UNIVERSITY OF LEEDS

This is a repository copy of *Light-Weight Congestion Control in Constrained IoT Networks*.

White Rose Research Online URL for this paper:
https://eprints.whiterose.ac.uk/196456/

Version: Accepted Version

Proceedings Paper:
Khalek, HA and Mhamdi, L (2023) Light-Weight Congestion Control in Constrained IoT Networks. In: 2022 IEEE Global Communications Conference, GLOBECOM 2022 - Proceedings. GLOBECOM 2022 - 2022 IEEE Global Communications Conference, 04-08 Dec 2022, Rio de Janeiro, Brazil. IEEE , pp. 6265-6270. ISBN 978-1-6654-3541-3

https://doi.org/10.1109/GLOBECOM48099.2022.10000648

© 2022, IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Reuse
Items deposited in White Rose Research Online are protected by copyright, with all rights reserved unless indicated otherwise. They may be downloaded and/or printed for private study, or other acts as permitted by national copyright laws. The publisher or other rights holders may allow further reproduction and re-use of the full text version. This is indicated by the licence information on the White Rose Research Online record for the item.

Takedown
If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing eprints@whiterose.ac.uk including the URL of the record and the reason for the withdrawal request.

White Rose
university consortium
Universities of Leeds, Sheffield & York

eprints@whiterose.ac.uk
https://eprints.whiterose.ac.uk/

Light-Weight Congestion Control in Constrained IoT Networks

Hussam Abdul Khalek   Lotfi Mhamdi
School of Electronic and Electrical Engineering
University of Leeds, UK
Email: {el20hak, L.Mhamdi}@leeds.ac.uk

***Abstract*—The Internet of Things (IoT) is a growing technology that remotely connects multiple devices (ranging across many fields and applications) over the Internet. The scalability of an IoT network mandates a reliable transport infrastructure. Traditional TCP control protocol is unsuitable for such domain, mainly due to energy and power consumption reasons. A lighter version of TCP, Light Weight IP (lwIP) provides a promising solution for current and projected future scalable IoT infrastructures. However, the original lwIP is just a simple mapping of the protocol, without insight into the IoT specific requirements. This paper examines the lwIP congestion control mechanism and addresses its shortcomings. In particular, a detailed examination is devoted to the various metrics such as Retransmission Time-Outs (RTOs) and its back-off epochs, the congestion window behaviour and progress in the absence (and presence) of congestion. In particular, we propose a set of novel algorithms to address both the IoT constraints nature (light-weight) as well as keeping up with scalability in IoT network size and performance. A detailed simulation study has been conducted to endorse the viability of our proposed set of algorithms for next-generation IoT networks.**

***Index Terms*—Congestion Control, IoT, lwIP, TCP.**

I. INTRODUCTION

The Internet of Things (IoT) describes the wireless network of embedded devices and systems that connect and exchange data with other devices and systems over the Internet [1]. IoT has become an integral part of the current technological advancement and is expected to substantially grow in the coming future. IoT technologies are included in multiple applications such as smart houses, smart cars, smart factories, smart security systems, smart and remote healthcare applications, cloud technologies and many other platforms spanning over multiple industries [1]. Currently, there are at least 10 billion active IoT devices, and it is estimated that by 2030 the number of active IoT devices will rise to around 25.4 billion devices [2].

Like all traditional wired and wireless communication networks, IoT-based networks also suffer from network congestion which occurs when the network has more data traffic than it can handle due to the excessive number of devices exchanging data on the network [3]. Unlike UDP, where middle-boxes such as firewalls and network address translation devices (NATs) might effectively block UDP packets, the TCP protocol in IoT yields a seamless integration with existing networks. Furthermore, the latest industry standardisation with regards to the Constrained Application Protocol (CoAP) [4] and its corresponding advanced congestion control algorithm (CoCoA) [5] suggests that TCP will be gaining considerable support in IoT scenarios in the future.

The main obstacle with TCP in IoT-based networks lies in the header over-head, stack size and memory usage [6]. Moreover, the conventional congestion control algorithm poses the issue of multiple end-to-end retransmissions of lost segments over lossy networks with high bit-error rate causing elevated power usage in such cases [6]. To overcome such obstacles, multiple solutions were considered, such as utilising the 6LoWPAN (IPv6 low-power wireless personal area network) adaptation layer over IPv6 to achieve packet compression suitable for IoT applications and embedded systems [7] [8]. Furthermore, light-weight TCP implementations such as micro-IP TCP (μIP) [9] and light-weight IP TCP (lwIP), provide a much smaller stack size with minimal memory usage making them suitable for IoT scenarios. The light-weight implementations can be made even more suitable for IoT scenarios by the use of Radio Duty Cycle (RDC) mechanisms [10] at the medium access control (MAC) layer to further minimise the power usage [6] [11]. To date, although multiple aspects of light-weight TCP implementations have been investigated for IoT scenarios, congestion control remains a challenge that is rarely investigated for constrained IoT networks.

The main objective of this paper is to explore TCP-based implementations for constrained IoT networks. The focus will be on enhancing the performance of the congestion control algorithm for light-weight TCP, particularly the open source lwIP implementation [9] [12]. More specifically, the aim is to increase the total network goodput as well as reduce the network congestion and consequently minimise the number of end-to-end retransmissions since they are a primary factor in energy consumption. This can be achieved by tuning congestion control parameters and setting the right algorithmic adjustments. For this, a set of novel techniques has been proposed. In particular, we propose a novel algorithm named *lwIP Back off* where proper adjustments to the round-trip time (RTT) and retransmission time-out (RTO) estimation mechanism and RTO back-off was carefully chosen. A further improvement, named *lwIP cwnd* was to adopt a more accurate congestion window behaviour than it is originally set in lwIP. Finally, we combined both techniques into a one scheme that we named *lwIP Back off & cwnd* that has been proven through extensive simulation to exhibit good performance and outperform existing algorithms.

The remainder of the paper is structured as follows. Section II discusses some relevant existing related work on light-weight congestion control algorithms. In Section III, analyses existing lwIP implementations. Section IV gives details of the newly proposed set of algorithms that adaptively tune relevant congestion metrics to improve the IoT network utilisation. We describe three different algorithms in Section IV-A, Section IV-B and Section IV-C respectively. Section V presents the simulation settings and discuss the experimental results. Finally, Section VI concludes the paper.

II. RELATED WORK

A study by Lim [6] investigated a way of improving congestion control of the light-weight μIP TCP stack for constrained IoT networks [9]. The study proposes a scheme involving parameter tuning as well as algorithmic and system-level adjustments. Lim‘s approach [4] involved investigating the performance of μIP TCP in a grid topology network with RDC via the Cooja network simulator in Contiki OS [13].

It was established that RDC is a significant tool for saving battery power in wireless sensor networks. However, RDC causes a lot of retransmissions when using light-weight μIP TCP due to the fixed RTO and the large RTT variations caused by the hidden node problem. The aim was to investigate the effect of RTT and RTO estimation on the performance of μIP TCP in constrained IoT networks, to examine the possibility of implementing variable RTO back-off and weak/strong RTT estimation inspired by CoCoA in CoAP and to propose additional mechanisms to further improve μIP TCP performance in constrained IoT networks with RDC enabled.

To evaluate the performance, Lim [6] proposed RPL-based 4×4 and 5×5 grid topology networks with a Linux RPL (Routing protocol for Low-Power and Lossy Networks) border router at the edge of the grid, maintaining a slip connection to a Linux TCP server. Each client maintains a custom μIP TCP implementation with an RDC option. The data exchange included 48 bytes of TCP payload at 64 bits/second over a period of ten minutes. As for the proposed scheme, weak RTT estimation was utilised alongside exponential back-offs with variable limits as well as dithering. The latter is implemented by setting the actual retransmission timer and adding a random duration to it. The results depicted a relatively improved performance when considering their proposed solutions especially on RDC-enabled scenarios. Their proposed scheme provided an increased number of segments across both 4×4 and 5×5 grid networks. However, the original implementation with the addition of dithering only provided a better performance with regards to the network goodput and fairness index.

Although μIP TCP can provide a good experimental starting point, it does not include the relevant TCP features to properly implement congestion control for IoT-based networks. For example, μIP TCP does not support a sliding window and maintains only a single MSS (maximum Segment Size) in its window size. It also does not support neither slow start, nor fast recovery/retransmit [11]. Furthermore, utilising a grid topology network does not isolate the performance measurement to congestion control only, due to the possibility of the hidden node problem as well as multi-hops and consequently packet loss. In this paper, we propose a more suitable approach and testing scheme to evaluate the congestion control performance of a light-weight TCP implementation in IoT scenarios.

III. OVERVIEW OF LIGHT-WEIGHT TCP IMPLEMENTATIONS

Exploring opportunities to make the congestion control mechanism more suitable for IoT scenarios, is central for devising an optimal approach to utilise light-weight TCP adaptations for constrained IoT networks and applications. Most TCP implementations follow the three main congestion control algorithms. Slow start, congestion avoidance as well as fast retransmit/recovery are utilised in that specific order to manage the number of outstanding data being sent over the network. For the original full TCP implementation, if the congestion window, *cwnd*, is less than *ssthresh*, slow start is used, while if *cwnd* is greater than *ssthresh*, congestion avoidance is used. If *cwnd* is equal to *ssthresh*, either of the algorithms can be used.

Light-weight TCP implementations, such as lwIP, follow a similar but more aggressive approach to adhere to IoT scenarios. Primarily, the initial *ssthresh* starts at a lower point at a maximum of $10 \times MSS$. Initially, and upon congestion or upon reaching the *ssthresh*, the *cwnd* is set to always start or restart from $1 \times MSS$. This forces a more conservative approach to try to minimise the points of congestion and ultimately retransmissions since the *cwnd* will take longer to reach the s*sthresh* limit. Meanwhile, important aspects of successful TCP congestion control are the RTT and RTO estimation mechanisms, which ultimately affect the algorithm‘s reactive response to retransmissions as well as the proactive response to prevent future congestion and retransmissions [9].

The features of different accessible light-weight TCP implementations that are currently being used in embedded systems and IoT applications are reported in [11]. Most importantly, it shows that μIP TCP does not implement a sliding window in the sense that it always uses one MSS for its window size. In other words, it can only have one unacknowledged TCP segment per connection. This causes a poor interaction between the sender and a receiver that is using a delayed acknowledgement mechanism. Thereby, resulting in long waiting times at the receiver and thus hindering the sender throughput [6].

In this paper, we seek to adopt a light-weight TCP implementation with a relatively small code size alongside adequate congestion control and other TCP features. The purpose is to avoid being restricted by a single MSS on the window size, and to be able to have at least the slow start, congestion avoidance and fast retransmit/recovery congestion control algorithms. These mentioned features are found in the open source lwIP stack. Therefore, lwIP provides an ideal platform to analyse the congestion control performance in constrained IoT networks, in addition to the fact that it has an optimised memory usage and a compressed code size. Moreover, lwIP is currently being utilised by various leading manufacturers of embedded systems such as Intel, Analog Devices, Xilinx and many others [6] [9].

IV. THE PROPOSED SCHEME

To improve the overall performance of lwIP congestion control mechanism and ultimately increase the total goodput as well as decrease the total number of retransmissions and congestion in general, multiple parameters and features were adjusted within the congestion control algorithm. Below, we describe a set of novel algorithms, each of which aims at addressing one congestion control feature.

A. *lwIP with dynamic RTO back-off*

Firstly, the RTO back-off procedure was investigated. By default, upon a retransmission, the RTO is configured to back-off from its estimated initial value by a doubling factor. This ultimately results in an exponential increase in the RTO value. To avoid having excessively large RTOs and inspired by CoCoA of CoAP, a dynamic back-off factor is investigated. The process involved multiple trial and error iterations by adjusting the back-off factor with respect to a specific RTO value until an improved and optimised result was observed. In particular, the RTO back-off feature of the lwIP algorithm was adjusted as follows:

$$RTO\_back\_off\_factor = \begin{cases} 2 & when\ RTO \leq 3s \\ 1.3 & when\ RTO > 3s \end{cases}$$

This adjustment helps in avoiding extremely high RTOs resulting in long waiting times when a packet is lost and requires retransmission. This lwIP adaptation is referred to as lwIP back-off in the results of this study.

B. *lwIP with Congestion Window adjustment*

This section explores the *cwnd* floor value and its impact on the congestion. Originally, upon a retransmission or upon reaching the *ssthresh*, the *cwnd* is configured to default back to $1 \times MSS$. Such aggressive configuration causes extreme variation in the congestion window value. For instance, the *cwnd* can drop from $10 \times MSS$ down to $1 \times MSS$ with such implementation. This can ultimately impact the total goodput over an extended period. Therefore, we propose to set the *cwnd* to 50% of its *ssthresh* limit instead of resetting it back to $1 \times MSS$ upon a retransmission or upon reaching the ssthresh. This optimal percentage value of 50% was deduced after multiple iterations of trial error involving different configurations of *cwnd* until an optimal improvement in the total goodput is observed. This lwIP adaptation is referred to as *lwIP cwnd* in the results of this study.

TABLE I: Simulation Settings

Network Parameters	**Values**
Topology	Star
Number of packets in MAC/link layer queue	8 entries
Packet size	1024 bytes
Number of packets	1000 packets
Number of clients	12 clients
Frequency	2.4 GHz
Client-Access Point Data rate	2 Mbps
Access Point-Server Data rate	10 Mbps
Server Stack	TCP

C. *lwIP with back off & cwnd adjustment*

Lastly, both dynamic RTO back-off and *cwnd* floor value adjustments were combined in the same implementation with several attempts to optimise the overall performance by tuning both the back-off factor and the *cwnd* configuration. The optimisation process resulted in maintaining the back-off factor and setting the *cwnd* to 50% of its *ssthresh* limit. This lwIP adaptation is referred to as lwIP back-off & cwnd in the results of this study.

V. EXPERIMENTAL SETTINGS AND RESULTS

This section starts by first describing the simulation settings and then we proceed to the experimental results.

A. *Experimental Settings*

We begin by describing the network topology and associated simulation parameters. We have used the OMNET++ simulator [14] and created a star topology network. The network includes 12 clients wirelessly connected to an access point at 2.4 GHz and adhering to the IEEE802.11 Wi-Fi standard. Meanwhile, the access point is connected to a TCP server via an Ethernet connection. The network follows a star topology with all 12 clients evenly spaced and at the same approximate distance from the access point. The star topology was deemed more suitable since it provides a fairer state of communication across all clients and eliminates the hidden node problem. This implementation would be more likely to accentuate the performance evaluation on the congestion control.

The client access-point wireless links run at 2 $Mbps$, while the access point-server Ethernet link runs at 10 $Mbps$. After establishing a TCP connection with the server, each client sends a 1024 Byte packet to the server, through the access point, every 0.4 seconds for a total of 1000 packets over a period of 400 seconds. The server acts as a sink by only responding with an ACK. TABLE I shows a compiled summary of the relevant network and simulation settings. The network simulation is configured to either run with original TCP clients as reference, or lwIP clients. The recorded parameters include RTT, RTO, *cwnd*, *ssthresh*, total goodput and the total number of retransmissions. The RTT, RTO, *cwnd*, *ssthresh* and the total number of retransmissions are recorded in accordance with a single client, while the total goodput represents the network‘s aggregated total bits/seconds received at the input of the access point.

B. Experimental Results

The first experimental result we evaluated is related to the Retransmission Time Out (RTO) effect. We begin by comparing the RTO values of the original lwIP scheme as compared to our proposed lwIP_back_off scheme, as described in Section IV-A. As depicted in Fig. 1, a smaller RTO estimation is observed throughout the entire simulation with a maximum of 7s for lwIP with dynamic RTO back-off and 24s for the original lwIP. This is attributed to the back off adjustment that dynamically sets the retransmission time out according to its latency.

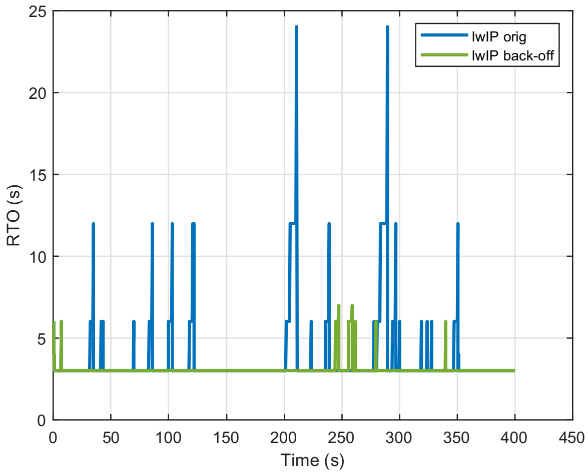

Fig. 1: Progression of RTO values of lwIP and lwIP_back_off algorithms.

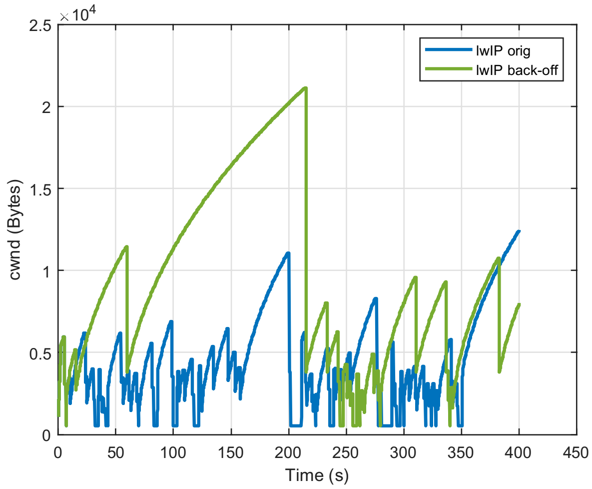

Fig. 2: Congestion window progress of lwIP and lwIP_back_off algorithms.

We further observed the congestion window behaviour over time of both the original lwIP algorithm and our proposed lwIP_back_off. Fig. 2 shows that the lwIP implementation with dynamic RTO back-off results in a *cwnd* with less congestion points as compared to the original lwIP implementation. In other words, the *cwnd* tends to climb to higher values with less congestion points represented by retransmissions or reaching the *ssthresh*. Such response depicts less overall congestion within the network and therefore results in higher network utilisation.

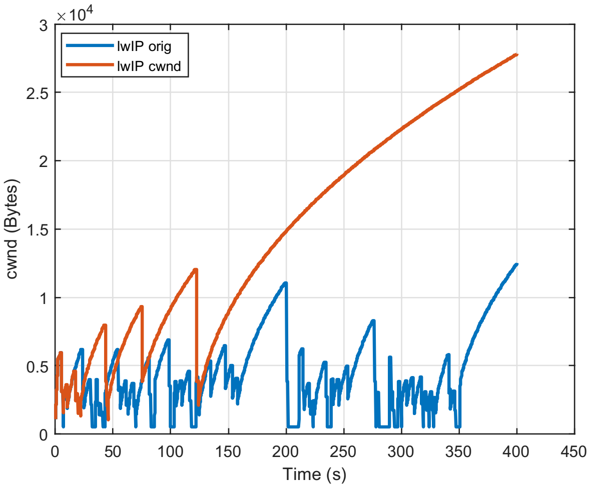

Fig. 3: Congestion window progress of lwIP and lwIP_cnwd algorithms.

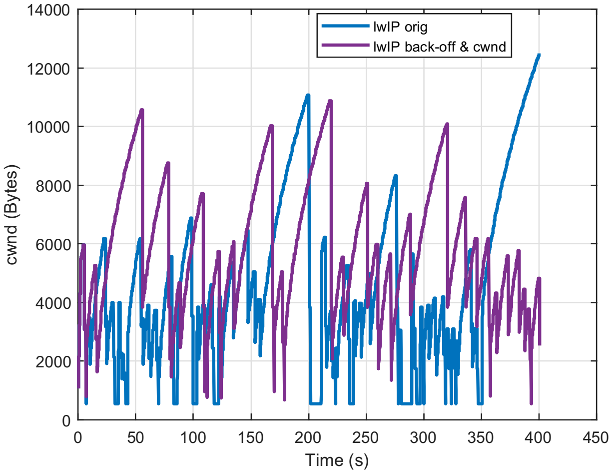

Fig. 4: Congestion window progress of lwIP and lwIP_back_off_cwnd algorithms.

A further technique has been to adaptively update the congestion window, as presented in Section IV-B with the proposed lwIP_cnwd algorithm. We have compared the congestion window evolution over time of both the original lwIP algorithm and that of lwIP_cnwd. . As can be seen in Fig. 3, the lwIP implementation with the *cwnd* adjustment results in a better *cwnd* behaviour in the sense that *cwnd* drops to a higher floor value upon a retransmission or when the *ssthresh* is reached in comparison to the original lwIP implementation that defaults to one MSS. Moreover, *cwnd* tends to climb to higher values with less congestion points represented. We further observed the congestion window growth of the original and the combined algorithms, as presented Section IV-C. A similar trend is observed in Fig. 4, where the network can sustain higher traffic and is able to absorb transient congestion periods resulting in higher network utilisation. Fig. 5 summarises the congestion window behaviour of all algorithms and shows the merits of each of them, as discussed above.

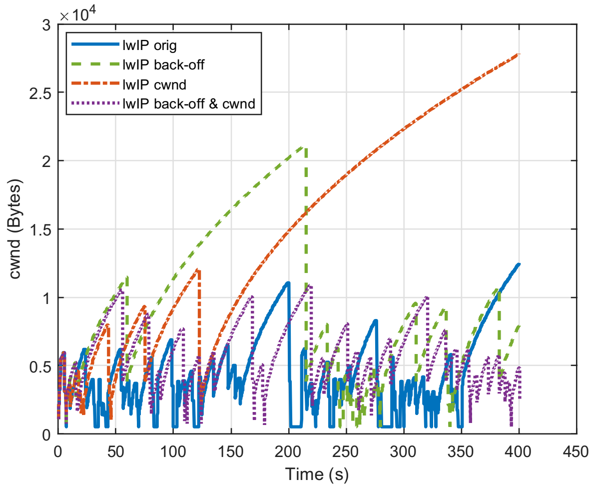

Fig. 5: Summary of Congestion window progress of all algorithms combined

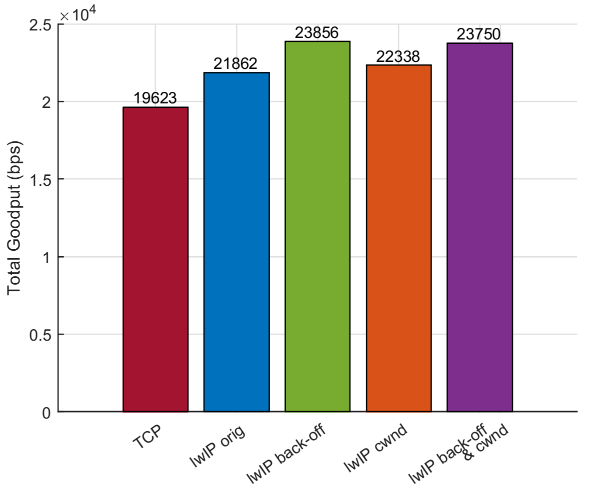

Fig. 6: Total network goodput for all algorithms.

Fig. 6 shows that the total network goodput of all algorithms. As can be seen from the Figure, the goodput was certainly improved from 22 Kbps for original lwIP implementation to 24 Kbps when using either of lwIP with dynamic RTO back-off adjustment and/or lwIP back-off & cwnd floor value adjustment, but it was slightly less than the lwIP with *cwnd* floor value alone.

We have also computed the total number of retransmissions of all algorithms over the simulation time, as depicted in Fig.7. As we can see, the total number of retransmissions significantly decreased from 136 for original lwIP to less than 27. We can see that this is dropped to 10 for lwIP with both dynamic RTO back-off and cwnd floor value adjustment which is the minimum number of retransmissions across all implementations. Therefore, the final combined implementation provides a slight trad-off on the network goodput when compared to lwIP with dynamic RTO back-off, but reduces the total number of retransmissions to a minimum.

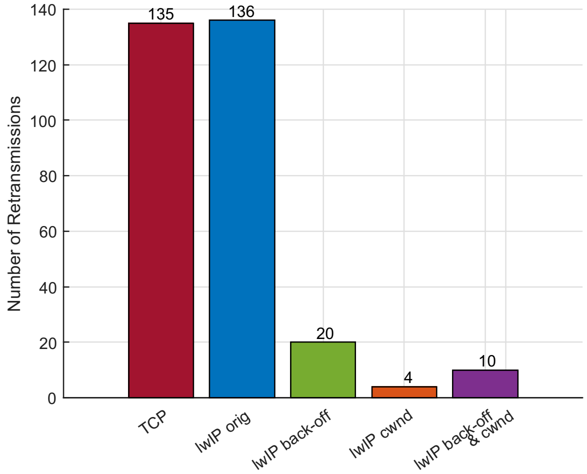

Fig. 7: Total Number of retransmissions of all algorithms.

VI. CONCLUSION

IoT devices and networks will be a significant part of the future. They are utilised for multiple applications across various industries and will only keep growing in the foreseeable future, warranting careful consideration to the reliability of the underlying IoT network and it congestion. Light-weight TCP implementations have been considered as suitable candidates for TCP-based IoT networks. This paper proposes a light-weight TCP set of implementations and evaluated their performance merits through simulations of a typical start topology IoT network. The results show that the proposed novel schemes produced an improvement in terms of total goodput of the IoT network as well as a significant drop in the total number of retransmissions for lwIP clients. In particular, finding the right optimal combination of congestion metrics, such as shown by the lwIP back-off & cwnd algorithm, is the right sailing direction in overcoming congestion in IoT networks. This is considered as a one step towards making ulta-scalable IoT networks that are more reliable both in traffic growth as well as node count.

REFERENCES

[1] I. Lee and K. Lee, “The internet of things (iot): Applications, investments, and challenges for enterprises,” *Business Horizons*, vol. 58, no. 4, pp. 431–440, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0007681315000373

[2] B. Jovanovic. Internet of Things statistics for 2021 Taking Things Apart, year = 1999, url = https://dataprot.net/statistics/iot-statistics/ ,.

[3] W.-C. Hung and K. E. Law, "Simple slow-start and a fair congestion avoidance for tcp communications," in *2008 Canadian Conference on Electrical and Computer Engineering*, 2008, pp. 001 771–001 774.

[4] D. Sembroiz, S. Ricciardi, and D. Careglio, "Chapter 10 - a novel cloud-based iot architecture for smart building automation," in *Security and Resilience in Intelligent Data-Centric Systems and Communication Networks*, M. Ficco and F. Palmieri, Eds. Academic Press, 2018, pp. 215–233.

[5] A. Betzler, C. Gomez, I. Demirkol, and J. Paradells, "Cocoa+: An advanced congestion control mechanism for coap," *Ad Hoc Networks*, vol. 33, pp. 126–139, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570870515000888

[6] C. Lim, "Improving congestion control of tcp for constrained iot networks," *Sensors*, vol. 20, no. 17, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/17/4774

[7] Y. A.-N. H. A. A. Al-Kashoash, H. Kharrufa and A. H. Kemp, "Congestion control in wireless sensor and 6lowpan networks: toward the internet of things," *Wireless Networks*, vol. 25, no. 8, pp. 4493–4522, 2019.

[8] A. Dunkels, J. Eriksson, N. Finne, F. sterlind, N. Tsiftes, J. Abeill, and M. Durvy, "Low-power ipv6 for the internet of things," in *2012 Ninth International Conference on Networked Sensing (INSS)*, 2012, pp. 1–6.

[9] A. Dunkels, "Design and implementation of the lwip tcp/ip stack," *Swedish Institute of Computer Science*, vol. 2, 03 2001.

[10] Z. Ahmed, M. Rehan, O. Chughtai, and W. Rehan, "Ad-rdc: A novel adaptive dynamic radio duty cycle mechanism for low-power iot devices," *IEEE Internet of Things Journal*, pp. 1–1, 2022.

[11] M. S. J. C. C. Gomez. TCP Usage Guidance in the Internet of Things (IoT), year = 2009, url = https://tools.ietf.org/id/draft-ietf-lwig-tcp-constrained-node-networks-05.html,.

[12] A. Dunkels, "Full tcp/ip for 8-bit architectures," ser. MobiSys '03. New York, NY, USA: Association for Computing Machinery, 2003, p. 8598. [Online]. Available: https://doi.org/10.1145/1066116.1066118

[13] C. Thomson, I. Romdhani, A. Al-Dubai, M. Qasem, B. Ghaleb, and I. Wadhaj, "Cooja simulator manual," 07 2016.

[14] A. Varga, "The omnet++ discrete event simulation system," *Proc. ESM'2001*, vol. 9, 01 2001.

[15] Z. Y. Z. e. a. AMusaddiq, A., "Routing protocol for low-power and lossy networks for heterogeneous traffic network," *EURASIP Journal on Wireless Communications and Networking*, vol. 21, pp. 126–139, 2020.

[16] G. Mulligan, "The 6lowpan architecture," in *Proceedings of the 4th Workshop on Embedded Networked Sensors*, ser. EmNets '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 7882. [Online]. Available: https://doi.org/10.1145/1278972.1278992

[17] IETF. TCP Congestion Control, IETF, RFC 5681, year = 2009, url = https://datatracker.ietf.org/doc/html/rfc5681.

[18] C. Gomez, A. Arcia-Moret, and J. Crowcroft, "Tcp in the internet of things: From ostracism to prominence," *IEEE Internet Computing*, vol. 22, no. 1, pp. 29–41, 2018.