

CoRE Working Group
Internet-Draft
Intended status: Informational
Expires: August 25, 2018

C. Bormann
Universitaet Bremen TZI
A. Betzler
Fundacio i2CAT
C. Gomez
I. Demirkol
Universitat Politecnica de Catalunya/Fundacio i2CAT
February 21, 2018

CoAP Simple Congestion Control/Advanced
draft-ietf-core-cocoa-03

Abstract

CoAP, the Constrained Application Protocol, needs to be implemented in such a way that it does not cause persistent congestion on the network it uses. The CoRE CoAP specification defines basic behavior that exhibits low risk of congestion with minimal implementation requirements. It also leaves room for combining the base specification with advanced congestion control mechanisms with higher performance.

This specification defines more advanced, but still simple CoRE Congestion Control mechanisms, called CoCoA. The core of these mechanisms is a Retransmission Timeout (RTO) algorithm that makes use of Round-Trip Time (RTT) estimates, in contrast with how the RTO is determined as per the base CoAP specification ([RFC 7252](#)). The mechanisms defined in this document have relatively low complexity, yet they improve the default CoAP RTO algorithm. The design of the mechanisms in this specification has made use of input from simulations and experiments in real networks.

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."

Internet-Draft

CoAP Simple CoCoA

February 2018

This Internet-Draft will expire on August 25, 2018.

Copyright Notice

Copyright (c) 2018 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

1.	Introduction	3
1.1.	Terminology	3
2.	Context	4
3.	Area of Applicability	4
4.	Advanced CoAP Congestion Control: RTO Estimation	5
4.1.	Blind RTO Estimate	6
4.2.	Measurement-based RTO Estimate	6
4.2.1.	Differences with the algorithm of RFC 6298	7
4.2.2.	Discussion	7
4.3.	Lifetime, Aging	8
5.	Advanced CoAP Congestion Control: Non-Confirmables	9
5.1.	Discussion	9
6.	IANA Considerations	9
7.	Security Considerations	10
8.	References	10
8.1.	Normative References	10
8.2.	Informative References	11
Appendix A.	Supporting evidence	11
A.1.	Older versions of the draft and improvement	12
A.2.	References	12
Appendix B.	Pseudocode	13
B.1.	Updating the RTO estimator	13
B.2.	RTO aging	14
B.3.	Variable Backoff Factor	14

Appendix C.	Examples	15
C.1.	Example A.1: weak RTTs	15
C.2.	Example A.2: VBF and aging	15
C.3.	Example B: VBF and aging	16
Appendix D.	Analysis: difference between strong and weak	

	estimators	16
Acknowledgements	17
Authors' Addresses	17

[1.](#) Introduction

CoAP, the Constrained Application Protocol, needs to be implemented in such a way that it does not cause persistent congestion on the network it uses. The CoRE CoAP specification defines basic behavior that exhibits low risk of congestion with minimal implementation requirements. It also leaves room for combining the base specification with advanced congestion control mechanisms with higher performance.

The present specification defines such an advanced CoRE Congestion Control mechanism, with the goal of improving performance while retaining safety as well as the simplicity that is appropriate for constrained devices. Hence, we are calling this mechanism Simple Congestion Control/Advanced, or CoCoA for short.

CoCoA calculates the retransmission time-out (RTO) based on RTT estimations with and without loss. By taking retransmissions (in a potentially lossy network) into account when estimating the RTT, this algorithm reacts to congestion with a lower sending rate. For non-confirmable packets, it also limits the sending rate to $1/\text{RTO}$; assuming that the RTO estimation in CoCoA works as expected, RTO should be slightly greater than the RTT, thus CoCoA would be more conservative than the original specification in [\[RFC7641\]](#).

In the Internet, congestion control is typically implemented in a way that it can be introduced or upgraded unilaterally. Still, a new congestion control scheme must not be introduced lightly. To ensure that the new scheme is not posing a danger to the network, considerable work has been done on simulations and experiments in real networks. Some of this work will be mentioned in "Discussion" subsections in the following sections; an overview is given in

[Appendix A](#). Extended rationale for this specification can also be found in the historical Internet-Drafts [\[I-D.bormann-core-congestion-control\]](#) and [\[I-D.eggert-core-congestion-control\]](#), as well as in the minutes of the IETF 84 CoRE WG meetings.

[1.1](#). Terminology

This specification uses terms from [\[RFC7252\]](#). In addition, it defines the following terminology:

Bormann, et al.

Expires August 25, 2018

[Page 3]

Internet-Draft

CoAP Simple CoCoA

February 2018

Initiator: The endpoint that sends the message that initiates an exchange. E.g., the party that sends a confirmable message, or a non-confirmable message (see [Section 4.3 of \[RFC7252\]](#)) conveying a request.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [BCP 14 \[RFC2119\]](#) [\[RFC8174\]](#) when, and only when, they appear in all capitals, as shown here.

The term "byte", abbreviated by "B", is used in its now customary sense as a synonym for "octet".

[2](#). Context

In the definition of the CoAP protocol [\[RFC7252\]](#), an approach was taken that includes a very simple basic scheme (lock-step with the number of parallel exchanges usually limited to 1) in the base specification together with performance-enhancing advanced mechanisms.

The present specification is based on the approved text in the [\[RFC7252\]](#) base specification. It is making use of the text that permits advanced congestion control mechanisms and allows them to change protocol parameters, including NSTART and the binary exponential backoff mechanism. Note that [Section 4.8 of \[RFC7252\]](#) limits the leeway that implementations have in changing the CoRE protocol parameters.

The present specification also assumes that, outside of exchanges, non-confirmable messages can only be used at a limited rate without an advanced congestion control mechanism (this is mainly relevant for [\[RFC7641\]](#)). It is also intended to address the [\[RFC8085\]](#) guideline about combining congestion control state for a destination; and to clarify its meaning for CoAP using the definition of an endpoint.

The present specification does not address multicast or dithering beyond basic retransmission dithering.

[3.](#) Area of Applicability

The present algorithm is intended to be generally applicable. The objective is to be "better" than default CoAP congestion control in a number of characteristics, including achievable goodput for a given offered load, latency, and recovery from bursts, while providing more predictable stress to the network and the same level of safety from catastrophic congestion. The algorithm defined in this document is

intended to adapt to the current characteristics of any underlying network, and therefore is well suited for a wide range of network conditions, in terms of bandwidth, latency, load, loss rate, topology, etc. In particular, CoCoA has been found to perform well in scenarios with latencies ranging from the order of milliseconds to peaks of dozens of seconds, as well as in single-hop and multihop topologies. Link technologies used in existing evaluation work comprise IEEE 802.15.4, GPRS, UMTS and Wi-Fi (see [Appendix A](#)). CoCoA is also expected to work suitably across the general Internet. The algorithm does require three state variables per scope plus the state needed to do RTT measurements, so it may not be applicable to the most constrained devices (say, class 1 as per [\[RFC7228\]](#)).

The scope of each instance of the algorithm in the current set of evaluations has been the five-tuple, i.e., CoAP + endpoint (transport address) for Initiator and Responder. Potential applicability to larger scopes needs to be examined.

[4.](#) Advanced CoAP Congestion Control: RTT Estimation

For an initiator that plans to make multiple requests to one destination endpoint, it may be worthwhile to make RTT measurements

in order to compute a more appropriate RTT than the default initial timeout of 2 to 3 s. In particular, a wide spectrum of RTT values is expected in different types of networks where CoAP is used. Those RTTs range from several orders of magnitude below the default initial timeout to values larger than the default. The algorithm defined in this document is based on the algorithm for RTT estimation defined in [\[RFC6298\]](#), with appropriately extended default/base values, as proposed in [Section 4.2.1](#). Note that such a mechanism must, during idle periods, decay RTT estimates that are shorter or longer than the default RTT estimate back to the default RTT estimate, until fresh measurements become available again, as proposed in [Section 4.3](#).

RTT variability challenges RTT estimation. In TCP, delayed ACKs contribute to RTT variability, since this option adds a delay of up to 500 ms (typically, 200 ms) before an ACK is sent by a receiving TCP endpoint. However, one important consideration not relevant for TCP is the fact that a CoAP round-trip may include application processing time, which may be hard to predict, and may differ between different resources available at the same endpoint. Also, for communications with networks of constrained devices that apply radio duty cycling, large and variable round-trip times are likely to be observed. Servers will only trigger their early ACKs (with a non-piggybacked response to be sent later) based on the default timers, e.g. after 1 s. A client that has arrived at a RTT estimate shorter than 1 s SHOULD therefore use a larger backoff factor for retransmissions to avoid expending all of its retransmissions

(MAX_RETRANSMIT, see [Section 4.2 of \[RFC7252\]](#), normally 4) in the default interval of 2 to 3 s. The approach chosen for a mechanism with variable backoff factors is presented in [Section 4.2.1](#).

It may also be worthwhile to perform RTT estimation not just based on information measured from a single destination endpoint, but also based on entire hosts (IP addresses) and/or complete prefixes (e.g., maintain an RTT estimate for a whole /64). The exact way this can be used to reduce the amount of state in an initiator is for further study.

[4.1](#). Blind RTT Estimate

The initial RTT estimate for an endpoint is set to 2 seconds (the initial RTT estimate is used as the initial value for both E_weak_

and E_strong_ below).

If only the initial RT0 estimate is available, the RT0 estimate for each of up to NSTART exchanges started in parallel is set to 2 s times the number of parallel exchanges, e.g. if two exchanges are already running, the initial RT0 estimate for an additional exchange is 6 seconds.

[4.2.](#) Measurement-based RT0 Estimate

The RT0 estimator runs two copies of the algorithm defined in [\[RFC6298\]](#), using the same variables and calculations to estimate the RT0, with the differences introduced in [Section 4.2.1](#): One copy for exchanges that complete on initial transmissions (the "strong estimator", E_strong_), and one copy for exchanges that have run into retransmissions, where only the first two retransmissions are considered (the "weak estimator", E_weak_). For the latter, there is some ambiguity whether a response is based on the initial transmission or the retransmissions. For the purposes of the weak estimator, the time from the initial transmission counts. Responses obtained after the third retransmission are not used to update an estimator.

The overall RT0 estimate is an exponentially weighted moving average computed of the strong and the weak estimator, which is evolved after each contribution to the weak estimator (1) or to the strong estimator (2), from the estimator (either the weak or strong estimator) that made the most recent contribution:

$$\text{RT0} := w_{\text{weak}} * E_{\text{weak_}} + (1 - w_{\text{weak}}) * \text{RT0} \quad (1)$$

$$\text{RT0} := w_{\text{strong}} * E_{\text{strong_}} + (1 - w_{\text{strong}}) * \text{RT0} \quad (2)$$

(Splitting this update into the two cases avoids making the contribution of the weak estimator too big in naturally lossy networks.)

The default values for the corresponding weights, w_weak and w_strong, are 0.25 and 0.5, respectively. These values have been found to offer good performance in evaluations (see [Appendix A](#)). Pseudocode and examples for the overall RT0 estimate presented are

available in [Appendix B.1](#) and [Appendix C.1](#).

[4.2.1](#). Differences with the algorithm of [RFC 6298](#)

This subsection presents three differences of the algorithm defined in this document with the one defined in [[RFC6298](#)]. The first two recommend new parameter settings. The third one is the variable backoff factor (VBF), which replaces [RFC6298](#)'s simple exponential backoff that always multiplies the RT0 by a factor of 2 when the RT0 timer expires.

The initial value for each of the two RT0 estimators is 2 s.

For the weak estimator, the factor K (the RTT variance multiplier) is set to 1 instead of 4. This is necessary to avoid a strong increase of the RT0 in the case that the RTTVAR value is very large, which may be the case if a weak RTT measurement is obtained after one or more retransmissions.

In order to avoid that exchanges with small initial RT0s (i.e. RT0 estimate lower than 1 s) use up all retransmissions in a short interval of time, the RT0 for a retransmission is multiplied by 3 for each retransmission as long as the RT0 is less than 1 s.

On the other hand, to avoid exchanges with large initial RT0s (i.e., RT0 estimate greater than 3 s) not being able to carry out all retransmissions within MAX_TRANSMIT_WAIT (normally 93 s), the RT0 is multiplied only by 1.5 when RT0 is greater than 3 s.

Pseudocode for the variable backoff factor is in [Appendix B.3](#).

The binary exponential backoff is truncated at 32 seconds. Similar to the way retransmissions are handled in the base specification, they are dithered between $1 \times \text{RT0}$ and $\text{ACK_RANDOM_FACTOR} \times \text{RT0}$.

[4.2.2](#). Discussion

In contrast to [[RFC6298](#)], this algorithm attempts to make use of ambiguous information from retransmissions. This is motivated by the high non-congestion loss rates expected in constrained node networks,

and the need to update the RT0 estimators even in the presence of

loss. This approach appears to contravene the mandate in [Section 3.1.1 of \[RFC8085\]](#) that "latency samples MUST NOT be derived from ambiguous transactions". However, those samples are not simply combined into the strong estimator, but are used to correct the limited knowledge that can be gained from the strong RTT measurements by employing an additional weak estimator. In fact, the weak estimator allows to better update the RTO estimator when mostly weak RTTs are available, either due to the lossy nature of links or due to congestion-induced losses. In the presence of the latter, and compared to a strong-only estimator ($w_{\text{weak}}=0$), spurious timeouts are avoided and the rate of retries is reduced, which allows to decrease congestion. Evidence that has been collected from experiments appears to support that the overall effect of using this data in the way described is beneficial (Appendix A).

Some evaluation has been done on earlier versions of this specification [[Betzler2013](#)]. A more recent (and more comprehensive) reference is [[Betzler2015](#)].

[4.3.](#) Lifetime, Aging

The state of the RTO estimators for an endpoint SHOULD be kept as long as possible. If other state is kept for the endpoint (such as a DTLS connection), it is very strongly RECOMMENDED to keep the RTO state alive at least as long as this other state. In the absence of such other state, the RTO state SHOULD be kept at least long enough to avoid frequent returns to inappropriate initial values. For the default parameter set of [Section 4.8 of \[RFC7252\]](#), it is strongly RECOMMENDED to keep it for at least 255 s.

If an estimator has a value that is lower than 1 s, and it is left without further update for 16 times its current value, the RTO estimate is doubled. If an estimator has a value that is higher than 3 s, and it is left without further update for 4 times its current value, the RTO estimate is set to be

$$\text{RTO} := 1 \text{ s} + (0.5 * \text{RTO})$$

(Note that, instead of running a timer, it is possible to implement these RTO aging calculations cumulatively at the time the estimator is used next.)

Pseudocode and examples for the aging mechanism presented are available in [Appendix B.2](#) and in [Appendix C.2](#).

5. Advanced CoAP Congestion Control: Non-Confirmables

A CoAP endpoint MUST NOT send non-confirmables to another CoAP endpoint at a rate higher than defined by this document. Independent of any congestion control mechanisms, a CoAP endpoint can always send non-confirmables if their rate does not exceed 1 B/s.

Non-confirmables that form part of exchanges are governed by the rules for exchanges.

Non-confirmables outside exchanges (e.g., [\[RFC7641\]](#) notifications sent as non-confirmables) are governed by the following rules:

1. Of any 16 consecutive messages towards this endpoint that aren't responses or acknowledgments, at least 2 of the messages must be confirmable.
2. An RT0 as specified in [Section 4](#) must be used for confirmable messages.
3. The packet rate of non-confirmable messages cannot exceed $1/\text{RT0}$, where RT0 is the overall RT0 estimator value at the time the non-confirmable packet is sent.

5.1. Discussion

The mechanism defined above for non-confirmables is relatively conservative. More advanced versions of this algorithm could run a TFRC-style Loss Event Rate calculator [\[RFC5348\]](#) and apply the TCP equation to achieve a higher rate than $1/\text{RT0}$.

[\[RFC7641\]](#), [Section 4.5.1](#), specifies that the rate of Non-Confirmables SHOULD NOT exceed $1/\text{RTT}$ on average, if the server can maintain an RTT estimate for a client. CoCoA limits the packet rate of Non-Confirmables in this situation to $1/\text{RT0}$. Assuming that the RT0 estimation in CoCoA works as expected, $\text{RT0}[k]$ should be slightly greater than the $\text{RTT}[k]$, thus CoCoA would be more conservative. The expectation therefore is that complying with the NON rate set by CoCoA leads to complying with [\[RFC7641\]](#).

6. IANA Considerations

This document makes no requirements on IANA. (This section to be removed by RFC editor.)

7. Security Considerations

The security considerations of, e.g., [RFC5681], [RFC2914], and [RFC8085] apply. Some issues are already discussed in the security considerations of [RFC7252].

If a malicious node manages to prevent the delivery of some packets, a consequence will be an RTT increase, which will further reduce network performance. Note that this type of attack is not specific for CoCoA (and not even specific for CoAP), and many congestion control algorithms increase the RTT upon packet loss detection. While it is hard to prevent radio jamming, some mitigation for other forms of this type of attack is provided by network access control techniques. Also, the weak estimator in CoCoA increases the chances of obtaining RTT measurements in the presence of heavy packet losses, allowing to keep the RTT updated, which in turn allows recovery from a jamming attack in reasonable time.

8. References

8.1. 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>>.
- [RFC2914] Floyd, S., "Congestion Control Principles", [BCP 41](#), [RFC 2914](#), DOI 10.17487/RFC2914, September 2000, <<https://www.rfc-editor.org/info/rfc2914>>.
- [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>>.
- [RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", [RFC 7252](#), DOI 10.17487/RFC7252, June 2014,

<<https://www.rfc-editor.org/info/rfc7252>>.

- [RFC8085] Eggert, L., Fairhurst, G., and G. Shepherd, "UDP Usage Guidelines", [BCP 145](#), [RFC 8085](#), DOI 10.17487/RFC8085, March 2017, <<https://www.rfc-editor.org/info/rfc8085>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in [RFC 2119](#) Key Words", [BCP 14](#), [RFC 8174](#), DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

Bormann, et al.

Expires August 25, 2018

[Page 10]

Internet-Draft

CoAP Simple CoCoA

February 2018

[8.2](#). Informative References

- [Betzler2013]
Betzler, A., Gomez, C., Demirkol, I., and J. Paradells, "Congestion control in reliable CoAP communication", ACM MSWIM'13 p. 365-372, DOI 10.1145/2507924.2507954, 2013.
- [Betzler2015]
Betzler, A., Gomez, C., Demirkol, I., and J. Paradells, "CoCoA+: an Advanced Congestion Control Mechanism for CoAP", Ad Hoc Networks Vol. 33 pp. 126-139, DOI 10.1016/j.adhoc.2015.04.007, October 2015.
- [I-D.bormann-core-congestion-control]
Bormann, C. and K. Hartke, "Congestion Control Principles for CoAP", [draft-bormann-core-congestion-control-02](#) (work in progress), July 2012.
- [I-D.eggert-core-congestion-control]
Eggert, L., "Congestion Control for the Constrained Application Protocol (CoAP)", [draft-eggert-core-congestion-control-01](#) (work in progress), January 2011.
- [RFC5348] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", [RFC 5348](#), DOI 10.17487/RFC5348, September 2008, <<https://www.rfc-editor.org/info/rfc5348>>.
- [RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", [RFC 5681](#), DOI 10.17487/RFC5681, September 2009, <<https://www.rfc-editor.org/info/rfc5681>>.

- [RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", [RFC 7228](#), DOI 10.17487/RFC7228, May 2014, <<https://www.rfc-editor.org/info/rfc7228>>.
- [RFC7641] Hartke, K., "Observing Resources in the Constrained Application Protocol (CoAP)", [RFC 7641](#), DOI 10.17487/RFC7641, September 2015, <<https://www.rfc-editor.org/info/rfc7641>>.

[Appendix A](#). Supporting evidence

(Editor's note: The references local to this appendix may need to be merged with those from the specification proper, depending on the discretion of the RFC editor.)

Bormann, et al.

Expires August 25, 2018

[Page 11]

Internet-Draft

CoAP Simple CoCoA

February 2018

CoCoA has been evaluated by means of simulation and experimentation in diverse scenarios comprising different link layer technologies, network topologies, traffic patterns and device classes. The main overall evaluation result is that CoCoA consistently delivers a performance which is better than, or at least similar to, that of default CoAP congestion control. While the latter is insensitive to network conditions, CoCoA is adaptive and makes good use of RTT samples.

It has been shown over real GPRS and IEEE 802.15.4 mesh network testbeds that in these settings, in comparison to default CoAP, CoCoA increases throughput and reduces the time it takes for a network to process traffic bursts, while not sacrificing fairness. In contrast, other RTT-sensitive approaches such as Linux-RT0 or Peak-Hopper-RT0 may be too simple or do not adapt well to IoT scenarios, underperforming default CoAP under certain conditions [1]. On the other hand, CoCoA has been found to reduce latency in GPRS and WiFi setups, compared with default CoAP [2].

CoCoA performance has also been evaluated for non-confirmable traffic over emulated GPRS/UMTS links and over a real IEEE 802.15.4 mesh testbed. Results show that since CoCoA is adaptive, it yields better packet delivery ratio than default CoAP (which does not apply congestion control to non-confirmable messages) or Observe (which introduces congestion control that is not adaptive to network

conditions) [3, 4].

[A.1.](#) Older versions of the draft and improvement

CoCoA has evolved since its initial draft version. Its core has remained mostly stable since [draft-bormann-core-cocoa-02](#). The evolution of CoCoA has been driven by research work. This process, including evaluations of early versions of CoCoA, as well as improvement proposals that were finally incorporated in CoCoA, is reflected in published works [5-10].

[A.2.](#) References

- [1] A. Betzler, C. Gomez, I. Demirkol, J. Paradells, "CoAP congestion control for the Internet of Things", IEEE Communications Magazine, July 2016.
- [2] F. Zheng, B. Fu, Z. Cao, "CoAP Latency Evaluation", [draft-zheng-core-coap-lantency-evaluation-00](#), 2016 (work in progress).
- [3] A. Betzler, C. Gomez, I. Demirkol, "Evaluation of Advanced Congestion Control Mechanisms for Unreliable CoAP Communications", PE-WASUN, Cancun, Mexico, 2015.

- [4] A. Betzler, J. Isern, C. Gomez, I. Demirkol, J. Paradells, "Experimental Evaluation of Congestion Control for CoAP Communications without End-to-End Reliability", Ad Hoc Networks, Volume 52, 1 December 2016, Pages 183-194.
- [5] A. Betzler, C. Gomez, I. Demirkol, J. Paradells, "Congestion Control in Reliable CoAP Communication", 16th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM'13), Barcelona, Spain, Nov. 2013.
- [6] A. Betzler, C. Gomez, I. Demirkol, M. Kovatsch, "Congestion Control for CoAP cloud services", 8th International Workshop on Service-Oriented Cyber-Physical Systems in Converging Networked Environments (SOCNE) 2014, Barcelona, Spain, Sept. 2014.
- [7] A. Betzler, C. Gomez, I. Demirkol, J. Paradells, "CoCoA+: an advanced congestion control mechanism for CoAP", Ad Hoc Networks journal, 2015.

[8] Bhalerao, Rahul, Sridhar Srinivasa Subramanian, and Joseph Pasquale. "An analysis and improvement of congestion control in the CoAP Internet-of-Things protocol." 2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC). IEEE, 2016.

[9] I Jaervinen, L Daniel, M Kojo, "Experimental evaluation of alternative congestion control algorithms for Constrained Application Protocol (CoAP)", IEEE 2nd World Forum on Internet of Things (WF-IoT), 2015.

[10] Balandina, Ekaterina, Yevgeni Koucheryavy, and Andrei Gurtov. "Computing the retransmission timeout in coap." Internet of Things, Smart Spaces, and Next Generation Networking. Springer Berlin Heidelberg, 2013. 352-362.

[Appendix B](#). Pseudocode

[B.1](#). Updating the RT0 estimator

```
// Default values
ALPHA = 0.125 // RFC 6298
BETA = 0.25 // RFC 6298
W_STRONG = 0.5
W_WEAK = 0.25

updateRT0(retransmissions, RTT) {
  if (retransmissions == 0) {
    RTTVAR_strong = (1 - BETA) * RTTVAR_strong
                  + BETA * (RTT_strong - RTT);
    RTT_strong = (1 - ALPHA) * RTT_strong + ALPHA * RTT;
```

```

    E_strong = RTT_strong + 4 * RTTVAR_strong;
    RTO = W_STRONG * E_strong + (1 - W_STRONG) * RTO;
} else if (retransmissions <= 2) {
    RTTVAR_weak = (1 - BETA) * RTTVAR_weak
        + BETA * (RTT_weak - RTT);
    RTT_weak = (1 - ALPHA) * RTT_weak + ALPHA * RTT;
    E_weak = RTT_weak + 1 * RTTVAR_weak;
    RTO = W_WEAK * E_weak + (1 - W_WEAK) * RTO
}
}

```

[B.2.](#) RTO aging

```

checkAging() {
    clock_time difference = getCurrentTime() - lastUpdatedTime;

    if ((RTO < 1s) && (difference > (16 * RTO))) {
        RTO = 2 * RTO;
        lastUpdatedTime = getCurrentTime();
    } else if ((RTO > 3s) && (difference > (4 * RTO))) {
        RTO = 1s + 0.5 * RTO;
        lastUpdatedTime = getCurrentTime();
    }
}

```

[B.3.](#) Variable Backoff Factor

```

backOffRTO() {
    if (RTO < 1s) {
        RTO = RTO * 3;
    } else if (RTO > 3s) {
        RTO = RTO * 1.5;
    } else {
        RTO = RTO * 2;
    }
}

```

[Appendix C.](#) Examples

[C.1.](#) Example A.1: weak RTTs

A large network of sensor nodes that report periodical measurements is operating normally, without congestion. The nodes transmit their sensor readings via CON messages every 20 s in an asynchronous way towards a server located behind a gateway, obtaining strong RTT measurements (RTT 1.1 s, RTTVAR 0.1 s) that lead to the calculation of an RT0 of 1.5 s (in average) in each node. In this mode of operation, no aging is applied, since the RT0 is refreshed before the aging mechanism applies.

Suddenly, upon detection of a global event, the majority of sensor nodes start transmitting at a higher rate (every 5 s) to increase the resolution of the acquired data, which creates heavy congestion that leads to packet losses and an important increase of real RTT between the nodes and the server (RTT 2 s, RTTVAR 1 s). Due to the packet losses and spurious retransmissions (which can fuel congestion even more), many nodes are not able to update their RT0 via strong RTT measurements, but they are able to obtain weak RTT measurements. A node with an initial RT0 of 1.5 s would run into a retransmission, before obtaining an ACK (given the RTT of 2 s and that the ACK is not lost).

This weak RTT measurement would increase the overall RT0 of the node to 1.875 s ($RT0 = 0.25 * 3 \text{ s} + 0.75 * 1.5 \text{ s}$). Following the same calculus (and RTT/RTTVAR values), after obtaining another weak RTT, the RT0 would increase to 2.156 s. At this point, the benefits of the weak RTT measurements are twofold:

1. Further spurious retransmissions are avoided as the RT0 has increased above the real RTT.
2. The increase of RT0s across the whole network reduces the rate with which retransmissions are generated, decreasing the network congestion (which leads to an RTT and packet loss decrease).

[C.2.](#) Example A.2: VBF and aging

Assuming that the frequency of message generation is even higher (every 3 s) and the real RTT would further increase due to congestion, the RT0 at some point would increase to 4 s. Since now the RT0 is above 3 s, no longer a binary backoff is used to avoid the RT0 growing too much in case of retransmissions. As the generation of data from the nodes ceases at some point (the network returns to a normal state), the aging mechanism would reduce the RT0 automatically

(with an RTT of 4 s, after 16 s the RTT would be shifted to 3 s before a new RTT is measured).

[C.3.](#) Example B: VBF and aging

A network of nodes connected over 4G with an Internet service is calculating very small RTT values (0.3 s) and the nodes are transmitting CON messages every 1 s. Suddenly, the connection quality gets worse and the nodes switch to a more stable, yet slower connection via GPRS. As a result of this change, the nodes run into retransmissions, as the real RTT has increased above the calculated RTT.

Since the RTT is below 1 s, the Variable Backoff Factor increases the backoff values quickly to avoid spurious retransmissions (0.9 s first retry, 2.7 s second retry, etc.). Further, if due to the packet losses and increased delays in the network no new RTT measurements are obtained, the aging mechanism automatically increases the RTT (doubling it) after 3.8 s ($16 * 0.3$ s) to adapt better to the sudden changes of network conditions. Without the Variable Backoff Factor and the aging mechanism, the number of spurious retransmissions would be much higher and the RTT would be corrected more slowly.

[Appendix D.](#) Analysis: difference between strong and weak estimators

This section analyzes the difference between the strong and weak RTT estimators. If there is no congestion, assume a static RTT of R' . Then, $E_strong_$ can be expressed as:

$$E_strong_ = R' + G,$$

since RTTVAR is reduced constantly by $RTTVAR = RTTVAR * 3/4$ (according to [\[RFC6298\]](#), and $SRTT=R'$), G would be dominant term in the $\max(G, K * RTTVAR)$ expression in the long run.

For the weak estimator: assume that the RTT setting converges to $E_strong_$ calculated above in the long run. If there is a packet loss, and an RTT is obtained for the first retransmission, then the weak RTT sample obtained by the weak estimator is:

$$RW' = R' + G + R'$$

Therefore, $E_weak_$ can be expressed as:

$$E_weak_ = RW' + \max(G, RW'/2) = 3 * R'$$

Internet-Draft

CoAP Simple CoCoA

February 2018

Acknowledgements

The first document to examine CoAP congestion control issues in detail was [[I-D.eggert-core-congestion-control](#)], to which this draft owes a lot.

Michael Scharf did a review of CoAP congestion control issues that asked a lot of good questions. Several Transport Area representatives made further significant inputs this discussion during IETF84, including Lars Eggert, Michael Scharf, and David Black. Andrew McGregor, Eric Rescorla, Richard Kelsey, Ed Beroaset, Jari Arkko, Zach Shelby, Matthias Kovatsch and many others provided very useful additions. Further reviews by Michael Scharf and Ingemar Johansson led to further improvements, including some more discussion in the appendices.

Authors from Universitat Politecnica de Catalunya have been supported in part by the Spanish Government's Ministerio de Economia y Competitividad through projects TEC2009-11453, TEC2012-32531, TEC2016-79988-P and FEDER.

Carles Gomez has been funded in part by the Spanish Government (Ministerio de Educacion, Cultura y Deporte) through the Jose Castillejo grant CAS15/00336. His contribution to this work has been carried out in part during his stay as a visiting scholar at the Computer Laboratory of the University of Cambridge, in collaboration with Prof. Jon Crowcroft.

Authors' Addresses

Carsten Bormann
Universitaet Bremen TZI
Postfach 330440
Bremen D-28359
Germany

Phone: +49-421-218-63921
Email: cabo@tzi.org

August Betzler
Fundacio i2CAT
Mobile and Wireless Internet Group
C/ del Gran Capita, 2
Barcelona 08034
Spain

Email: august.betzler@i2cat.net

Bormann, et al.

Expires August 25, 2018

[Page 17]

Internet-Draft

CoAP Simple CoCoA

February 2018

Carles Gomez
Universitat Politecnica de Catalunya/Fundacio i2CAT
Escola d'Enginyeria de Telecomunicacio i Aeroespacial
de Castelldefels
C/Esteve Terradas, 7
Castelldefels 08860
Spain

Phone: +34-93-413-7206
Email: carlesgo@entel.upc.edu

Ilker Demirkol
Universitat Politecnica de Catalunya/Fundacio i2CAT
Departament d'Enginyeria Telematica
C/Jordi Girona, 1-3
Barcelona 08034
Spain

Email: ilker.demirkol@entel.upc.edu

