International Journal of Internet, Broadcasting and Communication Vol.15 No.1 1-7 (2023)
http://dx.doi.org/10.7236/IJIBC.2023.15.1.1

IJIBC 23-1-1

A simulation study on TCP performance for constrained IoT networks

Chansook Lim

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

Abstract

TCP is considered a major candidate transport protocol even for constrained IoT networks due to its ability to integrate into the existing network infrastructures. Since TCP implementations such as uIP TCP often allow only a single TCP segment per connection to be unacknowledged at any given time due to resource constraints, the congestion control relies only on RTO management. In our previous work, to address the problem that uIP TCP performs poorly particularly when a radio duty cycling mechanism is enabled and the hidden terminal problem is severe, we proposed a RTO scheme for uIP TCP and validated the performance through Cooja simulation. In this study, we investigate the effect of other factors that were not considered in our previous work. More specifically, the effect of traffic intensity, the degree of the hidden terminal problem, and RDC is investigated by varying the offered load and the transmission range, and the RDC channel check rate. Simulation results imply that we need to further investigate how to improve TCP performance when the radio duty cycling mechanism is used.

Keywords: *TCP, IoT network, Cooja, congestion control*

1. Introduction

Protocol standardization for IoT makes TCP emerge as a major candidate transport protocol even for constrained IoT networks. TCP is known to be better than UDP for integration with the existing network infrastructures because UDP-based communications may be limited or blocked in existing infrastructures.

For the past decade, there have been several studies on TCP for constrained IoT networks. Hurni et al. attempted to improve TCP performance by examining the effect of distributed caching and local retransmission strategies [4]. Their main idea is that each intermediate node caches TCP segments and retransmits a segment whose ACK is considerably delayed, based on RTT estimation. In [5], Kim et al. conducted an experimental study on performance of TCP over RPL in an IPv6-based testbed using the TinyOS BLIP stack, one LBR and one Linux-based server. They find that TCP incurs significant throughput unfairness among nodes in multihop LLNs and that RPL may adversely affect TCP performance when RPL is unable to balance traffic load. Park et al. attempted to address the TCP fairness problem among LLN endpoints, by proposing TAiM (TCP assistant

Manuscript Received: October. 28, 2022 / Revised: November. 4, 2022 / Accepted: November. 8, 2022
Corresponding Author: chansooklim@hongik.ac.kr
Tel: +82-44-860-2549, Fax: +82-44-865-0460
Professor, Department of Software and Communications Engineering, Hongik University, Sejong, Korea

in the middle) [6]. TAiM intervenes in the middle of TCP communication only at LBR, and manipulates RTT of the passing flows so that a flow with low throughput can have shorter delay whereas a flow with high throughput may have longer delay. In [1], Gomez et al. provide recommendations for lightweight TCP implementation and suitable operation in IoT scenarios. For more powerful low-power embedded devices, Kumar et al. proposed a full-scale TCP, called TCPlp [7] that can fit well within CPU and memory constraints of modern wireless sensor network(WSN) platforms leveraging the full-featured TCP in FreeBSD OS. In [7], the authors deal with the issues related to hidden terminals and radio duty cycling. To reduce hidden-terminal-induced packet loss for TCP, they propose adding delay between link-layer retransmissions. Also, to overcome poor interaction of TCP's self-clocking with the duty-cycled link, they proposed Adaptive Duty Cycling. Their study is closely related with ours but different from ours in that they use a full-scale TCP. More recently, Sakamoto et al. presented an implementation of TCP/IP stack over Private LoRa [8].

In our previous work [9], we tried to improve TCP performance in a constrained network environment where a radio duty cycling (RDC) mechanism is used and the hidden terminal problem is severe. We proposed a scheme for managing the retransmission timer which adopts the notion of weak RTT estimation of CoCoA, exponential backoffs with variable limits, and dithering. Although we validated performance through Cooja simulation in our previous study, the investigation in diverse scenarios was left as future work.

In this work, we investigate the effect of other factors that were not considered in [9]. More specifically, we examine the effect of traffic intensity, the degree of the hidden terminal problem, and RDC on TCP performance by varying the offered load, the transmission range, and the RDC channel check rate. Cooja simulation results show that high offered load, severity of the hidden terminal problem, and a low RDC channel check rate are all the important factors affecting TCP performance.

2. Background

In this section, we describe the congestion control mechanism in the original uIP TCP/IP [2,3], ContikiMAC duty cycling mechanism [10] and our previous work [9] briefly.

The uIP TCP/IP stack is a small implementation of the TCP/IP protocol suite for embedded systems. uIP TCP allows only a single TCP segment per connection to be unacknowledged at any given time. Even if each node generates low-rate traffic, congestion can occur at areas where traffic is concentrated. Hence uIP TCP needs congestion control by RTO management. Contiki OS supports TCP using uIP TCP/IP stack.

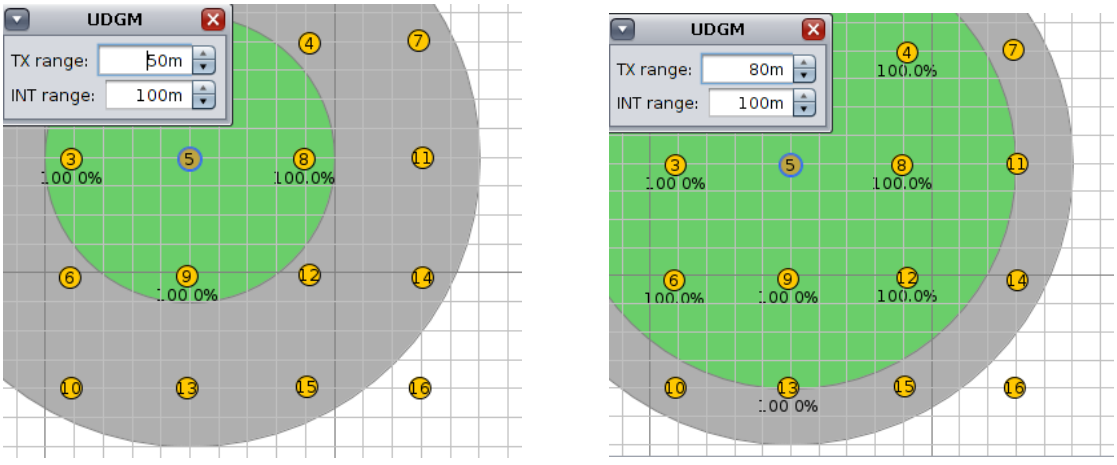
RDC (radio duty cycling) is a major tool to save battery power in wireless sensor networks. Contiki OS supports three RDC mechanisms: ContikiMAC, X-MAC and LPP(Low-Power Probing). Among them, we use ContikiMAC, which is the default RDC mechanism in Contiki OS. ContikiMAC [10] is a sender-initiated asynchronous RDC mechanism that does not use signaling messages and additional packet headers. ContikiMAC uses a power efficient wake-up mechanism with a set of timing constraints to allow devices to keep their transceivers off. To give an indication of radio activity on the channel, ContikiMAC wake-ups use a Clear Channel Assessment (CCA) mechanism that uses Received Signal Strength Indicator (RSSI) of radio transceiver. If a receiver detects a packet transmission during a wake-up, the receiver keeps on to be able to receive the packet. The most important issue of asynchronous RDC mechanisms is that it may result in long delays that accumulate over multihop path [13]. To allow switching off the RDC mechanisms, Contiki OS provides the NullRDC that leaves the radio always on.

The Cooja network simulator [11,12] is a Java-based application which was designed for simulation in the domain of wireless sensor networks. One of the main features of Cooja is the capability to emulate various

motes constituting wireless sensor networks. The emulation mechanism allows Cooja to perform fine-grained, precise and low-level simulations. Cooja supports a GUI interface to design, run, and analyze WSN simulations.

Our previous work [9] shows that uIP TCP performs poorly when an RDC mechanism is enabled and the hidden terminal problem is severe. The main cause is that the original uIP TCP uses the fixed RTO(3sec) for retransmissions after the first transmission. Since RDC and hidden terminals may jointly cause large RTT variations, the fixed RTO does not work well. To address the problem, we proposed a scheme for managing the retransmission timer which adopts the notion of weak RTT estimation of CoCoA, exponential backoffs with variable limits, and dithering. Weak RTT estimation means RTT estimation using retransmitted messages as well. In [9], we validated performance through Cooja simulation by varying several conditions including the scale of grid topologies, the loss rate, the link layer queue size. However, we fixed the offered load, the transmission range, and the interference range in order to focus on the hidden terminal problem in the given grid topology.

In this work, we investigate the effect of the offered load, the degree of the hidden terminal problem, and the channel check rate of RDC on TCP performance. To change the degree of the hidden terminal problem, we simply vary the ratio of the transmission range to the interference range. Figure 1 illustrates how the transmission range affects the degree of the hidden terminal problem. In this example, the interference range is fixed at 100m. Consider node 5. As shown in Figure 1(a), when the transmission range is 50m, node 5 suffers from the hidden terminal problem involving 9 nodes (node 4, 6, 7, 10, 11, 12, 13, 14, 15) which are within the interference range but beyond the transmission range. On the other hand, when the transmission range is 80m, the number of the nodes that can cause the hidden terminal problem for node 5 is reduced to 4 (node 7, 10, 14, and 15) as shown in Figure 1(b).



(a)when the transmission range is 50 m (b) when the transmission range is 80 m

Figure 1. Illustration of varying transmission range

3. Simulation

3.1 Simulation setup

In this study, we perform Cooja simulation using a 4x4 grid topology which consists of Sky motes. In the grid topology, the nodes in a same row or column are 40m away from each other and the top leftmost node was designated as the LBR, which is connected to a Linux machine on a virtual machine using a slip protocol.

Our main performance metrics are the total goodput, the retransmission ratio, and fairness. The total goodput is the total number of the segments that the Linux server receives. The retransmission ratio is the total number of retransmitted segments divided by the total number of segments received by the TCP server. Fairness between TCP flows is quantified using Jain's index.

Table 1 shows the parameter settings for our simulation. To focus on the effect of the offered load, hidden terminals, and RDC, we set the random loss rate to 0. For the RDC mechanism, we use ContikiMAC which is the default RDC mechanism for Cooja. To better understand the effect of the RDC, we use two RDC channel check rates, 8Hz and 16Hz. When NullRDC is enabled, no RDC mechanism is used. As previously mentioned, the interference range is fixed at 100m. The simulation duration is 10 minutes.

Typical application scenarios in lossy and low-power networks such as monitoring require multipoint-to-point traffic flows from the sensing devices to the central control point(i.e., the sink node). Hence, a large amount of traffic may be concentrated around the central control point depending on the network size even

Table 1. Simulation parameters

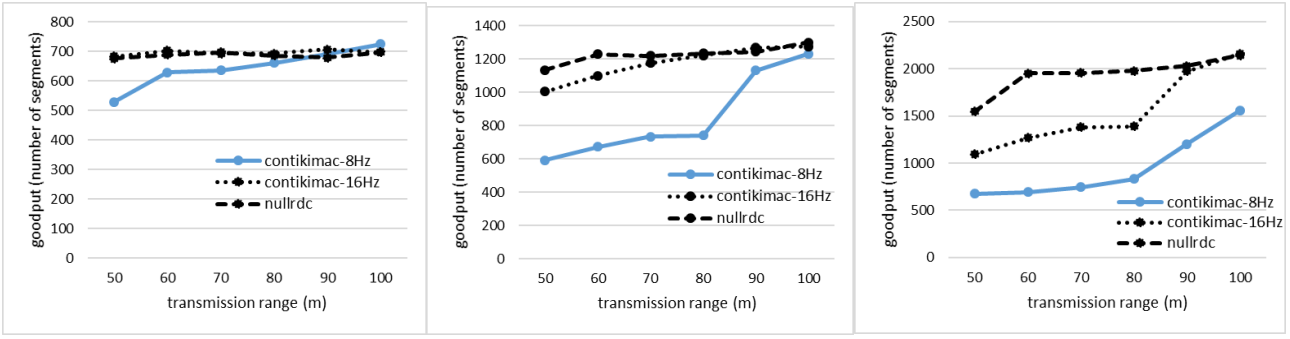
Parameters	Values
Radio model	Unit Disk Graph Medium (UDGM)
Random loss rate	0%
Interference range	100m
Radio duty cycling mechanism	NullRDC or ContikiMAC
RDC channel check rate	8Hz or 16Hz
RPL objective function	MRHOF
Number of packets in the link-layer queue	4
Length of TCP payload	48 bytes
Simulation duration	10 minutes

though each node generates low-rate traffic. If the number of sender nodes is 15 and the offered load is 10 segments per minute per node, the total offered load to the sink node is 150 segments per minute. Since our simulation duration is 10 minutes, the sink is supposed to receive up to 1500 segments. Unlike our previous work which fixed the offered load at 10 segments per minute per node, this study varies the offered load from 5 to 20 segments per minute per node.

To examine the effect of the ratio of the transmission range to the interference range on TCP performance, we fix the interference range at 100m and varies the transmission range from 50m to 100m.

3.2 Simulation results

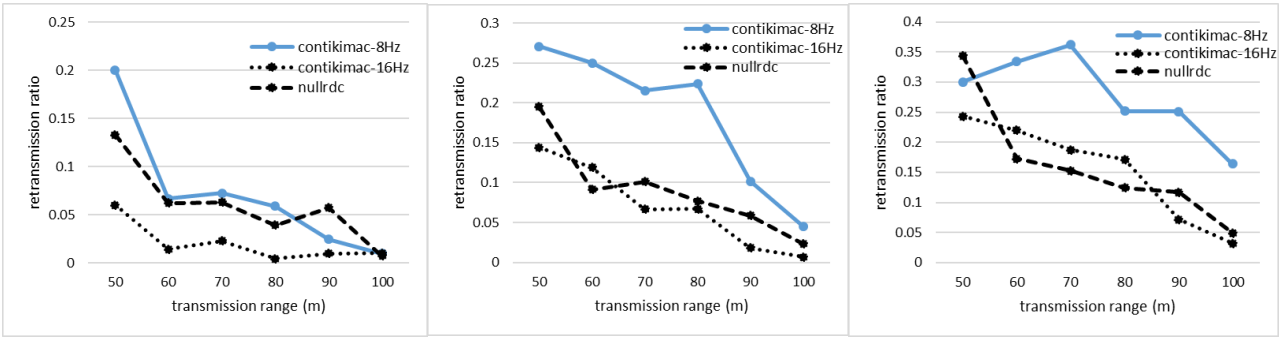
Figure 2 shows the total goodput with respect to the transmission range. When the NullRDC is used, TCP goodput is not very sensitive to the transmission range except when the offered load is 20 segments per minute per node and the transmission range is 50m. When the ContikiMAC is used, the goodput is heavily affected by the transmission range. For the offered load of 5 and 10 segments per minute per node, the TCP goodput of the ContikiMAC gets closer to that of the NullRDC as the transmission range increases and therefore the hidden terminal problem gets less severe. However, for the offered load of 20 segments per minute per node, the gap between the TCP goodput of ContikiMAC and that of the NullRDC is still considerable especially when the RDC channel check rate is 8Hz. This implies that we need to investigate how to improve TCP performance when the RDC is used but there is no severe hidden terminal problem.



(a) 5 segments per min. per node (b) 10 segments per min. per node (c) 20 segments per min. per node

Figure 2. Total goodput with respect to transmission range

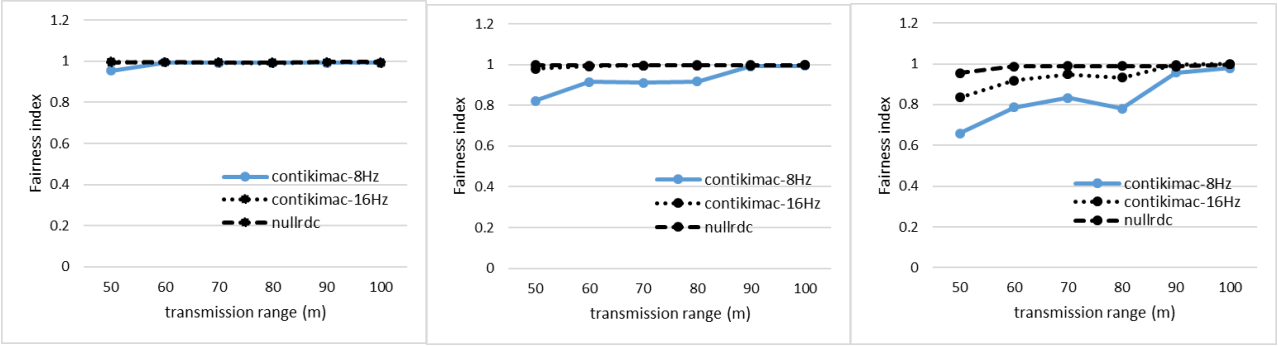
Figure 3 depicts the retransmission ratio with respect to the transmission range. Overall, for the ContikiMAC and the NullRDC both, the retransmission ratio decreases as the transmission range increases. It is notable that with the RDC channel check rate of 16Hz, the retransmission ratio of the ContikiMAC is comparable to that of the NullRDC. One explanation is that since the queue size is small in our simulation, the impact of the sleep interval is substantial. At a longer sleep interval, more packets wait in the queue to be sent, and therefore more packets are lost due to the small queue size. On the other hand, at a shorter sleep interval, there are fewer packets in the queue, and packet losses are reduced.



(a) 5 segments per min. per node (b) 10 segments per min. per node (c) 20 segments per min. per node

Figure 3. Retransmission ratio with respect to transmission range

Figure 4 shows the TCP fairness with respect to the transmission range. For all the offered load of 5, 10, and 20 segments per minute per node, the NullRDC demonstrates good TCP fairness. With the RDC channel check rate of 8Hz, the TCP fairness index of ContikiMAC gets lower as the transmission range decreases. The reason is that the hidden terminal problem worsens for shorter transmission ranges. The nodes far away from the LBR suffer from the effects of the hidden terminal problem more severely than the nodes near the LBR because the impact of hidden terminals is accumulated over multihop. Since the hidden terminals and the sleep interval jointly cause latency to accumulate over multihop, the nodes farther away from the LBR experience longer RTT and more packet losses, which leads to poor TCP goodput. However, with the RDC channel check rate of 16Hz, the TCP fairness is considerably improved because RTT decreases due to the shorter sleep interval. In particular, at the offered load of 5 and 10 segments per minute per node, the TCP fairness of the ContikiMAC with the RDC channel check rate of 16Hz is comparable to that of the NullRDC.



(a) 5 segments per min. per node (b) 10 segments per min. per node (c) 20 segments per min. per node

Figure 4. Fairness index with respect to transmission range

4. Conclusions

In this study, we investigated the effect of traffic intensity, hidden terminals, and RDC on TCP performance by varying the offered load, the transmission range and the RDC channel check rate in Cooja simulation. Our simulation results show that the gap of the TCP goodput between the ContikiMAC and the NullRDC is considerable even when the hidden terminal problem is mitigated. We plan to further investigate how to improve TCP performance when the radio duty cycling mechanism is used.

Acknowledgement

This work was supported by 2020 Hongik University Research Fund.

References

[1] C. Gomez, A. Arcia-Moret, and J. Crowcroft, "TCP in the Internet of Things: from ostracism to prominence," *IEEE Internet Computing*, Vol. 22, 1 2018, pp. 29-41. DOI: https://doi.org/10.1109/MIC.2018.112102200

[2] A. Dunkels, "Full TCP/IP for 8-bit architectures," 2003, pp. 85-98. DOI: https://doi.org/10.1145/1066116.1066118

[3] A. Dunkels, "uIP-A free small TCP/IP stack," *The uIP*, Vol. 1 2002.

[4] P. Hurni, U. Bürgi, M. Anwander, and T. Braun, "TCP performance optimizations for wireless sensor networks," 2012, pp. 17-32. DOI: https://doi.org/10.1007/978-3-642-28169-3_2

[5] H. Kim, H. Im, M. Lee, J. Paek, and S. Bahk, "A measurement study of TCP over RPL in low-power and lossy networks," *Journal of Communications and Networks*, Vol. 17, 6 2015, pp. 647-655. DOI: https://doi.org/10.1109/JCN.2015.000111

[6] M. Park and J. Paek, "TAiM: TCP assistant-in-the-middle for multihop low-power and lossy networks in IoT," *Journal of Communications and Networks*, Vol. 21, 2 2019, pp. 192-199. DOI: https://doi.org /10.1109/JCN.2019.000017

[7] S. Kumar, M.P. Andersen, H. Kim, and D.E. Culler, "Performant TCP for Low-Power Wireless Networks," 2020, pp. 911-932.

[8] J. Sakamoto, D. Nobayashi, K. Tsukamoto, T. Ikenaga, G. Sato, and K. Takizawa, "Poster: Implementation and Performance Evaluation of TCP/IP Communication over Private LoRa," 2022, pp. 1-2. DOI: https://doi.org/10.1109/ICNP55882.2022.9940334

[9] C. Lim, "Improving Congestion Control of TCP for Constrained IoT Networks," *Sensors*, Vol. 20, 17 2020, pp. 4774. DOI: https://doi.org/10.3390/s20174774

[10] A. Dunkels, "The contikimac radio duty cycling protocol," Swedish Institute of Computer Science, 2011.
[11] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with cooja," 2006, pp. 641-648. DOI: https://doi.org/10.1109/LCN.2006.322172
[12] K. Roussel and O. Zendra, "Using Cooja for WSN simulations: Some new uses and limits," 2016, pp. 319–324.
[13] R.C. Carrano, D. Passos, L.C. Magalhaes, and C.V. Albuquerque, "A comprehensive analysis on the use of schedule-based asynchronous duty cycling in wireless sensor networks," Ad Hoc Networks, Vol. 16 2014, pp. 142-164. DOI: https://doi.org/10.1016/j.adhoc.2013.12.009