

QUIC  
Internet-Draft  
Intended status: Experimental  
Expires: October 22, 2020

T. Li  
K. Zheng  
R. Jadhav  
J. Kang  
Huawei Technologies  
April 20, 2020

**Optimizing ACK mechanism for QUIC**  
**draft-li-quic-optimizing-ack-in-wlan-00**

Abstract

This document analyzes the problems caused by contentions and collisions on wireless medium between data packets and ACKs in WLAN and it proposes an optimized ACK mechanism that can minimize the intensity of ACK Frame in QUIC, improving the performance of transport layer connection.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any 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 October 22, 2020.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in [Section 4](#).e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

<a href="#">1.</a>	Requirements Language . . . . .	<a href="#">2</a>
<a href="#">2.</a>	Problem Statement . . . . .	<a href="#">2</a>
<a href="#">3.</a>	ACK Mechanism in Current QUIC . . . . .	<a href="#">2</a>
<a href="#">4.</a>	Optimized ACK Mechanism for QUIC . . . . .	<a href="#">3</a>
<a href="#">4.1.</a>	Reducing ACK intensity . . . . .	<a href="#">3</a>
<a href="#">4.2.</a>	OWD-based RTTmin estimation . . . . .	<a href="#">4</a>
<a href="#">4.3.</a>	Sender-Side Operation . . . . .	<a href="#">6</a>
<a href="#">4.4.</a>	Receiver-side Operation . . . . .	<a href="#">7</a>
<a href="#">4.5.</a>	Generating ACK . . . . .	<a href="#">7</a>
<a href="#">4.6.</a>	Modification to QUIC Protocol . . . . .	<a href="#">7</a>
<a href="#">4.6.1.</a>	Transport Parameter: ack-intensity-support . . . . .	<a href="#">7</a>
<a href="#">4.6.2.</a>	ACK-INTENSITY Frame . . . . .	<a href="#">8</a>
<a href="#">5.</a>	Security Considerations . . . . .	<a href="#">8</a>
<a href="#">6.</a>	IANA Considerations . . . . .	<a href="#">8</a>
<a href="#">7.</a>	References . . . . .	<a href="#">9</a>
<a href="#">7.1.</a>	Normative References . . . . .	<a href="#">9</a>
<a href="#">7.2.</a>	Informative References . . . . .	<a href="#">9</a>
	Authors' Addresses . . . . .	<a href="#">9</a>

## [1.](#) 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](#)].

## [2.](#) Problem Statement

High-throughput transport over wireless local area network (WLAN) becomes a demanding requirement with the emergence of 4K wireless projection, VR/AR-based interactive gaming, and more. However, the shared nature of the wireless medium induces contention between data transport and backward signaling, such as acknowledgement. ACKs share the same medium route with data packets, causing similar medium access overhead despite the much smaller size of the ACKs. Contentions and collisions, as well as the wasted wireless resources by ACKs, lead to significant throughput decline on the data path.

## [3.](#) ACK Mechanism in Current QUIC

[QUIC-TRANSPORT] specifies a simple delayed ACK mechanism that a receiver can send an ACK for every other packet, and for every packet when reordering is observed, or when the `max_ack_delay` timer expires. However, this ACK mechanism may not match the number of ACKs to the



transport's required intensity under different network conditions. For example, when the data throughput of a WLAN transport is extremely high, QUIC will generate a large number of ACKs. In this case, minimizing the ACK intensity of QUIC is not only a win for data throughput improvement but also a win for energy and CPU efficiency.

## **4. Optimized ACK Mechanism for QUIC**

### **4.1. Reducing ACK intensity**

ACK intensity can be quantified by the unit of Hz, i.e., number of ACKs per second. Byte-counting ACK and periodic ACK are two fundamental ways to reduce ACK intensity on the transport layer.

1. Byte-counting ACK: ACK intensity is controlled by sending an ACK for every  $L$  ( $L \geq 2$ ) incoming full-sized packets, in which the packet size equals to the Max Packet Size (set in the `max_packet_size` parameter in QUIC). The intensity of byte-counting ACK ( $f_b$ ) is proportional to data throughput ( $bw$ ):

$$f_b = bw/L * max\_packet\_size \quad (1)$$

In general,  $f_b$  can be reduced by setting a large value of  $L$ . However, for a given  $L$ ,  $f_b$  increases with  $bw$ . This means when data throughput is extremely high, the ACK intensity still might be comparatively large. In other words, the intensity of byte-counting ACK changes proportionately with bandwidth.

2. Periodic ACK: Byte-counting ACK's unbounded intensity can be attributed to the coupling between ACK sending and packet arrivals. Periodic ACK can decouple ACK intensity from packet arrivals, achieving a bounded ACK intensity when  $bw$  is high. The intensity of periodic ACK ( $f_{pack}$ ) is:

$$f_{pack} = 1/\alpha \quad (2)$$

Where  $\alpha$  is the time interval between two ACKs and is a function of RTT. However, when  $bw$  is extremely low, the ACK intensity is always as high as that in the case of a high throughput. In other words, the intensity of periodic ACK is unadaptable to bandwidth change, which wastes resources.

Combining these two ways, the minimum ACK intensity in a QUIC connection can be set as  $f_{quic} = \min\{f_b, f_{pack}\}$ . Through Equations (1) and (2), we have

$$f_{quic} = \min\{bw/(L * max\_packet\_size), 1/\alpha\} \quad (3)$$



We set  $\alpha = \text{RTTmin}/\beta$ , which means sending  $\beta$  ACKs per RTTmin. RTTmin is the minimum RTT observed for a given network path. As a consequence, the minimum ACK intensity in a QUIC connection can be given as follow:

$$f_{\text{quic}} = \min\{\text{bw}/(L \cdot \text{max\_packet\_size}), \beta/\text{RTTmin}\} \quad (4)$$

where  $\beta$  indicates the number of ACKs per RTT, and  $L$  indicates the number of full-sized data packets counted before sending an ACK. To minimize the ACK intensity, a smaller  $\beta$  or a larger  $L$  is expected. Sara Landstrom et al. has given a lower bound of  $\beta$  in [Sara], i.e.,  $\beta \geq 2$ . An upper bound of  $L$  can also be derived according to the loss rate on the data path ( $\text{plr\_data}$ ) and the ack path ( $\text{plr\_ack}$ ), i.e.,  $L \leq \text{feedback\_info}/(\text{plr\_data} \cdot \text{plr\_ack})$ , where  $\text{feedback\_info}$  denotes the amount of information carried by an ACK

Qualitatively, periodic ACK is applied when bandwidth-delay product ( $\text{bdp}$ ) is large (i.e.,  $\text{bdp} \geq \beta \cdot L \cdot \text{max\_packet\_size}$ ), and byte-counting ACK is applied when  $\text{bdp}$  is small (i.e.,  $\text{bdp} < \beta \cdot L \cdot \text{max\_packet\_size}$ ).

In terms of a transport with a large  $\text{bdp}$ ,  $\beta = 2$  should be sufficient to ensure utilization, but the large bottleneck buffer (i.e., one  $\text{bdp}$ ) makes it necessary to acknowledge data more often. In general, the minimum send window ( $\text{SWNDmin}$ ) can be roughly estimated as follow:

$$\text{SWNDmin} = \beta \cdot \text{bdp} / (\beta - 1) \quad (5)$$

Ideally, the bottleneck buffer requirement is decided by the minimum send window, i.e.,  $\text{SWNDmin} - \text{bdp}$ . Since doubling the ACK frequency reduces the bottleneck buffer requirement substantially from 1  $\text{bdp}$  to 0.33  $\text{bdp}$ ,  $\beta = 4$  is RECOMMENDED to provide redundancy [Sara], being more robust in practice.

#### **4.2. OWD-based RTTmin estimation**

In this document, the RTTmin is the minimum RTT samples observed at the sender for a given network path during a period of time, and OWDmin is the minimum OWD samples observed on the same network path during a period of time.

When multiple packets carrying departure timestamps are transported between endpoints via the same path, an RTT of this path can be sampled at the sender upon receiving an ACK frame. However, when sending fewer ACK frames, more data packets might be received during the ACK interval, generating only one RTT sample among multiple packets is likely to result in biases. For example, a larger minimum



RTT estimate. In general, the higher the throughput, the larger the biases. One alternative way to reduce biases can be that, each ACK frame carries multiple timestamps (as well as ACK delays in [\[QUIC-RECOVERY\]](#) for the sender to generate more RTT samples. However, (1) the overhead is high, which is unacceptable especially under high-bandwidth transport. Also, (2) the number of data packets might be far more than the maximum number of timestamps that an ACK frame is capable to carry.

An RTT estimation system contains a sender and a receiver. The sender can hardly generate per-packet RTT samples, which is the root cause of the minimum RTT estimation biases in the case of sending fewer ACKs. When multiple packets carrying departure timestamps are transported between endpoints via the same path, an RTT of this path can be sampled at the sender upon receiving an ACK frame. However, when sending fewer ACK frames, more data packets might be received during the ACK interval, generating only one RTT sample among multiple packets is likely to result in biases. For example, a larger minimum RTT estimate. In general, the higher the throughput, the larger the biases. One alternative way to reduce biases can be that, each ACK frame carries multiple timestamps (as well as ACK delays in [\[QUIC-RECOVERY\]](#)) for the sender to generate more RTT samples. However, (1) the overhead is high, which is unacceptable especially under high-bandwidth transport. Also, (2) the number of data packets might be far more than the maximum number of timestamps that an ACK frame is capable to carry. Since the receiver is capable to monitor per-packet state, the one-way delay (OWD) of each packet can be easily computed according to the departure timestamps (carried in the packet) and the arrival timestamps of each packet. In this case, QUIC SHOULD adopt the OWD-based RTTmin estimation. The rationale is that the variation of OWD reflects the variation of RTT over near-symmetric links. The OWD-based RTTmin estimation requires the sender to record the departure timestamp in each ack-eliciting packet. Meanwhile, at the receiver, the per-packet OWD samples SHOULD be computed upon packet arrivals and a function of computing the minimum OWD SHOULD be newly added. The receiver then generates an ACK frame to the sender, in which the ACK delay and departure timestamp for the packet that achieves the minimum OWD is reported. The ACK delay is defined as the delay incurred between when the packet is received and when the ACK frame is sent. Based on the information reported by the incoming ACK frames and the ACK arrival timestamps, the sender can generate RTT samples and then compute RTTmin accordingly.

In this document, RTTmin is used to update the ACK intensity. In general, RTTmin can also be used by other modules. For example, some congestion controllers depends on RTTmin to estimate the congestion





window [Neal]. RTTmin is also used by QUIC loss detection to reject implausibly small rtt samples [QUIC-RECOVERY].

#### 4.3. Sender-Side Operation

According to Formula (4), the run-time ACK intensity in QUIC are decided by bw, and RTTmin. Generally, the RTTmin and bw are calculated at the sender.

Before estimating the RTTmin, the RTT samples should be computed based on the ACK frames collected during a period of time. Assume that a packet is sent by the sender at time  $t_1$  and arrives at time  $t_3$ , and the ACK frame is sent at time  $t_4$ . The ACK delay can be computed at the receiver. For example, the receiver computes the ACK delay  $\text{delta}_t = t_4 - t_3$ , and syncs the ACK delay to the sender via an ACK frame. The ACK delay can also be computed at the sender. For example, the receiver directly syncs an ACK frame carrying  $t_4$  and  $t_3$  to the sender, the sender then computes the ACK delay  $\text{delta}_t = t_4 - t_3$ .

The sender therefore computes an RTT sample according to  $\text{delta}_t$ ,  $t_1$ , and the arrival time ( $t_2$ ) of the ACK frame, i.e.,  $\text{RTT\_sample} = t_2 - t_1 - \text{delta}_t$ . Measuring  $\text{delta}_t$  at the receiver assures an explicit correction for a more accurate RTT estimate. RTT samples SHOULD be smoothed using exponentially weighted moving average (EWMA) as specified in [RFC6298]. The sender then computes the RTTmin according to these RTT samples during a period of time.

The bw estimation can be acquired in a similar manner to BBR [Neal]. Since minimizing the ACK intensity induces excessive ACK delay, the value of bw may be the average value over a long period of time. However, the biases introduced in ACK intensity computation is limited.

After computing the  $f_{\text{quic}}$ , the sender periodically syncs it to the receiver to update the intensity of ACK Frame by sending a new ACK-INTENSITY frame.

The sender SHOULD generate an ACK-INTENSITY frame on a regular basis. For example, when the change of  $f_{\text{quic}}$  exceeds a threshold, the ACK-INTENSITY frame should be sent to update the ACK intensity in time. The interval of ACK-INTENSITY frame can also be set according to the update window of RTTmin and bw.



#### **4.4. Receiver-side Operation**

Currently, the QUIC receiver reports ACK delays for only the largest acknowledged packet in an ACK frame, hence an RTT sample is generated using only the largest acknowledged packet in the received ACK frame. For a more accurate RTTmin estimate when sending fewer ACK frames, QUIC SHOULD adopt the OWD-based RTTmin estimation. The OWD-based RTTmin estimation requires the QUIC receiver to filter the departure timestamp for the packet that achieves the minimum OWD during the interval between two ACK frames and report the ACK delay of this packet. Whether redefining the meaning of ACK delay or not, it depends on the negotiation between endpoints of the QUIC connection.

Upon packet arrivals, the receiver is capable to generate per-packet OWD samples according to the difference between packet departure timestamp and packet arrival timestamp. The receiver then computes the minimum OWD by comparing the per-packet OWD samples. The OWD estimation does not require clock synchronization here because the relative values are adopted.

Afterwards, based on the ACK delay and the departure timestamp corresponding to the packet that achieves the minimum OWD, the sender calculates the RTT of this packet as a minimum RTT sample. Ultimately, the minimum RTT is computed according to these minimum RTT samples.

The ACK Delay field SHOULD be carried in the ACK Frame. Other fields carried in the ACK frame have the same meaning as defined in [\[QUIC-RECOVERY\]](#).

The receiver adopts the newly updated ACK intensity once it receives the ACK-INTENSITY frame from the sender.

#### **4.5. Generating ACK**

The newly proposed ACK mechanism SHOULD be applied when there is no out-of-order delivery. When reordering happens, the ACK Frame SHOULD be generated immediately.

#### **4.6. Modification to QUIC Protocol**

##### **4.6.1. Transport Parameter: ack-intensity-support**

A new field named ack-intensity-support should be added for negotiation between both parties whether starting the dynamic ACK intensity function in QUIC connection. The endpoints sends this parameter during handshakes. Only when both parties agree, ACK intensity refreshment can be adopted.



ack-intensity-support (0x XX): This parameter has two values (0 or 1) specifying whether the sending endpoint is willing to adopt ACK intensity refreshment. When the value is set as 1, it means that the sending endpoint want to start ACK intensity refreshment during connection. When the value is set as 0, it means that the sending endpoint does not support this function.

#### 4.6.2. ACK-INTENSITY Frame

An ACK-INTENSITY frame is shown in Figure 1.

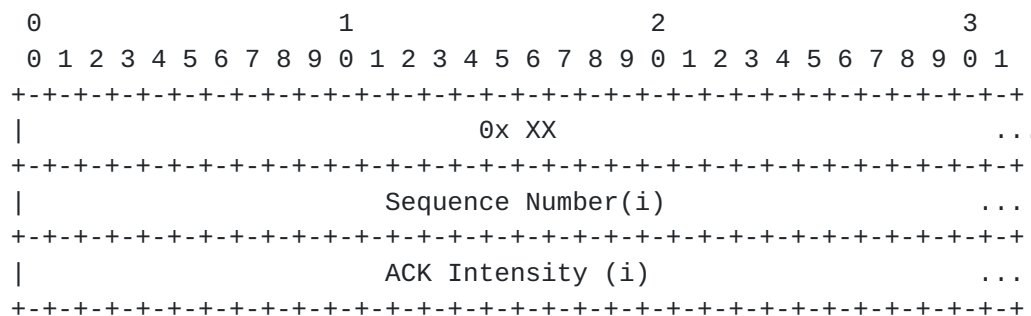


Figure 1: ACK-INTENSITY Frame

An ACK-INTENSITY frame contains the following fields:

Sequence Number: A variable-length integer indicating the sequence number assigned to the ACK-INTENSITY frame by the sender.

ACK Intensity: A variable-length integer indicating the updated `f_quic` calculated by the sender.

ACK-INTENSITY frames are ack-eliciting. However, their loss does not require retransmission.

Multiple ACK-INTENSITY frames SHOULD be generated by the sender during a connection to notify the receiver the variation of ACK intensity requirement under network dynamics.

## 5. Security Considerations

TBD

## 6. IANA Considerations

The value for ack-intensity-support transport parameter and ACK-INTENSITY frame should be allocated.



## **7. References**

### **7.1. Normative References**

- [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>>.
- [RFC6298] Paxson, V., Allman, M., Chu, J., and M. Sargent, "Computing TCP's Retransmission Timer", [RFC 6298](#), DOI 10.17487/RFC6298, June 2011, <<https://www.rfc-editor.org/info/rfc6298>>.

### **7.2. Informative References**

- [Neal] Cardwell, N., Cheng, Y., Gunn, C. S., Yeganeh, S. H., and V. Jacobson, "BBR: Congestion-based congestion control", ACM QUEUE 14(5):20-53, 2016.
- [QUIC-RECOVERY] Iyengar, J., Ed. and I. Swett, Ed., "QUIC Loss Detection and Congestion Control", [draft-ietf-quic-recovery-27](#) (work in progress), February 2020.
- [QUIC-TRANSPORT] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", [draft-ietf-quic-transport-27](#) (work in progress), March 2020.
- [Sara] Landstrom, S. and L. Larzon, "Reducing the tcp acknowledgment frequency", ACM SIGCOMM CCR 37(3):5-16, 2007.

#### Authors' Addresses

Tong Li  
Huawei Technologies  
D2-03, Huawei Industrial Base  
Longgang District  
Shenzhen  
China

Email: [li.tong@huawei.com](mailto:li.tong@huawei.com)





Kai Zheng  
Huawei Technologies  
Information Road, Haidian District  
Beijing  
China

Email: kai.zheng@huawei.com

Rahul Arvind Jadhav  
Huawei Technologies  
D2-03, Huawei Industrial Base  
Longgang District  
Shenzhen  
China

Email: rahul.jadhav@huawei.com

Jiao Kang  
Huawei Technologies  
D2-03, Huawei Industrial Base  
Longgang District  
Shenzhen  
China

Email: kangjiao@huawei.com

