in every packet transmitted on the network, the delay bit is set only once per round trip.

This document defines a hybrid measurement RFC 7799 [RFC7799] path signal to be embedded into a transport layer protocol, explicitly intended for exposing end-to-end RTT to measurement devices on path.

The document introduces a mechanism applicable to any transport-layer protocol, then explains how to bind the signal to a variety of IETF transport protocols, and in particular to QUIC and TCP.

The application of the Spin bit to QUIC is described in [I-D.ietf-quic-spin-exp] which adds the spin bit only (without the VEC) to QUIC for experimentation purposes.

Note that both the spin bit and the delay bit are inspired by RFC 8321 [RFC8321]. This is also mentioned in [I-D.trammell-quic-spin].

2. Spin bit and Delay bit mechanism

The main idea is to have a single packet, with a second marked bit (the delay bit), that bounces between client and server during the entire connection life. This single packet is called Delay Sample.

A simple observer placed in an intermediate point, tracking the delay sample and the relative timestamp in every spin bit period, can measure the end-to-end round trip delay of the connection. In the same way as seen with the spin bit and the VEC, it is possible to carry out other types of measurements. The next paragraphs give an overview of the observer capabilities.

In order to describe the delay sample working mechanism in detail, we have to distinguish two different phases which take part in the delay bit lifetime: initialization and reflection. The initialization is the generation of the delay sample, while the reflection realizes the bounce behavior of this single packet between the two endpoints.

The next figure describes the Delay bit mechanism: the first bit is the spin bit and the second one is the delay bit.

```
   +--------+   --  --  --  --  --   +--------+
   |        |       ----------->     |        |
   | Client |                        | Server |
   |        |      <-----------      |        |
   +--------+   --  --  --  --  --   +--------+

   (a) No traffic at beginning.

   +--------+   00  00  01  --  --   +--------+
   |        |       ----------->     |        |
   | Client |                        | Server |
   |        |      <-----------      |        |
   +--------+   --  --  --  --  --   +--------+

    (b) The Client starts sending data and
     sets the first packet as Delay Sample.

   +--------+   00  00  00  00  00   +--------+
   |        |       ----------->     |        |
   | Client |                        | Server |
   |        |      <-----------      |        |
   +--------+   --  --  01  00  00   +--------+

    (c) The Server starts sending data
     and reflects the Delay Sample.

   +--------+   10  10  11  00  00   +--------+
   |        |       ----------->     |        |
   | Client |                        | Server |
   |        |      <-----------      |        |
   +--------+   00  00  00  00  00   +--------+

   (d) The Client inverts the spin bit and
    reflects the Delay Sample.

   +--------+   10  10  10  10  10   +--------+
   |        |       ----------->     |        |
   | Client |                        | Server |
   |        |      <-----------      |        |
   +--------+   00  00  11  10  10   +--------+
```

```
      (e) The Server reflects the Delay Sample.

      +--------+   00  00  01  10  10   +--------+
      |        |       ----------->     |        |
      | Client |                        | Server |
      |        |      <-----------      |        |
      +--------+   10  10  10  10  10   +--------+

      (f) The client reverts the spin
       bit and reflects the Delay Sample.
```

Figure 1: Spin bit and Delay bit

2.1. Delay Sample generation

During this first phase, endpoints play different roles. First of all a single delay sample must be bouncing per round trip period (and so per spin bit period). According to that statement and in order to simplify the general algorithm, the delay sample generation is in charge of just one of the two endpoints:

- o the Client, when connection starts and spin bit is set to 0, initializes the delay bit of the first packet to 1, so it becomes the delay sample for that marking period. Only this packet is marked with the delay bit set to 1 for this round trip period; the other ones will carry only the spin bit;

- o the server never initializes the delay bit to 1; its only task is to reflect the incoming delay bit into the next outgoing packet only if certain conditions occur.

Theoretically, in absence of network impairments, the delay sample should bounce between client and server continuously, for the entire duration of the connection. Actually, that is highly unlikely mainly for two different reasons:

1) the packet carrying the delay bit might be lost during its journey on the network which is unreliable by definition;

2) one of the two endpoints could stop or delay sending data because the application is limiting the amount of traffic transmitted;

To deal with these problems, the algorithm provides a procedure to regenerate the delay sample and to inform a possible observer that a problem has occurred, and then the measurement has to be restarted.

2.1.1. The recovery process

In order to relieve the server from tasks that go beyond the mere reflection of the sample, even in this case the recovery process belongs to the client. A fundamental assumption is that a delay sample is strictly related to its spin bit period. Considering this rule, the client verifies that every spin bit period ends with its delay sample. If that does not happen and a marking period terminates without a delay sample, the client waits a further empty period; then, in the following period, it reinitializes the mechanism by setting the delay bit of the first outgoing packet to 1, making it the new delay sample. The empty period is needed to inform the intermediate points that there was an issue and a new delay measurement session is starting.

2.2. Delay Sample reflection

The reflection is the process that enables the bouncing of the delay sample between client and server. The behavior of the two endpoints is slightly different. With the exception of the client that, as previously exposed, generates a new delay sample, by default the delay bit is set to 0.

Server side reflection: when a packet with the delay bit set to 1 arrives, the server marks the first packet in the opposite direction as the delay sample, if it has the same spin bit value. While if it has the opposite spin bit value this sample is considered lost.

Client side reflection: when a packet with delay bit set to 1 arrives, the client marks the first packet in the opposite direction as the delay sample, if it has the opposite spin bit value. While if it has the same spin bit value this sample is considered lost.

In both cases, if the outgoing marked packet is transmitted with a delay greater than a predetermined threshold after the reception of the incoming delay sample (1ms by default), reflection is aborted and this sample is considered lost.

It is noteworthy that differently from what happens with the VEC for which the reflection always concerns the edge of the period, in this case reflection takes place for the packet that is carrying the delay bit regardless of its position within the period. For this reason it is necessary to introduce that condition of validation in order to identify and discard those samples that, due to reordering, might move to a contiguous period. Furthermore, by introducing a threshold for the retransmission delay of the sample, it is possible to eliminate all those measurements which, due to lack of traffic on the endpoints, would be overestimated and not true. Thus, the maximum estimation error, without considering any other delays due to flow control, would amount to twice the threshold (e.g. 2ms) per measurement, in the worst case.

3. Using the Spin bit and Delay bit for Hybrid RTT Measurement

Unlike what happens with the spin bit for which it is necessary to validate or at least heuristically evaluate the goodness of an edge, the delay sample can be used by an intermediate observer as a simple demarcator between a period and the following one eliminating the ambiguities on the calculation of the RTT found with the analysis of the spin-bit only. The measurement types, that can be done from the observation of the delay sample, are exactly the same achievable with the spin bit only (with or without the VEC).

3.1. End-to-end RTT measurement

The delay sample generation process ensures that only one packet marked with the delay bit set to 1 runs back and forth on the wire between two endpoints per round trip time. Therefore, in order to determine the end-to-end RTT measurement of a QUIC flow, an on-path passive observer can simply compute the time difference between two delay samples observed in a single direction. Note that a measurement, to be valid, must take into account the difference in time between the timestamps of two consecutive delay samples belonging to adjacent spin-bit periods. For this reason, an observer, in addition to intercepting and analyzing the packets containing the delay bit set to 1, must maintain awareness of each spin period in such a way as to be able to assign each delay sample to its period and, at the same time, identifying those periods that do not contain it.

3.2. Half-RTT measurement

An on-path passive observer that is sniffing traffic in both directions -- from client to server and from server to client -- can also use the delay sample to measure "upstream" and "downstream" RTT components. Also known as the half-RTT measurement, it represents the components of the end-to-end RTT concerning the paths between the client and the observer (upstream), and the observer and the server (downstream). It does this by measuring the delay between a delay sample observed in the downstream direction and the one observed in the upstream direction, and vice versa. Also in this case, it should verify that the two delay samples belong to two adjacent periods, for the upstream component, or to the same period for the downstream component.

3.3. Intra-domain RTT measurement

Taking advantage of the half-RTT measurements it is also possible to calculate the intra-domain RTT which is the portion of the entire RTT used by a QUIC flow to traverse the network of a provider (or part of it). To achieve this result two observers, able to watch traffic in both directions, must be employed simultaneously at ingress and egress of the network to be measured. At this point, to determine the delay between the two observers, it is enough to subtract the two computed upstream (or downstream) RTT components.

The spin bit is an alternate marking generated signal and the only difference than RFC 8321 [RFC8321] is the size of the alternation that will change with the flight size each RTT. So 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 and it can be easily performed by calculating the delay between two or more measurement points on a single direction by applying RFC 8321 [RFC8321].

4. Observer's algorithm and Waiting Interval

Given below is a formal summary of the functioning of the observer every time a delay sample is detected. A packet containing the delay bit set to 1:

- if it has the same spin bit value of the current period and no delay sample was detected in the previous period, then it can be used as a left edge (i.e. to start measuring an RTT sample), but not as a right edge (i.e. to complete and RTT measurement since the last edge). If the observation point is symmetric (i.e. it can see both upstream and downstream packets in the flow) and in the current period a delay sample was detected in the opposite direction (i.e. in the upstream direction), the packet can also be used to compute the downstream RTT component.

- if it has the same spin bit value of the current period and a delay sample was detected in the previous period, then it can be used at the same time as a left or right edge, and to compute RTT component in both directions.

Like stated previously, every time an empty period is detected, the observer must restart the measurement process and consider the next delay sample that will come as the beginning of a new measure, then as a left edge. As a result, being able to assign the delay sample to the corresponding spin period becomes a crucial factor for the proper functioning of the entire algorithm.

Considering that the division into periods is realized by exploiting the spin bit square wave, it is easy to understand that the presence of spurious spin edges -- caused by packet reordering -- would inevitably lead the observer to overestimate the amount of periods actually present in the transmission. This results in a greater number of empty periods detected and the consequent decrease of the actual RTT samples achievable. Therefore, in order to maximize the performance of the whole algorithm, the observer must implement a mechanism to filter out spurious spin edges.

To face this problem the waiting interval has to be introduced. Basically, every time a spin bit edge is detected, the observer sets a time interval during which it rejects every potential spurious edges observed on the wire. While, at the end of the interval it starts again to accept changes in the spin bit value. This guarantees a proper protection against the spurious edges in relation to the size of the interval itself. For instance, an interval of 5ms is able to filter out edges that have been reordered by a maximum of 5ms. Clearly, the mechanism does its job for intervals smaller than the RTT of the observed connection (if RTT is smaller than the waiting interval the observer can’t measure the RTT).

5. Adding a Loss signal for Packet loss measurement

It is possible to introduce a mechanism to evaluate also the packet loss together with the delay measurement. This can be achieved by introducing the loss signal, a single or two bits signal whose purpose is to mark a variable number of packets (from live traffic) which are exchanged two times between the endpoints realizing a two round-trip reflection. The overall exchange comprises:

- o The client first selects, generates and consequent transmits to the server a first train of packets, by marking the loss bit to 1;

- o The server, upon reception from the client of each one of the packets included in the first train, reflects to the client a respective second train of packets of the same size as the first train received, by marking the loss bit to 1;

- o The client, upon reception from the server of each one of the packets included in the second train, reflects to the server a respective third train of packets of the same size as the second train received, by marking the loss bit to 1;

- o The server, upon reception from the client of each one of the packets included in the third train, finally reflects to the client a respective fourth train of packets of the same size as the third train received, by marking the loss bit to 1.

Packets belonging to the first round (first and second train) represent the Generation Phase while those belonging to the second round (third and fourth train) represent the Reflection Phase.

A passive on-path observer, placed on whatever direction, can trivially count and compare the number of marked packets seen during the two mentioned phases (i.e. the first and third or the second and the fourth trains of packets, depending on which direction is observed) and estimate the loss rate experienced by the connection. This process is repeated continuously to obtain more measurements as long as the endpoints exchange traffic. These measurements can be called Round Trip(RT) losses

The general algorithm shown above gives an idea of its underlying principles but is not enough to make the whole process working properly.

Firstly, there is the issue that packet rates in the two directions may be different. Therefore, the right number of packets to be marked has to be chosen in order to avoid their congestion on the slowest traffic direction. As a consequence, this number is inevitably equal to the amount of packets transited, indeed, on the slowest direction. This problem can be easily addressed by a method wherein the two endpoints of a communication exchange marked packets interleaved with unmarked packets. From an implementation point of view, this result can be achieved by introducing a single token system that adjusts the number of outgoing marked packets. Basically, the token is enabled every time a packet arrives and disabled when a marked packet is transmitted. Since the creation of the initial train of marked packets is carried out by the client, the management and use of this single token is also assigned to it, which in fact "calculates" the correct number of packets to be marked each time.

Secondly, a mechanism to individually identify each train of packets must be provided to enable the observer to distinguish between trains belonging to different phases (Generation and Reflection). About this point, different approaches are used depending on the number of bits of the loss signal and it will be discussed in the next sections.

5.1. Round Trip Packet Loss measurement

Since the measurements are performed on a portion of the traffic exchanged between client and server, the observer calculates the end-to-end Round Trip Packet Loss that, statistically, will be equal to the loss rate experienced by the connection along the entire network path. So this measurement can be simply referred as the Round Trip Packet Loss (RTPL).

In addition, this methodology allows the Half-RTPL measurement and the Intra-domain RTPL measurement, in the same way as described in the previous sections for RTT measurement.

6. RTT dependent Packet Loss using one bit loss signal

The single bit loss signal is implemented using just one bit: marked packets have this bit set to 1, whereas unmarked ones have it set to 0. This solution requires a working spin-bit signal used to separate different trains of packets. In particular, a "pause" of at least one empty spin-bit period is introduced between each phase of the algorithm. An on-path observer can determine in this way if a phase (and therefore a train of packets) is ended and a new one is starting.

The client is in charge of almost the entire complexity of the algorithm. Its task can be summarized in 4 different points:

1. The client starts generating marked packets for two consecutive spin-bit periods; it maintains a generation token that is enabled every time a packet arrives and disabled when another one is forwarded. When this token is disabled, the generation process is paused (i.e. outgoing packets are transmitted unmarked) and resumes as soon as its value returns true, and that happens as soon as a packet is received. In addition, at the end of the first spin-bit period spent in generation, the reflection counter is unlocked to start counting incoming marked packets which will be later reflected;

2. When the generation is completed, the client waits to see in input an empty spin-bit period so as to be sure that everyone has seen at least that empty period. This one will be used by the observer as a divider between generated and reflected packets. During this phase, all the outgoing packets are forwarded with the loss bit set to 0. The reflection counter is still incremented every time a marked packet arrives;

3. The client starts reflecting marked packets until the reflection counter is zeroed; the generation token is also used (in the same way) during this phase to avoid congestion on the slowest traffic direction. In addition, at the end of the first spin-period spent in reflection, the reflection counter is locked to avoid incoming reflected packets incrementing it;

4. When the reflection is completed, the client waits to see in input an empty spin-bit period so as to be sure that everyone has seen at least that empty period. This one will be used by the observer as a divider between reflected and newly generated packets. During this phase, all the outgoing packets are forwarded with the loss bit set to 0. The whole process restarts going back to the first point.

As previously anticipated, the server simply reflects each incoming marked packet sent by the client. It maintains a simple counter that is incremented every time a marked packet arrives and decremented when a marked one is sent in the opposite direction.

This one bit loss signal methodology replies and exposes the RTT of the connection on the wire in any case, when the spin bit and the delay bit are used and when these are disabled.

6.1. Observer's logic for one bit loss signal

The on-path observer, placed in any direction, counts marked packets and separates different trains detecting empty spin-bit periods between them (one or more). Then, it simply computes the difference between a Generation train and a Reflection train to produce a statistical measurement of the Round Trip Packet Loss (RTPL) and of the connection end-to-end loss rate.

Here is an example. Packets are represented by two digits (first one is the spin bit, second one is the loss bit):

```
     Generation          Pause           Reflection       Pause
  ____________________ ______________ ____________________ ________
 |                    |              |                    |        |
  01 01 00 01 11 10 11 00 00 10 10 10 01 00 01 01 10 11 10 00 00 10
```

Figure 2: one bit loss signal example

Note that 5 marked packets have been generated of which 4 reflected.

7. RTT independent Packet Loss using two bits loss signal

An RTT independent version of this algorithm requires two bits and does not rely on the spin-bit signal to enable pause detection. That is because packets generated and reflected by the client are marked using two different marking values thus removing the need of introducing a pause between them. Furthermore, instead of generating marked packets for the duration of two spin-bit periods (as seen in the one bit loss signal), a fixed duration for the generation phase can be used (e.g., 100ms).

In this way, no information related to the RTT of the connection is exposed on wire.

Using a two bits loss signal, four possible values can be used inside each packet (i.e. 0 to 3). During the Generation phase, marked packets have the loss value set to 1 whereas unmarked ones to 0. On the contrary, during the Reflection phase, marked packets have the loss value set to 2 whereas unmarked ones to 3. By doing so, even unmarked packets have their own alternate marking methodology that can be used by intermediate points to compute the one-way loss rate between them (RFC 8321 [RFC8321]).

Even in this case, the client is in charge of almost the entire complexity of the algorithm. Its task can be summarized in 2 different points:

1. The client generates marked packets (i.e. with loss bits set to 1) for 100ms; it also maintains a generation token that is enabled every time a packet arrives and disabled when another one is forwarded. When this token is disabled, the generation process is paused (i.e. outgoing packets are transmitted unmarked with the loss bits set to 0) and resumes as soon as its value returns true.

2. When the generation is completed, the client starts reflecting marked packets (i.e. with loss bits set to 2) until the reflection counter is zeroed and for at least 100ms. The generation token is also used during this phase to avoid congestion on the slowest traffic direction; however, in this case, "unmarked" packets are transmitted with the loss bit set to 3. The whole process restarts going back to the first point.

Independently of the current phase of the algorithm, the reflection counter is increased every time a packet carrying a loss value equal to 1 arrives. Moreover, depending on the connection RTT, the client should vary the duration of the generation phase to different values. For example, for connection below 100ms of RTT the client generates for 100ms; for connection below 300ms of RTT it generates for 300ms and for connection below 1s of RTT it generates for 1000ms. This is necessary to ensure that the client has already received generated marked packets before the beginning of the reflection phase.

As regards the role of the server, it simply reflects each incoming marked packet sent by the client. It maintains two different counters for generated and reflected packets (i.e. loss bits to 1 and

2) in concomitance with a mechanism to reflect in output the same number of marked packets in the same order of arrival (with at most the reordering of packets arrived out of sequence).

7.1. Observer's logic for two bits loss signal

The on-path observer, placed in any direction, counts marked packets belonging to different phases simply looking at the loss value carried by each packet (therefore, it does not look at the spin-bit value anymore). Then, in the same way seen for the previous one bit algorithm, it simply computes the difference between a Generation train and a Reflection train to produce a statistical measurement of the Round Trip Packet Loss (RTPL) and of the connection end-to-end loss rate. Moreover, it can also count unmarked packets and, cooperating with a second observer placed in the same direction, compute the one-way loss rate between two intermediate points using the alternate marking methodology (RFC 8321 [RFC8321]).

Here is an example. Packets are represented by a single digit corresponding to the carried two-bits loss value (0 to 3):

```
       Generation            Reflection             Generation
  _____________________ _____________________ _____________________
 |                     |                     |                     |
  1 1 0 1 1 1 1 0 1 1 0 2 2 2 2 3 2 3 3 2 3 3 1 1 0 1 0 0 1 1 0 1 0
```

Figure 3: two bits loss signal example

Note that 8 marked packets have been generated of which 6 reflected; then again 6 marked packets are generated.

8. Protocols

8.1. QUIC

The binding of the delay bit signal to QUIC is partially described in [I-D.ietf-quic-spin-exp], which adds the spin bit only to the QUIC protocol. From an implementation point of view, the delay bit is placed in the partially unencrypted (but authenticated) QUIC header, alongside the spin bit, occupying one of the two bits left reserved for future experiments. As things stand, according to [I-D.ietf-quic-transport], the proposed scheme of the first header's byte would be 01SDRKPP.

Regarding the loss signal, since the use of the spin bit is not mandatory and many connection may not have it spinning, two different configuration are proposed:

If the spin-bit IS enabled (i.e. the RTT is already exposed on wire), use the 1 bit loss signal alongside the delay bit to improve delay measurements accuracy; in this configuration, the proposed scheme of the first header's byte would be 01SDLKPP;

If the spin-bit IS NOT enabled, use the 2 bits loss signal just to measure connection loss rate without exposing any RTT related information on wire; in this configuration, the proposed scheme of the first header's byte would be 01SLLKPP.

This implies that an observer must be able to determine whether the spin bit is active and correctly spinning or not (choosing, accordingly, the right version of packet loss measurement to be used).

8.2. TCP

The signal can be added to TCP by defining bit 4 of bytes 13-14 of the TCP header to carry the spin bit, and eventually bits 5 and 6 to carry additional information, like the delay bit and the 1 bit loss signal (or the two bits loss signal).

9. Security Considerations

The privacy considerations for the hybrid RTT measurement signal are essentially the same as those for passive RTT measurement in general.

10. Acknowledgements

tbc

11. IANA Considerations

tbc

12. References

12.1. Normative References

[I-D.ietf-quic-spin-exp]
Trammell, B. and M. Kuehlewind, "The QUIC Latency Spin Bit", draft-ietf-quic-spin-exp-01 (work in progress), October 2018.

[I-D.ietf-quic-transport]
Iyengar, J. and M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport", draft-ietf-quic-transport-23 (work in progress), September 2019.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[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>.

[RFC8321] Fioccola, G., Ed., Capello, A., Cociglio, M., Castaldelli, L., Chen, M., Zheng, L., Mirsky, G., and T. Mizrahi, "Alternate-Marking Method for Passive and Hybrid Performance Monitoring", RFC 8321, DOI 10.17487/RFC8321, January 2018, <https://www.rfc-editor.org/info/rfc8321>.

12.2. Informative References

[I-D.trammell-ippm-spin]
Trammell, B., "An Explicit Transport-Layer Signal for Hybrid RTT Measurement", draft-trammell-ippm-spin-00 (work in progress), January 2019.

[I-D.trammell-quic-spin]
Trammell, B., Vaere, P., Even, R., Fioccola, G., Fossati, T., Ihlar, M., Morton, A., and S. Emile, "Adding Explicit Passive Measurability of Two-Way Latency to the QUIC Transport Protocol", draft-trammell-quic-spin-03 (work in progress), May 2018.

[I-D.trammell-tsvwg-spin]
Trammell, B., "A Transport-Independent Explicit Signal for Hybrid RTT Measurement", draft-trammell-tsvwg-spin-00 (work in progress), July 2018.

Authors' Addresses

Mauro Cociglio
Telecom Italia
Via Reiss Romoli, 274
Torino 10148
Italy

Email: mauro.cociglio@telecomitalia.it

Giuseppe Fioccola
Huawei Technologies
Riesstrasse, 25
Munich 80992
Germany

Email: giuseppe.fioccola@huawei.com

Fabio Bulgarella
Politecnico di Torino

Email: fabio.bulgarella@guest.telecomitalia.it

Riccardo Sisto
Politecnico di Torino
Corso Duca degli Abruzzi, 24
Torino 10129
Italy

Email: riccardo.sisto@polito.it