International Journal of Internet, Broadcasting and Communication Vol.11 No.1 17-26 (2019)
http://dx.doi.org/10.7236/IJIBC.2019.11.1.17

IJIBC 19-1-3

A Survey on Congestion Control for CoAP over UDP

Chansook Lim

Department of Computer and Information Communications Engineering
Hongik University, Sejong, Republic of Korea
chansooklim@hongik.ac.kr

Abstract

The Constrained Application Protocol (CoAP) is a specialized web transfer protocol proposed by the IETF for use in IoT environments. CoAP was designed as a lightweight machine-to-machine protocol for resource constrained environments. Due to the strength of low overhead, the number of CoAP devices is expected to rise rapidly. When CoAP runs over UDP for wireless sensor networks, CoAP needs to support congestion control mechanisms. Since the default CoAP defines a minimal mechanism for congestion control, several schemes to improve the mechanism have been proposed. To keep CoAP lightweight, the majority of the schemes have been focused mainly on how to measure RTT accurately and how to set RTO adaptively according to network conditions, but other approaches such as rate-based congestion control were proposed more recently. In this paper, we survey the literature on congestion control for CoAP and discuss the future research directions.

Keywords: *CoAP, congestion, IoT, WSN*

1. Introduction

CoAP [1] defined by the IETF CoRE WG is a lightweight application layer protocol designed to be a specialized web transfer protocol for constrained nodes and constrained networks. CoAP is based on the REST (Representational State Transfer) model that defines architectural principles for designing web services. There are many CoAP implementations for diverse platforms [2] and CoAP based solutions have been realized for a variety of application areas including Cisco's FAN (field area network) [3] . In particular, since 2017, the number of CoAP devices has rapidly increased [4].

CoAP was originally designed to run on UDP, unlike HTTP that runs over TCP. Since CoAP was designed for networks consisting of resource constrained devices, a congestion control scheme for CoAP needs to be effective with minimal consumption of resources. This is a main reason why UDP is preferable to TCP in some circumstance. More recently, the CoRE WG defined CoAP over TCP for better integration with existing network infrastructures where middleboxes such as firewalls or NATs may block UDP packets [5]. Still, CoAP over lightweight UDP may be a good option in several respects. As mentioned in [5], CoAP

Manuscript Received: Dec. 16, 2018 / Revised: Dec. 20, 2018 / Accepted: Dec. 28, 2018
Corresponding Author: chansooklim@hongik.ac.kr
Tel: +82-44-860-2549, Fax: +82-44-865-0460
Department of Software and Communications Engineering, Hongik University, Sejong, Republic of Korea

over UDP is more efficient for transferring a small amount of data. Real-time IoT applications also may prefer UDP. Moreover, some IoT applications may need only congestion control without complete reliability in case that they need event-level reliability instead of packet-level reliability.

Since the default CoAP only defines a basic congestion control mechanism, the CORE WG has worked on an enhanced mechanism called CoCoA [6]. Besides, several other schemes have been proposed to improve the congestion control mechanism for CoAP. In this paper, we survey the research efforts to control congestion in CoAP/UDP-based networks.

The rest of the paper is organized as follows. Section 2 presents the overview of CoAP for the background knowledge. Section 3 summarizes the congestion control schemes for CoAP. We discuss future research directions in Section 4, and conclude in Section 5.

2. Overview of CoAP

CoAP is a web transfer protocol designed for machine-to-machine communication in constrained environments. CoAP is lightweight in several respects such as the message header size and the message exchange mechanism. Based on the REST architecture, CoAP provides a request/response interaction model between application endpoints. A CoAP client sends a request to a CoAP server using one of four methods, GET, PUT, POST, and DELETE which are mapped to equivalent methods of HTTP. The easy interface with HTTP aims at integration of constrained nodes and networks with Web. Fig. 1 illustrates a typical configuration with CoAP.

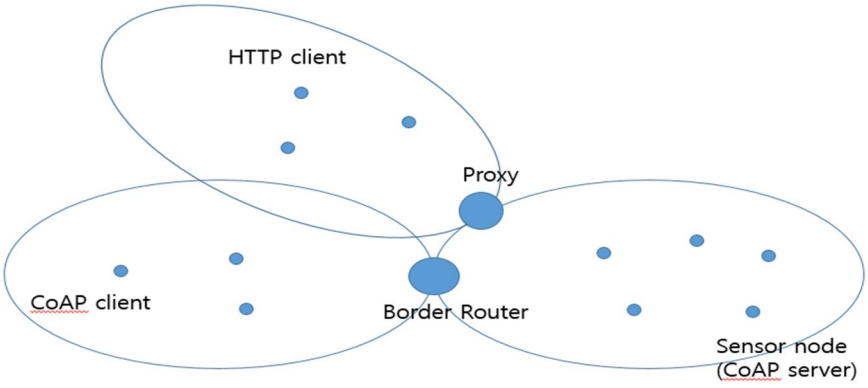

Figure 1. An illustrative configuration with CoAP

In addition to a request/response interaction model, CoAP also provides a simple publish/subscribe model which enables CoAP clients to observe resources, as specified in RFC 7641 [7]. When a client is interested in the current representation of a resource over a period of time, polling approaches do not work well due to high overhead. With the observe option, a CoAP client retrieves the representation of a resource and requests the server to update the representation as long as the client is interested in the resource.

CoAP supports four types of messages: Confirmable (CON), Non-confirmable (NON), Acknowledgement (ACK), and Reset. If reliable transmission is not required, NON messages can be sent. A sender may choose to transmit multiple copies of a NON message within a given maximum limit. CoAP provides lightweight reliability by sending Confirmable (CON) messages, which require ACK messages. If the sender does not receive any ACK message acknowledging a CON message before RTO expires, the CON message is retransmitted. Retransmission is allowed while the retransmission counter is less than MAX_RETRANSMIT which is usually set to 4.

CoAP is often compared with MQTT (Message Queue Telemetry) [8] which is an ISO standard protocol for constrained resources. While MQTT supports a publish-subscribe communication pattern only, CoAP supports request-response pattern and publish-subscribe pattern both.

3. Congestion Control for CoAP over UDP

This section summarizes the schemes proposed to control congestion in CoAP over UDP.

3.1 Reliable Communication

3.1.1 Basic congestion control of CoAP over UDP

The CoAP was originally designed to enable implementations that do not maintain round-trip-time (RTT) measurements. The basic congestion control of CoAP is carried out by setting RTO(retransmission timeout) without measuring RTT and by limiting the number of outstanding CON messages. For each new CON message, the initial retransmission timeout (RTO) is randomly chosen between ACK_TIMEOUT and ACK_TIMEOUT*ACK_RANDOM_FACTOR. When setting the parameters to the default values, RTO is chosen between 2 and 3seconds. CoAP employs a binary exponential backoff mechanism which doubles RTO whenever a timeout occurs. The number of simultaneous outstanding CON messages or request messages is limited to NSTART for the purpose of avoiding congestion. Table 1 lists the default values for transmission parameters.

Table 1. Default values for transmission parameters

ACK_TIMEOUT	2 seconds
ACK_RANDOM_FACTOR	1.5
MAX_RETRANSMIT	4
NSTART	1

3.1.2 CoCoA

To enhance the basic congestion control mechanism of CoAP, the CORE WG has worked on a congestion control mechanism called CoCoA [6] where several research efforts [9-12] are melted. The core of CoCoA is an RTO algorithm using RTT estimates. The main features of CoCoA are as follows:

- Strong RTO estimator and weak RTO estimator
- Variable Backoff Factor
- RTO aging

1) Strong/Weak RTT estimator

If RTO is too small, retransmissions may be performed too aggressively. Conversely, if RTO is too large, the sender may not be able to perform all retransmissions within the allowed time limit which is MAX_TRANSMIT_WAIT in CoAP. To estimate RTT accurately, TCP uses Karn's algorithm where an RTT sample is not taken from retransmitted TCP segments as specified in RFC 6298 [13]. In contrast, CoCoA uses two kinds of RTO estimators: the strong estimator and the weak estimator. For the strong estimator, RTT is measured only using CON messages which are acknowledged without retransmissions. For

the weak RTO estimator, CoCoA takes an RTT sample even when the ACK is triggered by a retransmitted CON message. While the weak estimator may benefit from the enhanced chance of getting RTT samples [9], it has a drawback that the estimated RTT can be longer than the actual RTT. To avoid taking a much larger RTT estimate than the actual RTT, CoCoA ignores RTT samples obtained after the third retransmission.

In setting RTO with RTT estimates, the variance of measured RTT values is considered. For the strong estimator ("E_{strong}"), the RTT variance multiplier K is set to 4. For the weak estimator, even RTT values consecutively measured may differ substantially from each other, and therefore the RTTVAR value can become very large. Thus, for the weak estimator ("E_{weak}"), K is set to 1 instead of 4 to avoid increasing the RTO excessively. The strong and the weak RTO estimators are calculated as follows.

$$E_{weak} = SRTT + \max(G, RTTVAR)$$

$$E_{strong} = SRTT + \max(G, 4 * RTTVAR)$$

where G is the clock granularity. Whenever the strong or weak estimator is updated, the overall RTO, which is an EWMA(exponentially weighted moving average) of RTO estimator values, is recomputed. To reduce the impact of inaccurate measurements of the weak estimator, CoCoA uses different weights for the recent contribution depending on whether the contribution is from the weak estimator or the strong estimator as follows.

$$RTO = 0.25 \times E_{weak} + 0.75 \times RTO$$

$$RTO = 0.5 \times E_{strong} + 0.5 \times RTO$$

2) Variable backoff factor

The default CoAP doubles RTO whenever the retransmission timer expires and a retransmission occurs. Since the weak estimator of CoCoA may have a larger RTT estimate than the actual one, the binary exponential backoff mechanism (BEB) may cause network underutilization. Conversely, too small RTO results in unnecessary retransmission. To address these problems, CoCoA uses a variable backoff factor (VBF) as follows:

$$VBF(RTO) = \begin{cases} 3, & RTO < a \\ 2, & a \le RTO \le b \\ 1.3, & RTO > b \end{cases}$$

where a and b are the thresholds. CoCoA sets a and b to 1 second and 3 seconds, respectively.

3) RTO aging

To avoid keeping RTO which is not valid any more, CoCoA uses an RTO aging mechanism. If RTO is too short or too long and is not updated for long, CoCoA tries to make RTO move close to the default value. More specifically, CoCoA doubles the current RTO if the RTO value lower than 1 second is not updated for more than 16 times the current RTO. If RTO is larger than 3 seconds, and it is not updated for 4 times its current value, the RTO estimate is set to 1 second + RTO/2. Table 2 summarizes the algorithm for setting RTO in CoCoA.

Table 2. RTO setting algorithm of CoCoA

Event	Action
Initiation	RTO = 2 seconds
Taking an RTT sample from the original transmission	$E_{strong\_}$ = SRTT + max(G, 4*RTTVAR) RTO = 0.5*E_{strong} + 0.5*RTO
Taking an RTT sample from the first retransmission or the second retransmission	E_{weak} = SRTT + max(G, RTTVAR) RTO = 0.25*E_{weak} + 0.75*RTO

Retransmission timer expires	if RTO > 3 seconds then
	RTO = 1.5 * RTO
	else if RTO ≥ 1 second and RTO ≤ 3 seconds then
	RTO = 2 * RTO
	else
	RTO := 3 * RTO

RTO (< 1second) not updated for 16*RTO	RTO := 2*RTO
RTO (> 3 seconds) not updated for 4*RTO	RTO := 1 second + RTO/2

4) Comparison of CoAP and CoCoA

Jarvinen et al. evaluated the congestion control mechanisms for CoAP experimentally [14]. The mechanisms they evaluated are Default CoAP, CoCoA, Basic-RTO, Linux RTO, and Peak Hopper [15]. Basic-RTO algorithm set the RTO value by choosing a random number between RTT and 1.5*RTT. Peak Hopper runs two RTO algorithms in parallel to obtain a short-term history RTO and a long-term history RTO and then takes the maximum of the two RTO values. To observe the effect of congestion only, they emulated error-free links. The results show that the Default CoAP experiences higher median client completion time than RTT-based algorithms. Linux RTO and Peak Hopper have lower median client completion time than CoCoA, because the weak RTO estimator of CoCoA generates longer RTTs as the frequency of retransmissions increases. However, for a large number of clients, the client completion times for Linux RTO and Peak Hopper have long tailed distributions because the strong RTO estimator tends to cause a high number of RTO backoffs for some CON-ACK pairs.

Betzler et al. [16] also conducted experiments to compare Default CoAP, CoCoA, Basic RTO, Linux RTO, Peak Hopper and CoCoA-S which is a CoCoA variant using the strong RTO estimator only. FlockLab that they used involves lossy links and different route lengths, leading to a high RTT variance and a high loss rate. They define a settling time as the time that it takes the clients to finish at least 80 percent of the bursty traffic transactions. TCP-based RTO algorithms show longer settling times on average than the default CoAP and CoCoA since TCP-based algorithms are able to take few RTT measurements due to packet losses in the presence of continuous and bursty traffic both. In their test scenarios, CoCoA outperformed the other schemes in terms of throughput, settling time, retransmission ratio, etc.

Ancillotti et al. compared the congestion control mechanisms between the default CoAP and CoCoA+ via simulation [17]. Their simulation results show that CoCoA+ can perform worse than the default CoAP in scenarios with bursty traffic or small RTTs. According to their analysis, the main reasons are that CoCoA+ initializes RTO values aggressively and that RTT estimates obtained from retransmissions are not reliable.

3.1.3 Other efforts to enhance accuracy of RTT

1) CoCoA 4-state-strong

Bhalerao et al. found that CoCoA underperforms compared to the default CoAP in highly lossy networks [18]. They point out two causes related to the weak estimator of CoCoA. First, inaccurate RTT measurement by the weak estimator leads to disproportionate estimation of RTOs. Second, CoCoA behaves very conservatively in the presence of wireless losses because RTO excessively backs off without distinguishing congestion losses from wireless losses. As the loss rate increases, the throughput of CoCoA decreases.

To address this problem, they propose CoCoA 4-state-strong, a variant of CoCoA. CoCoA 4-state-strong has two features. First, it improves accuracy of RTT measurement by using a retransmission ID or by measuring the time from the latest transmission instead of the initial transmission based on the assumption that a received ACK is not likely from the old (re)transmission particularly when RTO is backed off due to congestion.

Another major feature of CoCoA-4state-strong is that each CoAP transaction is in one of the 4 states depending on the number of retransmissions. The state determines the RTO backoff factor and the weight for calculating EWMA of the overall RTO. It contrasts with CoCoA which uses de facto two states – the strong and weak estimator. The state transition depends on retransmission and acknowledgement as shown in Figure 2. Each transaction starts in state 1. For each transmission, the transaction moves to the next higher state. Conversely, for each acknowledgement, the transaction moves to the next lower state. The higher the state is, the higher the backoff factor is. CoCoA 4-state-strong behaves similarly to CoCoA in that a small backoff factor is used for a high RTO value. However, CoCoA 4-state-strong tries to reduce excessive backoffs in the presence of wireless losses. Assuming that intermittent failures are due to wireless losses, a transaction does not change its state when it sees the first loss. As the frequency of failures increases, more and more weight is assigned to the newly obtained RTO estimate than to the previous overall RTO.

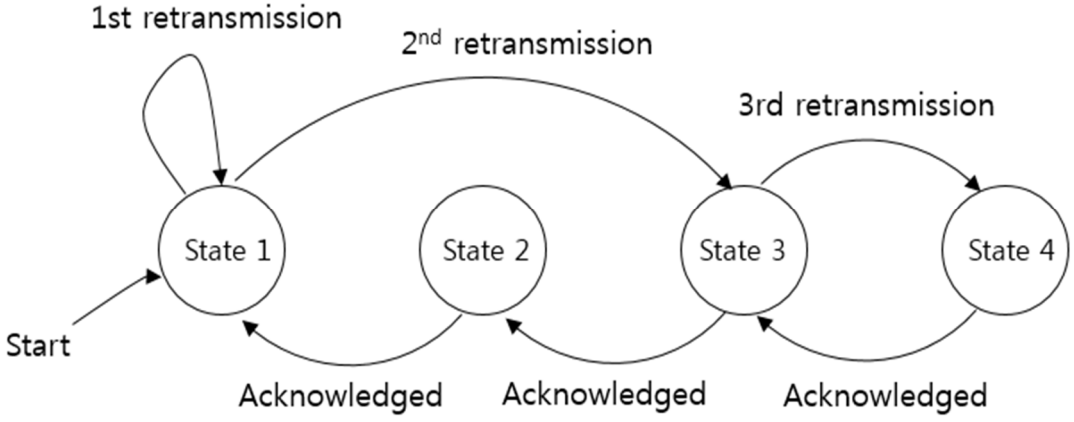

Figure 2. State Transition of CoCoA 4-state-strong

2) Scheme proposed by Lee et al.

Lee et al. also propose using the retransmission count to enhance the accuracy of RTT measurement [19]. They propose putting a lower bound on RTT variation to avoid unnecessary retransmissions due to too a small RTO.

3.1.4 Rate-based congestion control

Ancillotti et al. propose COAP-R which adopts a rate-based approach to regulate the sending rate of CoAP sources [20]. COAP-R assumes that a network has a tree topology rooted at the sink. COAP-R

leverages a property of max-min fair allocation that a rate allocation is max-min fair if and only if every connection has a bottleneck link which is fully utilized or saturated. Thus, the aim of COAP-R is to determine the bottleneck link of each connection and to make the sending rate of the connection obey the bottleneck constraint.

In COAP-R, the bottleneck bandwidth determines an upper bound on the total transmission rate of all the upstream flows that share that link. Each node calculates the aggregate maximum rate demand towards its parent node and sends the sink a CON message containing the information. When a CON message reaches the sink, each node along the path from the source to the sink has the updated information about the maximum rate demand of its subtree. The sink assumes that each source in a subtree demands an equal share of the bottleneck bandwidth and applies a progressive filling algorithm to determine a feasible max-min fair rate allocation. Rate allocation is triggered when the sink transmits an ACK message in reply to a CON message. The Rate option in an ACK message for a node conveys the aggregate rate to the sources in the subtree of the node. Upon receiving an ACK message with a newly allocated rate, a source node controls its sending rate. If the current sending rate is greater than the allocated rate, the source decreases the sending rate for the next transmissions.

3.2 Unreliable Communication

1) Basic congestion control of CoAP over UDP

CoAP communication using NON messages is unreliable since a NON message does not require an acknowledgement. Unreliable communication is appropriate for applications such as repeated readings from a sensor. Another typical use case of NON messages is notification in CoAP with observe. In the extended CoAP to observe resources (RFC 7641), publishers need to send notification messages to subscribers [7]. Whether to use CON messages or NON messages for notification depends on the application requirement. While notification by NON messages would be suitable for resources that change in a predictable or regular fashion, CON messages can be used to send notification for resources that infrequently change.

For NON messages, the basic CoAP does not have any specific congestion control mechanism. However, a couple of basic constraints limiting the sending rate holds regardless of the message type. First, a CoAP endpoint can always send NON messages as far as the sending rate does not exceed 1 byte/second. Second, the number of simultaneous outstanding interaction that are transmitted to a given client to NSTART, which is 1 by default.

In observing resources, a single request by a client (subscriber) can cause a great number of notifications from a server. RFC 7641 suggests controlling congestion of NON messages by its rate-limiting rules [7]. If the server measures RTT, the server can send a client one NON notification per RTT on average. If the server cannot maintain RTT, the sending rate for a client must not exceed one message every 3seconds.

2) CoCoA

CoCoA has the following rules for notifications sent in NON messages. First, if there are not responses or acknowledgements out of 16 messages received by a CoAP endpoint, there should be at least two CON messages. Second, the sending rate of NON messages cannot exceed 1/RTO based on the assumption that RTO estimation works.

3) Back pressure congestion control – network layer approach

Some argue that end-to-end congestion control may not be sufficient in WSNs because data transmission causing congestion does not last long enough to be controlled using end-to-end congestion control mechanisms. Castellani et al. propose a lightweight back pressure (BP) congestion control scheme for CoAP/6LoWPAN networks [21].

They compared four BP variants: IdealBP, Griping, Deaf and Fuse. First, IdealBP restrains the layer 3 device from transmitting as long as the queue length at the nexthop is higher than that of the local queue. The datagram at the current node is transmitted to the nexthop whenever their queue differential is positive and the remote queue length at layer 3 is smaller than a pre-determined threshold. Second, Griping ensures that the time interval between subsequent BP control messages is longer than a given thrshold. Whenever a receiver gets a new packet and the queue length is larger than a threshold, it transmits a BP control message back to the source of the packet. Subsequent BP control messages from the same source must be paced. Upon receiving a BP message, the transmitter halves the sending rate. Third, in Deaf which is a cross-layer approach, a receiver stops sending layer 2 ACKs whenever the layer 3 queue length is larger than a pre-defined threshold. This has an effect of an implicit BP message. A transmitter handles layer 3 retransmission with a binary exponential backoff mechanism. Lastly, Fuse combines Griping and Deaf. In Fuse, a receiver behaves as a Griping node as long as the queue length is smaller than the maximum queue size. If the queue is full, the receiver stops sending layer 2 ACKs and sends explicit BP control messages. A transmitter controls the sending rate in an AIMD manner as in Griping.

In applying the back pressure algorithms to CoAP traffic generated by the request/response model, the authors believe that BP should not slow down response traffic, because any dropped response would be perceived as a loss in the network. Therefore, in the modified BP variants for CoAP, BP is triggered only to push back CoAP requests based on the queue length metric. Their simulation results using NON traffic shows that Deaf performs relatively well for CoAP bidirectional traffic.

4. Discussions

In this section, we suggest the future research directions.

Accurate estimation of RTT and RTO. The majority of studies are focused on how to measure RTT more accurately and how to set RTO adaptively in reliable communication using CON messages. Although CoCoA was proposed to address the drawback of the default CoAP, several studies show that neither the default CoAP nor CoCoA is the winner for all traffic patterns. To enhance accuracy of RTT estimates, some schemes distinguish between an initial transmission and retransmissions by using retransmission id [18,19]. However, since this approach requires additional bookkeeping overhead, further investigation is required.

Differentiation of congestion loss from random loss. This is an issue which has long been studied in the literature of wireless networks. Without distinguishing congestion loss from random loss, a sender may reduce the sending rate unnecessarily. In [18], the authors observed that RTO in CoCoA excessively backs off in a lossy environment without distinguishing congestion loss from wireless loss, and that this leads to a throughput decrease. Since many studies on CoAP have proposed congestion control schemes based on LLNs with no random loss, how to distinguish between congestion loss and random loss needs to be examined further.

Interaction of CoAP with closed-loop network layer protocols. In evaluation of the backpressure congestion control scheme, Castellani et al. used only NON messages [21]. This makes performance

evaluation simple because NON messages do not trigger ACK messages and retransmission. However, when more than one closed-loop protocols are used together, the interaction between the protocols needs to be considered. A study [22] shows that RPL may adversely affect the performance of TCP if RPL does not consider load balancing. This may be the case with CoAP and RPL as well particularly when CON messages are used. How CoAP interacts with RPL is an interesting issue.

5. Conclusions

We reviewed the research literature on congestion control for CoAP over UDP. CoAP has been designed for machine-to-machine communication in resource-constrained environments. Since CoAP is a lightweight protocol compatible with the existing web transfer protocol, the number of CoAP devices is expected to increase rapidly. When CoAP runs over UDP for wireless sensor networks, CoAP itself needs to support congestion control mechanisms. For reliable communication, CoAP performs acknowledgement and retransmission, but the default CoAP only defines a minimal mechanism which sets RTO without estimating RTTs. Thus, the majority of the efforts to improve congestion control have been focused mainly on how to measure RTT more accurately and how to make RTO more adaptive to network conditions. Despite the efforts, further research is required for viable mechanisms to enhance accuracy of RTT with low overhead. In addition, we hope to see new approaches to congestion control for diverse scenarios.

Acknowledgement

This work was supported by 2016 Hongik University Research Fund.

References

[1] Z. Shelby, K. Hartke and C. Bormann, “The Constrained Application Protocol (CoAP),” RFC 7252, June 2014.

[2] M. Iglesias-Urkia, A. Orive, and A. Urbieta, “Analysis of CoAP Implementations for Industrial Intenet of Things: A Survey,” Elsevier Procedia Computer Science, 2016.
DOI: https://doi.org/10.1016/j.procs.2017.05.323

[3] “A Standardized and Flexible IPv6 Architecture for Field Area Networks: Smart-Grid Last-Mile Infrastructure,” Cisco white paper 2014.

[4] The CoAP protocol is the next big thing for DDoS attacks.
https://www.zdnet.com/article/the-coap-protocol-is-the-next-big-thing-for-ddos-attacks/

[5] C. Bormann, S. Lemay, H. Tschofenig, K. Hartke, B. Silverajan, and B. Raymor, “CoAP (Constrained Application Protocol) over TCP, TLS, and Websockets,” RFC 8323, 2017.

[6] C. Bormann, A. Betzler, C. Gomez, and I. Demirkol, “CoAP Simple Congestion Control/Advanced,” draft-ietf-core-ccoa-01, 2017.

[7] K. Hartke, “Observing Resources in the Constrained Application Protocol (CoAP),” RFC 7641, September 2015.

[8] Andrew Bank and Rahul Gupta, MQTT Version 3.1.1 OASIS Standard 29 October 2014.

[9] August Betzler, Carles Gomez, Ilker Demirkol, and Josep Paradells, “CoCoA+: An advanced congestion control for CoAP,” Elsevier Ad Hoc Networks 2015.
DOI: https://doi.org/10.1016/j.adhoc.2015.04.007

[10] August Betzler, Carles Gomez, Ilker Demirkol, and Josep Paradells, “Congestion control in reliable CoAP communication,” ACM MSWiM’13 Proceedings of the 16th ACM international conference on Modeling, analysis & simulation of wireless and mobile systems, 2013.

DOI: 10.1145/2507924.2507954

[11] August Betzler, Carles Gomez, Ilker Demirkol, and Matthias Kovatsch, "Congestion Control for CoAP Cloud Services," IEEE Emerging Technology and Factory Automation (ETFA), 2014.

DOI: 10.1109/ETFA.2014.7005340

[12] August Betzler, Carles Gomez, and Ilker Demirkol, "Evaluation of Advanced Congestion Control Mechanisms for Unreliable CoAP communications," ACM PE-WASUN 2015.

DOI: 10.1145/2810379.2810390

[13] V. Paxson, M. Allman, J. Chu, and M. Sargent, "Computing TCP's Retransmission Timer," RFC 6298, June 2011.

[14] Ilpo Jarvinen, Laila Daniel, and Markku Kojo, "Experimental Evaluation of Alternative Congestion Control Algorithms for Constrained Application Protocol (CoAP)," IEEE 2nd World Forum on Internet of Things (WF-IoT), 2015.

DOI: 10.1109/WF-IoT.2015.7389097

[15] H. Ekstrom and R. Ludwig, "The peak-hopper: a new end-to-end retransmission timer for reliable unicast transport," IEEE Infocom 2004.

DOI: 10.1109/INFCOM.2004.1354671

[16] August Betzler, Carles Gomez, Ilker Demirkol, and Josep Paradells, "CoAP Congestion Control for the Internet of Things", IEEE Communications Magazine, July 2016.

DOI: 10.1109/MCOM.2016.7509394

[17] Emilio Ancillotti, Raffaele Bruno, "Comparison of CoAP and CoCoA+ Congestion Control Mechanisms for Different IoT Application Scenarios," IEEE Symposium on Computers and Communications (ISCC), 2017.

DOI: 10.1109/ISCC.2017.8024686

[18] R. Bhalerao, S. Subramanian, and J. Pasquale, "An Analysis and Improvement of Congestion Control in the CoAP Internet-of-Things Protocol" Proc. IEEE Consumer Communications & Networking Conference (CCNC), Las Vegas, accepted Oct 2015.

DOI: 10.1109/CCNC.2016.7444906

[19] Jung June Lee, Kyung Tae Kim and Hee Yong Youn, "Enhancement of congestion control of Constrained Application Protocol/Congestion Control/Advanced for Internet of Things environment," SAGE International Journal of Distributed Sensor Networks, 2016.

DOI: https://doi.org/10.1177/1550147716676274

[20] Emilio Ancillotti, Raffaele Bruno, Carlo Vallati, and Enzo Mingozzi, "Design and Evaluation of a Rate-Based Congestion Control Mechanism in CoAP for IoT Applications," 2018 IEEE 19th International Symposium on WoWMoM, June 15, 2018.

DOI: 10.1109/WoWMoM.2018.8449736

[21] Angelo P. Castellani, Michele Rossi, and Michele Zorzi, "Back pressure congestion control for CoAP/6LoWPAN networks," Elsevier Ad Hoc Networks 2014.

DOI: https://doi.org/10.1016/j.adhoc.2013.02.007

[22] Hyung-sin Kim, Heesu Im, Myung-Sup Lee, Jeongyeup Paek, and Saewoong Bahk, "A Measurement Study of TCP over RPL in Low-power and Lossy Networks," Journal of Communications and Networks, Vol. 17, No.6, Dec. 2015.

DOI: 10.1109/JCN.2015.000111