International Journal of Advanced Engineering Research and Science (IJAERS)
Peer-Reviewed Journal
ISSN: 2349-6495(P) | 2456-1908(O)
Vol-8, Issue-10; Oct, 2021
Journal Home Page Available: https://ijaers.com/
Article DOI: https://dx.doi.org/10.22161/ijaers.810.18

Congestion Control in Unicast and Multicast CoAP-based Communications

Fathia Ouakasse, Said Rakrak

Cadi Ayyad University, Faculty of Sciences and Techniques, Laboratory of Computer and Systems Engineering (L2IS), Marrakesh, Morocco

Received: 07 Sep 2021,

Received in revised form: 08 Oct 2021,

Accepted: 14 Oct 2021,

Available online: 20 Oct 2021

***Keywords*— CoAP, Congestion control, Algorithm, Unicast, Multicast.**

***Abstract*—** *Lightweight devices and constrained resources used in the Internet of Things (IoT) applications have developed in exorbitant numbers, generating a large amount of data required for intelligent data processing. One of the foremost emerging messaging protocols used to address the requirements of these lightweight IoT nodes is Constrained Application Protocol (CoAP). Considering the unlimited number of messages and notifications generated by these devices, the problem of congestion occurs in CoAP communications. In this context, to fulfill successfully the need of transactions and succeed to handle reliably unicast and multicast communications, CoAP dispose of congestion control mechanisms to manage both unicast and multicast communications. The challenge addressed in this paper consists of designing appropriate congestion control mechanisms for CoAP that ensures a secure network operation while keeping the utilization of network resources efficiently. Therefore, in this paper, we shed the light on congestion control algorithms used to manage unicast and multicast communications CoAP based; we present a critical analysis of its performances and highlight some of its shortcomings and pitfalls. We combine and put forward two of our proposed adaptive algorithms of congestion control based on network conditions in both unicast and multicast communications.*

I. INTRODUCTION

The IoT has invaded all the sides of our life and plenty of new applications are emerging in different fields of our environment. Furthermore, more and more technologies related to the IoT are opened out to the benefit of improving the quality of life. Nevertheless, although the fast progress of IoT and related technologies, the need for suitable and appropriate solutions involving constrained devices is necessary to overcome the problem persisting like network congestion.

The Constrained Application Protocol CoAP has been designed by the Internet Engineering Task Force (IETF) to support IoT with lightweight messaging for devices operating in a constrained environment. It defines two interactions types between end-points based on client/server model: a) One-to-one interaction (request/reply) and b) Multi-cast interaction (Client wants to interrogates multiple servers). Like HTTP, Clients have the power to manage resources using requests: GET, PUT, POST and DELETE to perform Create, Retrieve, Update, and Delete operations.

CoAP has successfully fulfilled the need of the lightweight features required to handle communication between constrained devices in IoT environment. However, these devices are generating an enormous

amount of messages and notifications that cause the network congestion.

Network congestion in CoAP represents the great limitation that hinders the right functioning of this protocol and causes the loss of packets. It can also significantly damage the performance of a network, manifesting in increased packet latencies, while a network may even become useless if the congestion collapse occurs [1]. On the other hand, unlike HTTP, CoAP does not run over TCP, it runs over UDP.

Consequently, the challenge addressed in CoAP based communication consists of designing a suitable congestion control mechanism that ensures a safe network operation while keeping the use of network resources efficiently.

Communication between clients and servers is afforded through connectionless datagrams. Retries and reordering are implemented within the application stack. So, retransmissions management is a challenge in CoAP. CoAP also allows UDP broadcasts and multicasts for addressing [2]. In this paper, we discuss two communication types Unicast and Multicast CoAP based communications.

In unicast communications, CoAP allows communications between single nodes (client to server or server to client). So, it must compute an adequate Retransmission Time Out (RTO) for the next transmission for each node. In this context, the core CoAP specification defines a basic congestion control mechanism that consists of the utilization of a back off mechanism to compute the Retransmission Time Out (RTO) for subsequent transmission. It is based on the use of a fixed RTO value that is doubled in each retransmission. Nevertheless, albeit the core CoAP specification defines a basic congestion control mechanism to make it ready to handle congestion control by itself, researches have proved that CoAP is not capable of being adaptive to network conditions.

To address all those aforementioned problems, many congestion control algorithms were proposed. In this paper we present the most important ones, we compare them and we highlight our own adaptive solutions.

On the other hand, in many IoT application fields, additionally to unicast communication, nodes should be addressed in groups, so as to manage the requirements of multiple communications between different and several devices, CoAP supports group communication [3]. Multicast communication is a communication driven from a single node to multiples nodes (client to servers or server to clients). However, CoAP ensures multicast communication in one sense, from a server to multiple clients, but in the other sense -from a client to multiple servers- , it relies on unicast communications. This has led to the problem of network congestion [4]. To resolve this problem, researchers propose to insert a delay between consecutive requests.

In addition to a review of congestion control algorithms in multicast CoAP based communication, we discuss our own proposition of the delay to insert between two requests.

The remainder of this paper is organized as follow: A description of CoAP congestion control algorithms in unicast communication is presented as well as our proposed adaptive algorithm in the second section. Then, in the third section, related works to group communication in CoAP including multicast group communications are described. Afterwards, the proposed improved congestion control algorithm is detailed. Finally, a conclusion and some future directions are closing up our paper in the fourth section.

II. COAP CONGESTION CONTROL IN UNICAST COMMUNICATIONS

2.1. Basic CoAP congestion control algorithm

Although, CoAP has become increasingly proposed and proved its effectiveness in gathering data from smart sensors and controlling constrained devices, the problem of network congestion in CoAP represents the great limitation that causes the packets loss.

Indeed, the core CoAP specification defines a basic congestion control mechanism that consists of the use of a backoff mechanism to compute the Retransmission Time Out (RTO) for the next transmission. It is based on the use of a fixed RTO value, which is doubled in each retransmission. Nevertheless, even if the core CoAP specification defines a basic congestion control mechanism to make it able to handle congestion control by itself, researches have proved that CoAP is not capable of being adaptive to network conditions and this goes back to the fact that it doesn't take into consideration the RTT of a packet since the network conditions may change frequently because of the dynamic topology and the density of WSN nodes [5]. Therefore, the calculation of an appropriate RTO is essential to overcome the problem of congestion.

2.2. Classical TCP congestion control algorithm

Unlike the basic CoAP congestion control that uses a fixed RTO, the computation of RTO in the classical TCP congestion control is based on the history variation of RTT. The specification of this algorithm is proposed by RFC 6298 [6].

According to this specification, the calculation of the actual RTO to use in the next transmission is based on two variables; smoothed average of RTT (SRTT) and RTT variation (RTTVAR); where SRTT is used to preserve the history of RTT and RTTVAR keeps the history of RTT variation. Both of these parameters are constant and their impact factors respectively are 7/8 and 3/4.

Initially, the RTO value is set to 1 second. After the first transmission, the first RTT value is received. To compute the following RTO value, SRTT is set as RTT received and RTTVAR is set as RTTreceived/2, the following formulas are used [7]:

SRTT = RTT

RTTVAR = RTT/2

After subsequent RTT measurements are received, the following formulas are applied:

$$SRTT = (1 - \alpha) * SRTT + \alpha * RTT \quad (1)$$

$$RTTVAR = (1 - \beta) * RTTVAR + \beta * |SRTT - RTT| \quad (2)$$

$$RTO = SRTT + \max (G, K*RTTVAR) \quad (3)$$

The formula (3) is used to estimate the RTO value to be used in the following transmission. When the RTO timer expires, the RTO value is doubled [8].

According to RFC 2988, the value of the constant K in (3) is 4. Furthermore in (1) and (2) α and β are also constants and their values respectively are 1/8, 1/4.

Moreover, the G value defines the clock granularity in seconds and according to experiences, finer clock granularities inferior or equal to 100 ms perform somewhat better than other granularity values [8]. Thus, it is recommended to choose the G value not greater than 100 ms [6]. At the same time, G should be at least one order of magnitude smaller than the RTT [9].

2.3. CoCoA

Since the basic CoAP congestion control mechanism can hardly meet the requirements of many IoT applications, several approaches were proposed to improve the aforementioned CoAP shortcomings. The CoAP Simple Congestion Control/Advanced CoCoA [10] is the most important extension of CoAP that has been standardized by IETF. Indeed, basic CoAP does not care about the network characteristics; it behaves the same way in any type of network. Therefore, in order to optimize the CoAP congestion control abilities, CoCoA, based on TCP RTO estimation algorithm, uses RTT measurements to add state information about individual RTOs for different destination endpoints. CoCoA algorithm is based on two mechanisms: (i) a strong estimator; the packet is received on the initial transmission without any retransmissions and (ii) a weak estimator; RTT value is measured after at most two retransmissions. Both of these mechanisms implement the same algorithm but have different sets of state variables. The overall RTO estimated in the formula (6) is made from the estimator that made the most recent contribution using either formula (4) or formula (5) [9].

$$RTO_{recent} = 0.25 * RTO_{weak} + 0.75 * RTO_{recent} \quad (4)$$

$$RTO_{recent} = 0.5 * RTO_{strong} + 0.5 * RTO_{recent} \quad (5)$$

$$RTO_{overall} = 0.5 * RTO_{recent} + 0.5 * RTO_{overall} \quad (6)$$

The fact that CoCoA uses a constant backoff factor and RTO aging mechanism penalize its performances. The reason why authors in [11] propose a mechanism using a variable backoff factor depending on the estimated RTO called CoCoA+. The improvement in this mechanism helps to avoid quick retransmissions for low RTO values, and to avoid slow retransmissions for large RTO values. In addition, in the case when the RTO value has not been updated for a long time, CoCoA+ adds an incorporated RTO aging mechanism.

Furthermore, in [12], the authors design a 4-state estimator scheme for CoCoA depending on the number of times a packet has been retransmitted. The transaction starts in state 1, and each time a packet is retransmitted, its state increases by one. Each time a packet is successfully transmitted and acknowledged within its stipulated time, its state decreases by one. This allows setting the backoff parameters accordingly.

Nevertheless, in the presence of a high number of packet losses, subsequent updates of the weak RTO estimator can cause some unexpected or unpredictable problems [13]. Since CoAP limits the Max_Retransmit in four, a new RTT_{weak} might be obtained after the second, third, fourth, or fifth transmission. Thus, the specification of correspondence between each transmission and its CoAP acknowledgment might be not possible. This mechanism may have a great impact on the calculation of the overall RTO. In addition, when the RTT_{weak} is measured after multiple retransmissions, the new calculated RTO might increase in a considerable way compared to RTO_{init}.

So, in the aforementioned congestion control mechanisms, the issue of setting a right RTO value with burst traffic is still limited because setting a correct and an accurate RTT of retransmitted packet is hardly obtained [14]. In table 1, CoAP congestion control algorithms are listed with basic characteristics.

Table .1: CoAP Congestion control algorithm's characteristics

Algorithm	Backoff method	RTT estimation	RTO aging	Derived from
CoAP	BBF	None	No	None
CoCoA	VBF	Strong & weak	Yes	Linux RTO
CoCoA-S	VBF	Strong & weak	Yes	CoCoA
CoCoA-E	VBF	Strong & weak	Yes	CoCoA & Eifel
4-state-strong	VBF	Four estimations	Yes	CoCoA

III. COAP CONGESTION CONTROL IN MULTICAST COMMUNICATIONS

In the IoT, applications use group communication to make transactions between its different nodes, this goes back to the fact that nodes should be addressed either individually or in groups.

In many IoT applications, nodes addressed in group, i.e., a one to many communication patterns is essential to meet the needs of the application. Furthermore, in some applications, to increase the accuracy and the reliability of gathered data, it is important to collect information from more than one sensor. Moreover, the information gathered at the same time from many sensors may be very crucial to decide the appropriate way to intervene in situations that require real time intervention. So, all these scenarios and others require a communication with a group of sensors as recognized in the Charter of IETF CoRE Working Group [15].

IETF CoRE working group has first recognized the need to support a non-reliable multicast message. Thus, they have developed a specification for Group Communication for CoAP in RFC 7390 [16] to explain how we can use the CoAP protocol in a group communication context. Indeed, Group communication based on CoAP consists of sending a single non-confirmable message to multiple nodes grouped into a specific group using UDP/IP multicast for the requests, and unicast UDP/IP for the responses (if there was any). This means that all the nodes grouped in this group receive the same exact message [17].

It was proved that the use of multicast communication for sending requests is very efficient but it does not affect the number of responses sent by the destination nodes since these are sent as unicasts. In the same context, authors in [18] presented an alternative lightweight forwarding algorithm for efficient multicast support in Low-power and Lossy Networks (LLNs). This allows reducing a number of requests in the LLN since it sends one request to multiple destinations at the same time instead of a unicast for each destination.

The problem of congestion happens when the traffic load offered to a network approaches the network capacity [19]. This phenomenon is one of the main obstacles that still hinder the well functioning of many protocols and thus impacts directly the efficiency of the communication. On the other hand, requests in group communication using CoAP engender a multitude of responses from different nodes, potentially causing congestion. Therefore, both the group communication multicast-based requests and the group communication CoAP unicast-based responses to these multicast requests must be conservatively controlled.

Indeed, it defines a random delay called leisure that consists of a period of time delay inserted between multiple multicast requests. This leisure could be a default value either used by the server or computed according to formula (7).

$$Leisure = S*G/R \quad (7)$$

Where, G is an estimated group size, R is a target data transfer rate R and S is an estimated response size.

Nevertheless, in the case when a single client is communicating with multiple servers using unicasts, CoAP does not specify a congestion control mechanism. To overcome this situation, authors in [20] proposed a simple solution consisting of a delay inserted between consecutive requests; this led to a limitation in the rate at which requests are sent.

In the following paper, we propose an improved formula to calculate the estimated delay to introduce between requests in order to reduce the network congestion.

IV. RESULTS AND DISCUSSION

4.1. Adaptive unicast congestion control algorithm

In addition to the fact that the basic CoAP congestion control does not use the RTT of previous transactions to estimate the following RTO, it also does not take into consideration the utility of the packet loss ratio as well, to adapt its behavior to network conditions.

The packet loss is defined as the number of packets that failed to reach their destination across a computer network. Packet loss is caused either by link-layer interference or by network congestion and it is measured as a percentage of packets lost according to packets sent.

Indeed, in IoT networks, the packet loss is considered one of the big consequences of the network congestion problem. Based on this fact, in [20], we propose an

improved congestion control algorithm based on the packet loss ratio and the RTT value considered in the previous transmission.

Furthermore, in order to provide an adaptive dynamic retransmission timeout that can be suitable for network conditions in the IoT applications, we propose to update the RTO value in each retransmission according to the packet loss ratio. The correlation between actual and previous RTO values seems primordial to adapt the recent RTO value to network conditions. Basically, the RTT and the packet loss ratio which is in a frequent change. Therefore, there is no need to aging techniques because our RTO is in a frequent change according to the packet loss changes and it will never keep a fixed value for an extended period. Thus, the server notifies the client with the packet loss ratio based on sequence numbers of received messages i.e. when the server receives a message, it gets a set of sequence numbers and it recognizes the sequence numbers missed then it calculates a packet loss percentage according to packets sent. In our conception, two scenarios are proposed; (i) if the packet loss ratio is lower than 50%, the RTO value will be updated according to formula (7) in order to prevent unnecessarily long idle time, otherwise, (ii) if the packet loss ratio exceeds 50%, the RTO will be updated in order to correct the loss according to formula (8). In other words, when the packet loss has a low value (pl < 0.5), we conserve nearly the same RTO value as the previous value ($RTO_{recent} \approx RTO_{previous}$), this is in order to reduce idle time (waiting time). On the other hand, when the pl increases (pl > 0.5) the RTO conserve as well nearly the same value as the previous value, this is in order to correct the loss of packets. These formulas aim to adapt the RTO calculation to network conditions (RTT and packet loss) by conserving nearly the same value of the retransmission timeout.

Initially, like the basic CoAP specification [15], we initiate the RTO to a random value between Ack_TimeOut and Ack_TimeOut*Ack_Random_Factor. Once the RTO is initiated, a message is sent to the corresponding client. Then after the reception of the message, the receiver calculates the RTT and the packet loss values based on the received packets and the sequence numbers. Afterward, the formulas (8) and (8) are used in order to calculate the following RTO to use in the next retransmission.

$$RTO_{recent} = RTT * packet\_loss\_ratio + (1-packet\_loss\_ratio) * RTO_{previous} \quad (8)$$

$$RTO_{recent} = RTO_{previous} * packet\_loss\_ratio + (1-packet\_loss\_ratio) * RTT \quad (9)$$

The detailed algorithm of our proposition is drawn in Figure 1.

```
RTO_init=random(ACK_TIMEOUT,ACK_TIMEOUT*ACK_RANDOM_FACTOR)

RTO = RTO_init

For (i=1; i<packet.packet_nbr ; i++)
        send_coap_packet ( )
        packet.pl=calculate_packet_loss ( )
        packet.rtt=calculate_rtt ( )
        receive_coap_ack ( )

        if (pl<0.5)
                then
                RTO=RTO*(1-packet.pl) + packet.rtt*packet.pl
                else
                RTO=RTO*packet.pl + (1-packet.pl)*packet.rtt
        endif
endfor
```

Fig. 1: Pseudo code of the proposed algorithm

This proposition presents a dynamic and controlled retransmission timeout adapted to be appropriate and suitable for the IoT communications particularities. The two mechanisms presented by our proposition effectively limit the growth of RTO values since it is the previous RTO that makes the higher weight in each of formula (8) and (9).

4.2. Adaptive multicast congestion control algorithm

Experiences show that communications via unicasts between a single client and multiple servers automatically engender a congestion of the network. In order to reduce the problem of congestion, we proposed in [21] a simple adaptive solution based on the leisure defined in the RFC 7390 [3].

Indeed, the fact that the CoAP congestion control, designed for group communication between a single client and multiple servers, doesn't take into consideration the link delay to calculate the delay to insert between consecutive multicast requests, this leads to a congestion control mechanism insensitive to network conditions.

Therefore, in order to improve the delay and to adapt the behavior of our solution to network conditions, we propose a delay between unicast requests depending on the link delay and the estimated group size as shown in formula (10).

$$D = \text{average link delay} * G / G-1 \quad (10)$$

Furthermore, the link delay represents the behavior of the network; if it increases, it means that congestion is more likely to happen, so in order to manage this problem, the estimated delay between unicast requests has to increase. On the other hand, if the link delay decreases, it means that the network is more available and the delays between requests have to be short adapting its behavior to the condition of the network.

To conclude, thanks to its flexibility and its ability to adapt its behavior to different network conditions, this

proposition consistently presents high performances and short response times; it has the ability to increase the number of successful transactions and to decrease the packet loss ratio.

V. CONCLUSION

IoT has offered the ability to transfer data between objects over a network using sensors without any human intervention. In this paper, two important research areas in CoAP based communications were highlighted.

In the two cases of unicast and multicast CoAP-based communication, the number of transactions is very high due to the use of lightweight devices and constrained resources. Thus, in such a network, the problem of congestion is very frequent. Nevertheless, authors propose solutions for congestion control insensitive to network conditions, the thing that lowers its performances.

In this paper, we seek to contribute to the current debate in the literature about congestion issue in CoAP-based communications.

The scientific contribution consists of a conducted large-scale study describing some algorithms used to control congestion in CoAP based communications.

Two suitable solutions to ensure safe network operation, while using network resources efficiently, were presented; unicast CoAP-based communication adaptive to network condition (an estimation of RTO value to use for the next transaction based on the packet loss ratio and the RTT of the previous transmission). Moreover, in multicast group CoAP-based communication an adaptive formula for the calculation of the delay to introduce between consecutive multiple requests was discussed.

ACKNOWLEDGEMENTS

We acknowledge the support provided by colleagues the members of Laboratory of Computer and Systems Engineering (L2IS) in the Faculty of Science and Techniques Cadi Ayyad University-Marrakesh. Thanks extended also to colleagues in Private University of Marrakesh.

REFERENCES

[1] Paxson, V. Allman, M. Computing TCP's Retransmission Timer. RFC 2988, last updated 2021.

[2] Jaffey, T. (2014). MQTT and CoAP, IoT Protocols. Eclipse.

[3] Ishaq, I. Hoebeke, J. Moerman, I. Demeester, P. (2016). Observing CoAP groups efficiently. Ad Hoc Networks, 37, P2, pp. 368-388. Doi.org/10.1016/j.adhoc.2015.08.030.

[4] Yuan, H. Yugang, N. Fenghao, G. (2014). Congestion Control for Wireless Sensor Networks: A survey. Control and Decision Conference. Changsha, China, pp 4853-4858. DOI: 10.1109/CCDC.2014.6853042.

[5] Davis, EG. Calveras, A. Demirkol, I. (2013). Improving Packet Delivery Performance of Publish/Subscribe Protocols in Wireless Sensor Networks. Journal of sensors, 13, pp. 648-680. Doi: 10.3390/s130100648

[6] Paxson, V. Allman, M. Chu, J. Sargent, M. (2015). Computing TCP's Retransmission Timer. Computer Science. RFC.

[7] Balandina, E. Koucheryavy, Y. Gurtov, A. (2013). Computing the Retransmission Timeout in CoAP. The 13th International Conference on Next Generation Wired/Wireless Networking, NEW2AN, St. Petersburg, Russia, pp. 352-362. DOI: 10.1007/978-3-642-40316-3_31

[8] Ancillotti, E. Bruno, R. (2017). Comparison of CoAP and CoCoA+ congestion control mechanisms for different IoT application scenarios. IEEE Symposium on Computers and Communications (ISCC), Heraklion, Greece. DOI: 10.1109/ISCC.2017.8024686.

[9] Bormann, C. Betzler, A. Gomez, C. Demirkol, I. (2016). CoAP Simple Congestion Control/Advanced. Draft bormann-core-cocoa-00. Ver. draft-bormann-core-cocoa.

[10] Betzler, A. Gomez, C. Demirkol, I. Paradells, J. (2016). CoAP Congestion Control for the Internet of Things. IEEE Communications Magazine, 54, Issue. 7, pp. 154 -160. DOI: 10.1109/MCOM.2016.7509394.

[11] Järvinen, I. Daniel, L. Kojo, M. (2015). Experimental Evaluation of Alternative Congestion Control Algorithms for Constrained Application Protocol (CoAP). 2nd World Forum on Internet of Things (WF-IoT), Milan, Italy, pp. 453-458. DOI: 10.1109/WF-IoT.2015.7389097.

[12] Bhalerao, R. Subramanian, SS. Pasquale, J. (2016). An Analysis and Improvement of Congestion Control in the CoAP Internet-of-Things Protocol. Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, USA, pp. 889-894. DOI: 10.1109/CCNC.2016.7444906.

[13] Betzler, A. Gomez, C. Demirkol, I. Paradells, J. (2015). CoCoA+: An advanced congestion control mechanism for CoAP. Ad Hoc Networks, 33, pp. 126 – 139. Doi.org/10.1016/j.adhoc.2015.04.007.

[14] Yadav, RK. Singh, N. Piyush, P. (2020), Genetic CoCoA++: Genetic Algorithm based Congestion Control in CoAP. IEEE 4th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India. DOI: 10.1109/ICICCS48265.2020.9121093.

[15] Betzler, A. Gomez, C. Demirkol, I. Paradells, J. (2013), Congestion Control in Reliable CoAP Communication. 16th ACM international conference on Modeling, analysis & simulation of wireless and mobile systems MSWiM, Barcelona, Spain, pp. 365- 372. Doi.org/10.1145/2507924.2507954.

[16] Rahman, A. Dijk, E. (2014), Group Communication for the Constrained Application Protocol (CoAP). RFC 7390.

[17] Shelby, Z. Hartke, K. Bormann, C. (2016). Constrained Application Protocol (CoAP) draft-ietf-core-coap-17. CoRE Working Group, Internet-Draft.

[18] Antonini, M. Cirani, S. Ferrari, G. Medagliani, P. Picone, M. Veltri, L. (2014). Lightweight multicast forwarding for service discovery in low-power IoT networks. In Proceedings of 22nd International Conference on Software, Telecommunications and Computer Networks (SoftCOM), Split, Croatia, pp. 133–138. DOI: 10.1109/SOFTCOM.2014.7039103.

[19] Ishaq, I. Hoebeke, J. Van den Abeele, F. Moerman, I. Demeester, P. (2014). Flexible Unicast-Based Group Communication for CoAP-Enabled Devices. Sensors, 14, no. 6, pp. 9833-9877. DOI: 10.3390/s140609833.

[20] Ouakasse, F. Rakrak, S. (2019). An Improved Adaptive CoAP Congestion Control Algorithm. International Journal of Online and Biomedical Engineering (iJOE), 15, No. 03, pp. 96-109.

[21] Ouakasse, F. Rakrak, S. (2017). An adaptive solution for congestion control in CoAP-based Group Communications. International Journal of Advanced Computer Science and Applications (IJACSA), 8, Issue 6. (DOI): 10.14569/IJACSA.2017.080629.