Received December 1, 2018, accepted December 14, 2018, date of publication December 24, 2018,
date of current version January 11, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2888939

Latency Analysis of Wireless Networks for Proximity Services in Smart Home and Building Automation: The Case of Thread

DAPENG LAN[1], ZHIBO PANG[2], (Senior Member, IEEE), CARLO FISCHIONE[3], (Senior Member, IEEE), YU LIU[4], AMIR TAHERKORDI[1], AND FRANK ELIASSEN[1], (Member, IEEE)

[1]Department of Informatics, University of Oslo, 0316 Oslo, Norway
[2]ABB Corporate Research, 722 26 Västerås, Sweden
[3]Department of Network and Systems Engineering, Royal Institute of Technology, 114 28 Stockholm, Sweden
[4]Department of Science and Technology, Linköping University, 581 83 Linköping, Sweden

Corresponding author: Zhibo Pang (pang.zhibo@se.abb.com)

This work was supported by Norwegian Research Council under Grant 262854/F20 (DILUTE project).

ABSTRACT Proximity service (ProSe), using the geographic location and device information by considering the proximity of mobile devices, enriches the services we use to interact with people and things around us. ProSe has been used in mobile social networks in proximity and also in smart home and building automation (Google Home). To enable ProSe in smart home, reliable and stable network protocols and communication infrastructures are needed. Thread is a new wireless protocol aiming at smart home and building automation (BA), which supports mesh networks and native Internet protocol connectivity. The latency of Thread should be carefully studied when used in user-friendly and safety-critical ProSe in smart home and BA. In this paper, a system level model of latency in the Thread mesh network is presented. The accumulated latency consists of different kinds of delay from the application layer to the physical layer. A Markov chain model is used to derive the probability distribution of the medium access control service time. The system level model is experimentally validated in a multi-hop Thread mesh network. The outcomes show that the system model results match well with the experimental results. Finally, based on an analytical model, a software tool is developed to estimate the latency of the Thread mesh network, providing developers more network information to develop user-friendly and safety-critical ProSe in smart home and BA.

INDEX TERMS Proximity service, smart home, building automation, Internet of Things, IEEE 802.15.4, latency, Thread, wireless sensor network.

I. INTRODUCTION

Proximity service (ProSe) exploits proximity of the mobile devices by utilizing information such as geographic location and physical devices data, changing the way we interact with people and things [1]. ProSe has been considered having a good potential usage in mobile social networks in proximity (MSNP). MSNP is a wireless peer-to-peer (P2P) opportunistic network using geo-proximity in addition to nowadays mobile social networks such as Facebook a and Linkedin [2]. However, ProSe is also playing an important role in other areas, such as smart home and BA, containing the interacting services between people and people, people and things and things and things [3]. In smart home and BA, ProSe has become increasingly prosperous with the advancement of new technologies such as big data, artificial intelligence and wireless networks. For examples, Apple Homepod, Amazon Echo and Google Home utilize the above technologies to provide seamless proximity services for users by using geographic location, physical devices data and other information. Importantly, these ProSe and application scenarios need stable support from network protocols and communication infrastructures with different requirements, like latency, reliability, bandwidth and so on.

To ensure a satisfactory user experience of ProSe, one important factor is network latency [4]. ProSe generally uses IP-based networks, which introduce extra latency due to the IP stack. However, users need to have smooth user experience without long waiting time when interacting with other people or things. Also, some other services in smart home and BA require quality of service (QoS) guarantees [5].

Safety-critical applications in smart home and BA, such as healthcare systems for elderly people, fire alarms, require reliability and time predictability. Google Home fully utilizes Thread protocol as network protocol and communication infrastructure to provides various use scenarios and products such as thermostats, cameras, and alarm systems. Thread protocol, mainly introduced for smart homes and BAs [6], is a new WSN protocol developed by Thread Group to connect and control IoT products. To better understand and develop ProSe, it is crucial to investigate wireless network protocols with improved responsiveness for ProSe such as the Thread protocol.

To the best of our knowledge, this paper is the first work that investigates the latency of the Thread protocol through an experimental study, though security [7] and energy [8] respectively, have been investigated. We propose practical methods and guidelines to study the latency of Thread as an IoT protocol used in ProSe of smart home and BA. The deep understanding of the latency of Thread protocol facilitates designing and developing ProSe because developers can precisely analyze and guarantee QoS in smart home and BA systems. To this end, we propose a Markov chain-based system-level model to theoretically analyze the latency of the Thread mesh network and verify the model results through an experimental study. Based on the proposed analytical model, we developed a software tool for ProSe developers to estimate the latency of the Thread mesh network. The evaluation of the model in the multi-hop Thread mesh network shows that the model results match well with the experimental results.

The rest of the paper is organized as follows. In Section II, related work on latency of WSNs and mathematical models for WSNs are presented. In Section III, an overview of the Thread protocol is introduced. In Section IV, a system level model for the latency of the Thread mesh network is proposed considering the latency introduced at different network layers, from the application layer to the physical layer. The accuracy of the model is experimentally validated in Section V. Section VI concludes the paper and proposes our future-work.

II. RELATED WORK

A. LATENCY OF WSNs

Latency of WSNs is critical in many applications such as industrial automation, healthcare systems and safety-critical applications in BA, such as fire alarm detections [9]. Many works have investigated the latency of WSNs in these areas. In [10], the latency of asynchronous MAC protocols has been surveyed in delay-sensitive wireless sensor networks. The work [11] pointed out that it is hard to predict the accurate latency introduced by WSNs when the MAC implements CSMA/CA. Instead, the work applied a Time-Division Multiple Access (TDMA)-based protocol to a slightly modified IEEE 802.15.4 to assure that the transmission latency of sensors is not more than 10 ms. Sahoo *et al.* [12] proposed a new channel access mechanism for IEEE 802.15.4e-based low latency deterministic network shared slots. Also based on 802.15.4e, Daneels *et al.* [13] came up with recurrent low-latency scheduling in Time-Slotted Channel Hopping networks. However, when modeling the latency of WSNs, practical hardware limitations may cause significant differences between theoretical models and real tests [14]. In fact, the work [15] mentioned that the latency of multi-hop networks is increased significantly by packet relaying. Despaux [16] proposed an approach for modeling and estimating the end-to-end delay for WSNs considering the protocol implementation and hardware limitations. In [17], a blue probability generation function (PGF) was proposed to derive the delay of successfully received packets. However, in practice, understanding the latency of WSNs requires experimental studies combined with the theoretical analysis. Deep understanding of the distribution latency of WSNs together with experimental studies can guarantee the QoS in smart home and BA. It also provides practical guidelines for developers to design user-friendly ProSe when developing applications.

B. MATHEMATICAL MODELS OF WSN MAC

Recently, several analytical studies for the latency of IEEE 802.15.4 have been proposed, e.g., [18]–[21], inspired by Bianchi's work [22]. Due to the stochastic nature of MAC protocols, these models are mainly based on Markov Chains. Park *et al.* [17] and [19] have proposed a model with the Markov chains considering the retry limit, acknowledgments and unsaturated traffic. Effective and simple approximation of delay, reliability and energy consumption have been proposed in these works. Furthermore, in [17], based on [19] and [23], a PGF method has been applied to estimate the delay introduced in packet delivery. However, only the star topology was considered in these works. Hence, delay is estimated in a one-hop fashion, whereas the analysis verification is based on simulation rather than experimental results. According to the work in [14], the gap between mathematical modeling and measurement of WSNs cannot be neglected. Although the hardware factors related to delay and experiment results were considered in [19], a more comprehensive statistical characterization of latency has not been provided. Marco *et al.* [24] have proposed an analytical model considering a multi-hop IEEE 802.15.4 network with hidden terminals. Wang *et al.* [25] have worked with multi-hop networks and estimated the end-to-end delay. The work considered the carrier sense multiple access with the collision avoidance (CSMA/CA) protocol for nodes running the TinyOS operating system. However, both of the works above only focused on modeling MAC protocol logic, without considering the implementation factors in the network, i.e., the impact of hardware delay between processors and transceivers in WSNs. Yan *et al.* [26] proposed a mathematical model for MAC access delay. The model was based on the queuing theory and an easy-to-use tool in package level simulation was developed. Nevertheless, that work only focused on the slotted MAC. The unslotted MAC, multi-hop communication and experimental validation were not considered in that work.

In [20], a Markov chain model was proposed to analyze the slotted CSMA/CA in both saturated and unsaturated conditions in the ACK mode. However, the proposed model did not consider retransmissions. Faridi *et al.* [21] proposed a modified Markov model in an attempt to model the slotted CSMA/CA mechanism in the ACK mode. The model also includes retransmissions with finite retry limits.

The packet delay, throughput and power consumption about IEEE 802.15.4 MAC have been analyzed in [27]. However, none of the papers mentioned above considered the following aspects altogether: the unslotted or beaconless MAC, multi-hop scenarios, implementation factors, system level models and software tools for latency estimation. These characteristics make the modeling and validation of the latency of the Thread mesh network different from the results available in the literature.

The contributions of this paper are the following:

1) A thoroughly study of the latency of a new IoT protocol, Thread stack, theoretically based on the model in [18], adding real measurement factors and multi-hop communication.
2) An experimental study of the Thread mesh network through real testbed in BA and validating the proposed analytical model.
3) The specification and implementation of an easy-to-use software tool, which gives clear guidelines to the ProSe developers to guarantee a certain level of QoS and develop user-friendly applications.

III. OVERVIEW OF THE THREAD PROTOCOL

In this section, we describe the protocol stack of Thread as the delay generated by each layer needs to be considered. Specifically, we describe the mechanism of non-beacon enabled IEEE 802.15.4 CSMA/CA with the backoff process. Understanding the Thread protocol and the non-beacon enabled IEEE 802.15.4 CSMA/CA backoff process helps us to propose the system level model in the subsequent sections.

A. PROTOCOL STACK

Thread is a new IP-based wireless networking protocol established in July 2014 by companies such as Google Nest, ARM, and Samsung [6]. The Thread protocol combines the solutions of several standards to fit the needs of typical BA applications. As Fig. 1 shows, Thread does not standardize the application layer. This provides the supplier with the freedom to use proper application layers according to their needs. In our paper, the constrained application protocol (CoAP) is used, which is a constrained web transfer protocol (similar to HTTP) intended for resource-constrained devices. The feature that supports IP networking allows Thread to easily interconnect with other IP-based mobile devices. At the network layer and transport layers, Thread uses User Datagram Protocol (UDP) on top of IP routing and IPv6 over Low power WPAN (6LoWPAN). In MAC and PHY layers, Thread uses the IEEE 802.15.4 wireless specifications, supporting mesh network with a maximum speed of 250 kb/s over the 2.4 GHz band [6]. IEEE 802.15.4 wireless specification has been used in many other protocols such as WirelssHART, Zigbee and Internet Engineering Task Force (IETF) IoT stack.

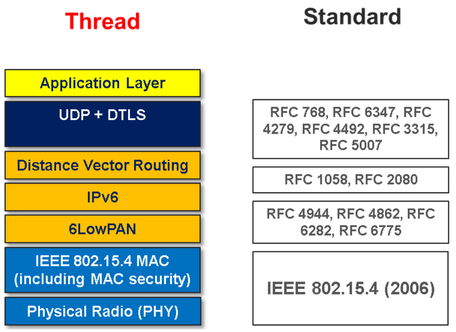

Thread
Standard
Application Layer
UDP + DTLS
Distance Vector Routing
IPv6
6LowPAN
IEEE 802.15.4 MAC (including MAC security)
Physical Radio (PHY)
RFC 768, RFC 6347, RFC 4279, RFC 4492, RFC 3315, RFC 5007
RFC 1058, RFC 2080
RFC 4944, RFC 4862, RFC 6282, RFC 6775
IEEE 802.15.4 (2006)

FIGURE 1. Thread protocol stack [6] (authorized by Thread Group).

There are five types of devices in the Thread network: border routers (BR), routers, router-eligible end device, end devices, and sleepy end devices. The Thread stack supports full mesh connectivity between all routers in the Thread network. The number of routers will decide which kind of topology it will form. If there is only one router or border router, then a basic star topology is formed. If there is more than one router then a mesh topology is automatically formed. The details can be found in [6].

B. IEEE 802.15.4 MAC

In this section, the non-beacon enabled IEEE 802.15.4 CSMA/CA used in the Thread protocol is summarized. Fig. 2 depicts the flowchart of the non-beacon enabled IEEE 802.15.4 CSMA/CA channel access mechanism. We denote the MAC parameters by $W_0 = 2^{macMinBE}$, $m_0 = macMinBE$, $m_b = macMaxBE$, $m = macMaxCSMABackoffs$, and $n = macMaxFrameRetries$. For the Thread protocol, the default parameters are $m_0 = 5$, $m_b = 8$, $m = 4$, and $n = 3$. Upon transmitting packets, the MAC layer performs the following (initialize retransmission counter R to 0, the number of backoff (NB) to 0 and backoff exponential (BE) to *macMinBE*).

1) Before sending the packets, wait for a random period of time in the range $[0, 2^{BE} - 1]$ backoff time unit S_b to avoid collisions. The backoff time unit S_b is 20 T_s with T_s = 16 μs.
2) Perform Clear Channel Assessment (CCA) to check if the channel is busy or idle. If channel is idle, go to Step 4, else go to Step 3.
3) Busy channel: increase the value NB and BE by one. However, the maximum value of BE is *macMaxBE*. At this time, if NB is less than the value of *macMaxCSMABackoffs*, the process returns to step one (Otherwise, the process ends with the channel access failure).

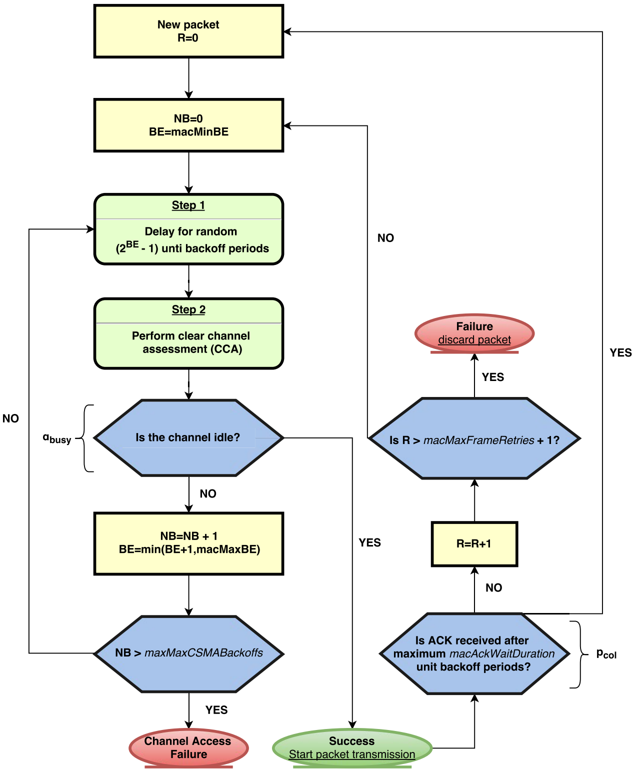


FIGURE 2. The unslotted IEEE 802.15.4 CSMA/CA channel access mechanism considering packet transmissions and retransmissions, adapted from [28].

4) Idle channel: the MAC layer sends out the packet and waits for the ACK. If the ACK is received within maximum *macAckWaitDuration*, the packet is successfully sent. Otherwise, the packet has collided and needs to be retransmitted with retry counter incrementation ($R = R + 1$). If R exceeds *macMaxFrameRetries*, the algorithm terminates with collision fail and the packet is discarded (otherwise the algorithm initializes the NB and BE again and returns to Step 1).

IV. MATHEMATICAL MODEL OF THE THREAD MESH NETWORK

In this section, we propose a system level model for the latency of the Thread mesh network. We analyze the system model considering the implementation of the Thread protocol, from the application layer to the physical layer. In a WSN for BA, we define the round trip time (RTT) as the time duration from the node (border router) sending out a message from the application layer (e.g., CoAP request) until receiving the response from the destination (e.g., CoAP ACK). The parameters used in this paper and a short description of each parameter are provided in Table 1.

A. SYSTEM MODEL OF LATENCY OF MULTI-HOP THREAD MESH NETWORK

TABLE 1. Main symbols used in the paper.

Symbol	Meaning
L	Packet length including overhead and payload
t_{ack}	The ACK waiting time
L_{ack}	The length of ACK frame
IFS	Inter-Frame Spacing
$t_{m,ack}$	The timeout of ACK
L_s	$L_s = L + t_{ack} + L_{ack} + IFS$
L_c	$L_c = L_{L+t_{m,ack}}$
NB	Number of CSMA backoffs
BE	Back-off exponent
T_{RTT}	The total round trip time in the system model
T_0	Propagation time from application layer to physical layer
T_1	Over the air time plus relay's processing time
T_2	Propagation time from physical layer to application layer
T_{IPS0}	Processing time from application layer to MAC layer
T_{MAC0}	MAC layer processing time when sending packet
T_{PHY0}	PHY layer processing time when sending packet
T_{ROUTE}	Forwarding processing time of one relay node
T_{IPS2}	Processing time from MAC layer to application layer
T_{MAC2}	MAC layer processing time when receiving packet
T_{PHY2}	PHY layer processing time when receiving packet
n	Maximum number of retransmission
α_{busy}	Probability that CCA is busy
λ	The network traffic conditions
τ	Node attempts carrier sensing in a randomly chosen time slot
m	Maximum number of backoffs
m_b	Maxinum backoff exponent
m_0	Initial backoff exponent
S_b	Backoff unit time

The system level model is based on the packet transmission path in the Thread mesh network. Without loss of generality, in Fig. 3, the three-hop round trip procedures are reported. The source node is the border router while the destination node is the end device. The RTT can be separated into two parts: the sending route and the receiving route. The sending route begins at the border router's application layer and then goes through the network layer, the MAC layer and the physical layer where the transceiver transmits the message. Then the packet will be in the routing path which consists of the over-the-air (OTA) time and the relay nodes' processing time. When the packet reaches the physical layer of the destination, the destination node starts processing the packet and sending this packet from the physical layer to the application layer. At this point, the sending route is finished while the receiving route begins. The destination node sends back the ACK command following the same route as the sending route.

Based on the description above, RTT is the sum of three terms: the first is the propagation time from the application layer to the physical layer, denoted as T_0; the second is OTA time (send and return paths) plus the relay nodes' processing time T_1; and the third is the propagation time from the physical layer to the application layer T_2. The whole RTT is defined as:

$$T_{\text{RTT}} = 2 \times T_0 + T_1 + 2 \times T_2 \,. \tag{1}$$

Now, we define in details the components of the previous equation. T_0 consists of the processing time from the application layer to the MAC layer, denoted as IP stack (IPS) processing time T_{IPS0}, the MAC layer processing time T_{MAC0} and the physical layer processing time T_{PHY0}, such that:

$$T_0 = T_{\text{IPS0}} + T_{\text{MAC0}} + T_{\text{PHY0}}. \tag{2}$$

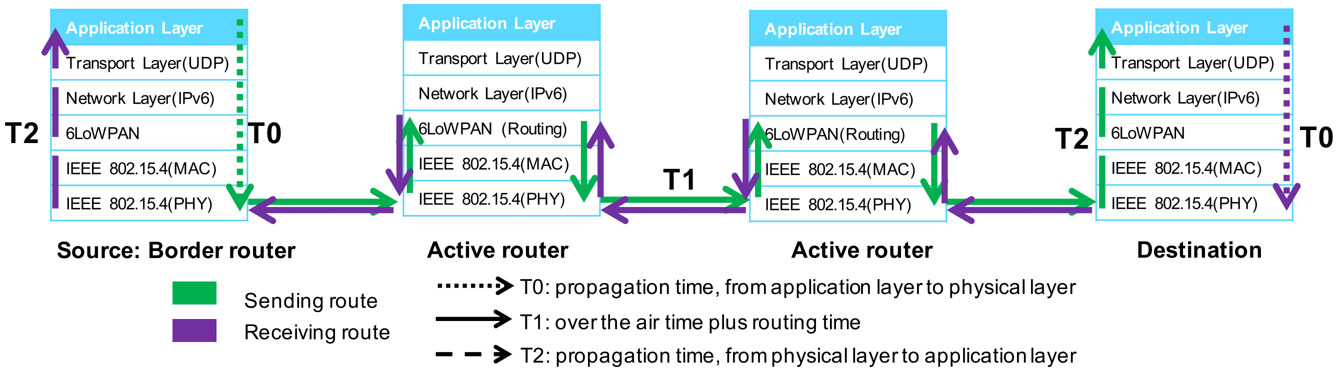


FIGURE 3. System level model of the latency of Thread mesh network.

T_1 includes the OTA time and the processing time in the relay nodes. The OTA time can be neglected as the speed of the wireless waveform in the air is the speed of light. Therefore, T_1 depends on the relay nodes' routing time. When relay nodes are forwarding the message, the packets are firstly entering the physical layer, the MAC layer and then the 6LowPAN layer where the next forwarded nodes are decided. Then forwarded messages will go through the MAC layer and the physical layer to transmit to the next node. Once this node receives the MAC ACK back from the next relay node, the forwarding process of this node is finished. The forwarding processing time of one relay node can be defined as T_{ROUTE}:

$$T_{\mathrm{ROUTE}} = T_{\mathrm{MAC2}} + T_{\mathrm{PHY2}} + T_{\mathrm{PHY0}} + T_{\mathrm{MAC0}}. \tag{3}$$

In the Thread mesh network, we indicate the number of hops by h. The total routing time is related to h. Considering the total routing time in multi-hop networks:

$$T_1 = 2 \times (h - 1) \times T_{\mathrm{ROUTE}}. \tag{4}$$

Similarly, T_2 also consists of the processing time of the physical layer T_{PHY2}, MAC layer T_{MAC2} and the MAC to application layer time T_{IPS2} when receiving the messages. Therefore,

$$T_2 = T_{\mathrm{IPS2}} + T_{\mathrm{MAC2}} + T_{\mathrm{PHY2}}. \tag{5}$$

In conclusion, the T_{RTT} in Eq. (1) is related to the time T_{PHY0}, T_{MAC0}, T_{IPS0}, T_{PHY2}, T_{MAC2}, T_{IPS2} and the number of hops n. T_{PHY0}, T_{IPS0}, T_{PHY2}, T_{MAC2}, and T_{IPS2} are decided by the hardware factors and the implementation of the Thread protocol. T_{MAC0} is the MAC layer service time that is not fixed because of the exponential back-off process in the CSMA/CA mechanism. This random back-off time depends on the traffic rate and the number of nodes in the network. Usually, PGF is used to get the distribution of MAC layer service time. To derive the PGF, two other parameters, the probability of busy channel assessment and packet collision rate need to be calculated as well, which will be discussed in the next two sections.

B. BUSY CHANNEL AND COLLISION PROBABILITY

In Fig. 2, two probabilities are involved in the channel access and data transmission process, the probability of busy channel assessment (α_{busy}) and packet collision rate (p_{col}). In [18], a Markov chain is proposed to model the MAC traffic in a single-hop IEEE 802.15.4 consisting of N devices that communicate with a PAN coordinator. By considering the slotted version of CSMA/CA with retransmission and ACKs, a three dimension Markov chain is proposed to calculate α_{busy} and p_{col}. The acquired results in [18] provide ground work for our system model considering the un-slotted version of IEEE 802.15.4 with retry limits, ACK and single CCA instead of double CCAs.

According to the stochastic process mechanism, the Markov chain model is described by a three dimensional (3D) *{s(t), c(t), r(t)}* process as depicted in Fig. 4. *{s(t), c(t), r(t)}* respectively represent the back-off stage, the state of the back-off counter and the state of the retransmission counter at time t. We assume the probability of nodes starting sensing the channel is independent. Therefore, the stationary probability τ that the node attempts carrier sensing is constant and independent from the other nodes. The protocol MAC parameters are denoted by $\mathbf{V} = (m_0, m, n)$, m_b = macMaxBE, $W_0 = 2^{m_0}$, $W_m = 2^{\min(m_0+m, m_b)}$. The states from $(i, W_m - 1, j)$ to $(i, W_0 - 1, j)$ are the back-off states. States $(i, 0, j)$ represent CCA. State $(-1, k, j)$ and $(-2, k, j)$ represent the successful transmission and the packet collision state, respectively.

With the help of derived transition probabilities and stationary probability in the Markov chain model, α_{busy} and p_{col} are obtained. They are the essential aspects of the delay analysis of CSMA/CA because they determine the likelihood of CSMA backoffs and retransmission. The term α_{busy} consists of the busy channel access probability due to data transmission (α_1) and ACK transmission (α_2).

$$\alpha_{busy} = \alpha_1 + \alpha_2. \tag{6}$$

$$\alpha_1 = L(1 - (1 - \tau)^{N-1})(1 - \alpha_{busy}). \tag{7}$$

$$\alpha_2 = L_{ack} \times \frac{N\tau(1 - \tau)^{N-1}}{1 - (1 - \tau)^N} \times \frac{\alpha_1}{L}. \tag{8}$$

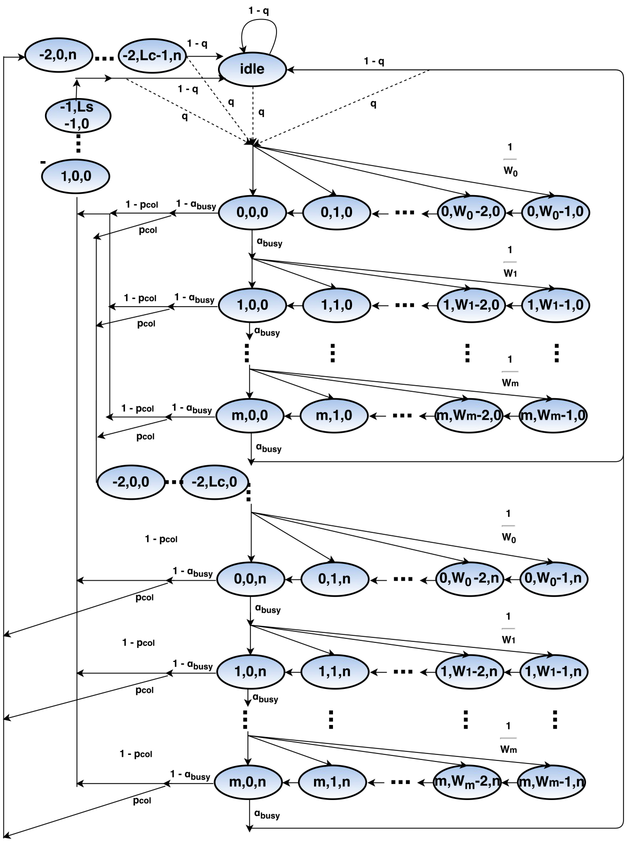


FIGURE 4. Markov chain model of unslotted IEEE 802.15.4 CSMA/CA revised from [18].

The collision probability p_{col} is expressed with Eq. 9.

$$p_{col} = 1 - (1-\tau)^{N-1}. \tag{9}$$

This part rounds off the necessary parameters to derive the probability distribution function of the MAC layer service time. For further details about the terms and the derivation of the equations, the reader can refer to [18] and the original contributor of the Markov Chain [22].

C. DELAY ANALYSIS OF CSMA/CA MECHANISM

In this section, we derive the distribution of MAC layer service time T_{MAC0}. In [23], the PGF of MAC layer service time of IEEE 802.11 was introduced while its revised version adapting to slotted IEEE 802.15.4 was presented in [17]. These two works provide ground works for us to derive the PGF for MAC layer service time of un-slotted IEEE 802.15.4. The MAC layer service time starts at the instant when nodes begin to contend for the transmission and ends at the instant when either the packets are successfully transmitted or the packets are discarded due to channel access failure or retry limits. The distribution is discrete as the back-off interval time is the basic time unit S_b. The PGF *B(Z)* of T_{MAC0} is first derived in order to get the distribution of MAC layer service time. The moment of T_{MAC0} of any degree can be obtained by derivation of *B(Z)*. Below is the mean and variance of T_{MAC0}.

$$E(T_{MAC0}) = B'(1). \tag{10}$$

$$E[(T_{MAC0} - E[T_{MAC0}])^2] = B''(1) + B'(1) - \{B'(1)\}^2. \tag{11}$$

where the prime denotes the derivative with respect to Z. Therefore, we derive the PGFs of T_{MAC0} in the following.

First, we give some definitions as follows:

We denote $S_t(Z)$ as the PGF of the time period for successful transmission, $C_t(Z)$ the PGF of the time period for collision with acknowledgment, $S_c(Z)$ the PGF of the sensing time, and $G_i(Z)$ the PGF of time period of fail sensing due to busy channel. The $S_t(Z)$ and $C_t(Z)$ can be written as:

$$S_t(Z) = Z^{L_s}. \tag{12}$$

$$C_t(Z) = Z^{L_c}. \tag{13}$$

the symbols $L_s = L + t_{ack} + L_{ack} + IFS$ and $L_c = L_{L+t_{m,ack}}$ are the time periods for successful packet transmission and packet collision, listed in Table. 1. Therefore, the time period for successful transmission and packet collision are decided by the transmission rate, the packet length and the overhead, and the specific transmission scheme i.e., DATA/ACK scheme.

From Fig. 4, the PGF of the random back-off process *$H_i(Z)$* can be derived by the production from 0 to *i*-th stage as

$$H_i(Z) = \prod_{k=0}^{i} W_k(Z). \tag{14}$$

$W_k(Z)$ is the PGF of the back-off time at the *i*-th:

$$W_k(Z) = \begin{cases} \Sigma_{l=0}^{2^i W_0 - 1} \dfrac{M_d(Z)^l}{2^i W_0} & \text{if } i \le m_b - m_0 \\ \Sigma_{l=0}^{2^{m_b} - 1} \dfrac{M_d(Z)^l}{2^{m_b}} & \text{otherwise.} \end{cases} \tag{15}$$

$$= \begin{cases} \dfrac{1 - Z^{2^i W_0}}{2^i W_0 (1-Z)} & \text{if } i \le m_b - m_0 \\ \dfrac{1 - Z^{2^{m_b}}}{2^{m_b}(1-Z)} & \text{otherwise.} \end{cases} \tag{16}$$

It is assumed that $M_d(Z) = Z$ as the decrement of the backoff counter happens with probability 1. Then the signal transfer function of i sensing fail due to busy channel condition $G_i(Z)$ is

$$G_i(Z) = \sum_{k=1}^{i} S_c(Z)^i. \tag{17}$$

where $S_c(Z) = Z$ is the sensing time. The next step is to derive the PGF of *Ts*:

Proposition 1: Let $B(Z)$ be the probability generating function of the MAC layer service time; then,

$$
\begin{aligned}
B(Z) = S_t(Z)\sum_{j=0}^{n}\Pr(\mathcal{S}_j|\mathcal{U}_t)C_t(Z)^j \\
\times\left(S_c(Z)\times\sum_{i=0}^{m}C_i(Z)H_i(Z)\right)^{j+1} + \sum_{j=0}^{n}\Pr(\mathcal{C}_j|\mathcal{U}_t) \\
\times\left(C_j(Z)S_c(Z)\sum_{i=0}^{m}G_i(Z)H_i(Z)\right)^{j} G_{m+1}(Z)H_{m+1}(Z) \\
+\Pr(\mathcal{F}_j|\mathcal{U}_t)\left(C_t(Z)\times S_c(Z)\sum_{i=0}^{m}G_i(Z)H_i(Z)\right)^{n+1}.
\end{aligned}
\tag{18}
$$

where

$$\Pr(\mathcal{S}_j|\mathcal{U}_t) = \frac{P_c^j(1-P_c)(1-\alpha_{busy}^{m+1})^{j+1}}{Pr(\mathcal{U}_t)}. \tag{19}$$

$$\Pr(\mathcal{C}_j|\mathcal{U}_t) = \frac{P_c^j(1-\alpha_{busy}^{m+1})^j\alpha_{busy}^{m+1}}{Pr(\mathcal{U}_t)}. \tag{20}$$

$$\Pr(\mathcal{F}_j|\mathcal{U}_t) = \frac{P_c^{n+1}(1-\alpha_{busy}^{m+1})^{n+1}}{Pr(\mathcal{U}_t)}. \tag{21}$$

with,

$$
\begin{aligned}
Pr(\mathcal{U}_t) = \frac{1-(P_c(1-\alpha_{busy}^{m+1}))^{n+1}}{1-P_c(1-\alpha_{busy}^{m+1})}(1-P_c+P_c\alpha_{busy}^{m+1}) \\
+P_c^{n+1}(1-\alpha_{busy}^{m+1})^{n+1}.
\end{aligned}
\tag{22}
$$

and G_i(Z) is given by Eq. (17).

Proof: The PGF of MAC layer service time is derived from the Markov chain model shown in Fig. 4. A transmission commences whenever the node performs CCA successfully. When a packet transmits in the channel, it has the probability P_c of collision. If collision happens, the node initials the new backoff procedure until reaching maximum retransmission time after $C_t(Z)$. The node also has a probability $1-P_c$ to finish the transmission after $S_t(Z)$. $\mathcal{S}_j$ and $\mathcal{C}_j$ denote the event of successful transmissions and discarded packet due to channel access failure after j collisions, respectively. $\mathcal{F}$ is the event of discarded packets due to retry limits and $\mathcal{U}_t$ means all the possible events $\mathcal{S}_j$, $\mathcal{C}_j$, and $\mathcal{F}$. the probabilities $\Pr(\mathcal{S}_j|\mathcal{U}_t)$, $\Pr(\mathcal{C}_j|\mathcal{U}_t)$, and $\Pr(\mathcal{F}_j|\mathcal{U}_t)$ are given by Eqs. (19), (20), and (21), with the normalization of $Pr(\mathcal{U}_t)$. Since a packet transmits after successfully performing one CCA, the probability for successful channel access is $1-\alpha_{busy}^{m+1}$. $H_{m+1}(Z)$ considers the random backoff process while $G_{m+1}(Z)$ is the sensing delay for discarded packets because of channel sensing fails. Now it is possible to derive $B(Z)$ for the packet transmission process shown in Fig. (4). $B(Z)$ is simply the signal transfer function from the start state to the end state as a function of P_c, α_{busy}, m, n and Z. Therefore, $B(Z)$ consists of three parts: the first is the successful packet transmission, and the second and third parts consider the time period of the discarded packet due to channel access failure and retry limits. Combing with Eq. (16)-(17) and (19)-(22), Eq. (18) follows. This concludes the proof.

As B(Z) is a smooth function of Z, it can be written as power series:

$$B(Z) = \sum_{i=0}^{\infty}\Pr(T_{MAC0} = i)Z^i. \tag{23}$$

Therefore, the mean and the variation of the MAC service time T_{MAC0} can be obtained from $B(Z)$ by using Eq. (10) and (11). For further details about the terms and derivation of equations, the reader needs to refer to the works [17] and [18].

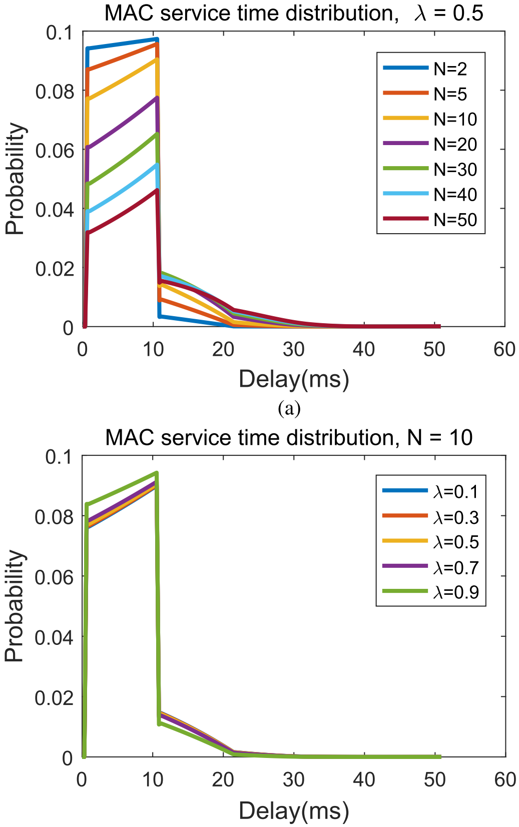
Once T_{MAC0} is known, we can also learn the probability distribution of T_{RTT} according to the Eq. (1) - (5). In the next section, we will experimentally validate the probability distribution of T_{RTT}.

V. NUMERICAL RESULTS AND EXPERIMENTAL VALIDATION

First, we present the numerical results of the distribution of MAC service time we derived in the previous section by observing the number of nodes in the network and the effect of traffic conditions. Then, to quantify the system level model, the hardware related latencies are further considered in the experimental results. Finally, to validate the system level model we presented in the previous section in Eq. (1), we perform experiments which allow us to confirm the analysis. Details are described in the following.

A. STATISTICAL PROBABILITY DISTRIBUTION OF MAC SERVICE TIME

By calculating the analytical equations of the model above, the distribution of MAC service time is presented with respect to the node number N and the network traffic conditions λ. According to the parameters used in the Thread mesh network, the settings are as follows: $m_0 = 5, m = 4$, and $n = 3$. A fixed packet length of $L = 7$ is used in calculations. Fig. 5 (a) shows the probability distribution of MAC service time as a function of different nodes $N = 2, 5, 10, 20, 30, 40, 50$ with given $\lambda = 0.5$, which is drawn by MATLAB. It can be easily observed that with the increase of the node number N, the side lobes of the distribution function also increase and the probability of low latency trapezoid area decreases. The reason is that the increase of node number N will make the network traffics heavier and thus increase both the busy channel probability α_{busy} and collision probability P_c. Fig. 5 (b) presents the relationship between network traffic condition λ and the probability distributions with node number $N = 10$. The heavier the traffic situation in the network, the heavier the tail. This is because increasing traffic rate will also increase the failures in channel sensing and packet transmitting. Besides, node number N has a stronger influence on the probability distributions than λ. For $N = 2$, probability distribution is similar to a deterministic one.



(a)

(b)

FIGURE 5. Numerical results of MAC layer service time distribution. (a) shows the relationship between the probability distributions and the node number N with λ = 0.5. (b) shows the relationship between the probability distributions and λ with N = 10.

B. LATENCY RELATED TO HARDWARE AND SOFTWARE IMPLEMENTATION

In this section, the latency of other layers besides the MAC layer is experimentally investigated, which is closely related to the hardware aspects and implementation of the software stack. First, the hardware platform and the software stack for experimental study are introduced. Second, the experimental tests are presented to understand the latency introduced by hardware and software implementation.

In the experimental study, the hardware board FRDM-KW24D512 and Thread 1.0.0 preview software stack from NXP are used. FreeRTOS is used as the operating system, which may introduce extra latency. Reminding the equation T_{RTT} in Eq. (1), T_{PHY0}, T_{MAC0}, T_{IPS0}, T_{PHY2}, T_{MAC2}, and T_{IPS2} need to be known in order to calculate the total *RTT*. T_{MAC0} is the MAC layer service time distribution with respect to node number N and traffic condition λ. The other parameters will be gained from the experimental evaluation on the test bed.

In the software stack, the time stamps are added to get the time interval of each layer as shown in Fig. 6. At first, the packet length of 10 Bytes is applied in the test. The results are shown in Table 2. Then the packet length with the varying application layer loads of 20 Bytes, 30 Bytes, 40 Bytes, 50 Bytes, and 60 Bytes are tested in turns. The results show that delays of IPS TX and PHY TX are proportional to packet length. Therefore, linear regression is used to estimate latency in IPS TX and PHY TX states, as shown in Fig 7. Here the data length counts only the payload of the application layer, e.g., CoAP's payload. The linear regression model fits well with the testing data both in the physical and IPS TX layers. For the physical layer, the delay comes from the packet copying between the micro-controller and radio transceivers through the serial peripheral interface (SPI), where

$$T_{PHY0} = \left\lceil \frac{L}{vS_b} \right\rceil. \tag{24}$$

For IPS TX, it is decided by the copying speed inside the micro-controller in software stack implementation from the application layer to the network layer. The delay of PHY RX, MAC RX, and IPS RX are constant in our test case as the ACK length is fixed, as shown in Table 2.

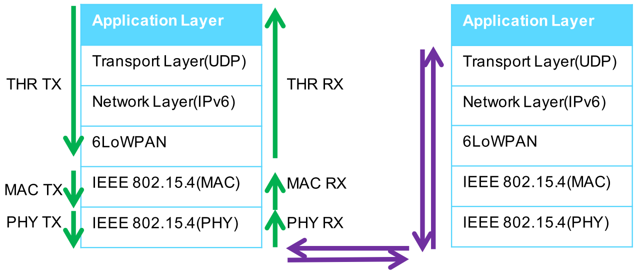
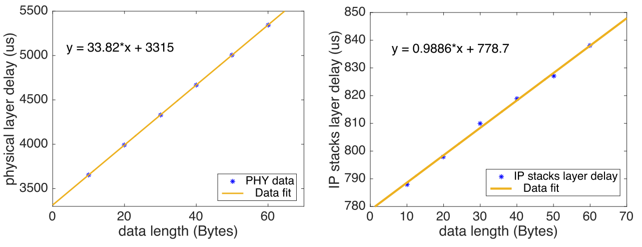


FIGURE 6. Latency of different layers in Thread network.



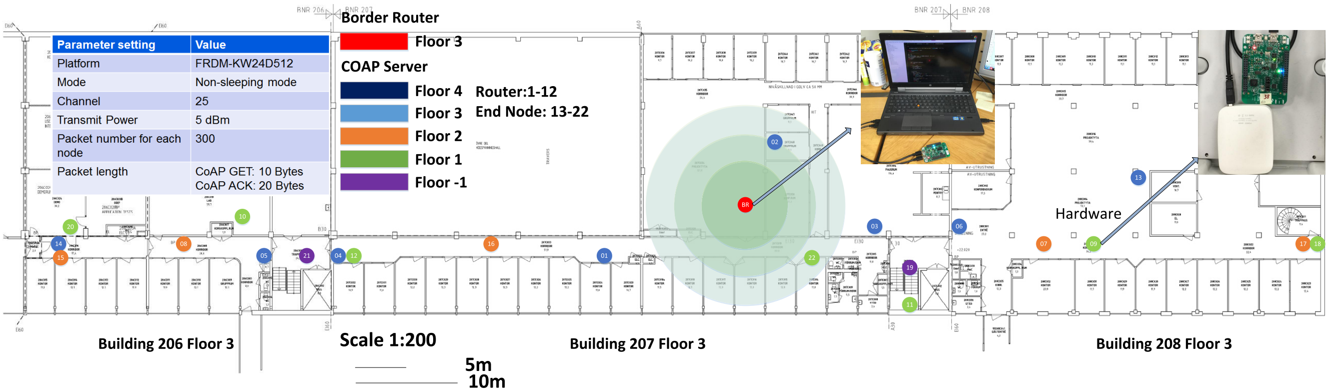
(a)

(b)

FIGURE 7. Experimental test results of physical layer latency and IP stack layer latency.

TABLE 2. Experimental data of the thread latency in different layers with data length 10 bytes.

Average result of 500 test times	delay (μs)
IPS TX	788
MAC TX	5490
PHY TX	3653
PHY RX	266
MAC RX	445
IPS RX	544



Parameter setting	Value
Platform	FRDM-KW24D512
Mode	Non-sleeping mode
Channel	25
Transmit Power	5 dBm
Packet number for each node	300
Packet length	CoAP GET: 10 Bytes CoAP ACK: 20 Bytes

FIGURE 8. Parameters set-up for the testbed and deployment of nodes in the building.

C. EXPERIMENTAL VALIDATION OF THREAD MESH NETWORK

In this section, we experimentally implement a testbed in a real building environment to test the latency of the Thread mesh network and compare experimental results with analytical results.

To establish the Thread multi-hop mesh network, 23 nodes are deployed in four floors, as shown in Fig. 8. The top left side of the figure shows the experimental parameters settings in the test bed. Each node has the ID from 0 to 22. Node 0 is the BR which sends the RTT packets. Nodes 1 to 12 are the active routers while nodes 13 to 22 are the end nodes (ED). In the graph, the red circle in the third floor denotes Thread BR, the root of mesh networks, connecting to our desktop computer. Other nodes are spreading in the building from the base floor to the third floor, mainly in the meeting rooms, lounges and corridors. When the test begins, BR starts sending CoAP GET messages to each node from node 1 to 22. The BR uses its hardware clock to calculate the time interval between sending the message and receiving the ACK. BR sends 300 packets to each node and also records how many hops to reach that node. All the RTT data are divided into six groups according to the hops. For each hop, the cumulative distribution function (CDF) curves are drawn to represent the RTT based on hop counts.

Fig. 9 shows the experimental CDF results of the latency of six hops in the Thread mesh network. The probabilities of RTT within 200 ms from one to six hops are near 0.99. In general, RTT increases with the number of hops. The CDF curve interval between their neighbors are close to each other. As the red square shapes note, there exists stair effects in the CDF curve because of CoAP retransmissions. When the first packet drops, CoAP retransmissions are trigged after roughly two seconds and can be trigged up to four times. Fig. 9 shows a small number of packets that experience very long latency once CoAP retransmission is triggered due to packet loss in the lower layers (caused by the negative factors like fading, interference, and path loss). There are more CoAP retransmissions for high number of hops, such as six hops. These long latency packets count less than one percent in the total experiment packets. Note that our proposed analytical model does not consider the application layer retransmission. Thus, we regard these long latency packets as dropout packets.

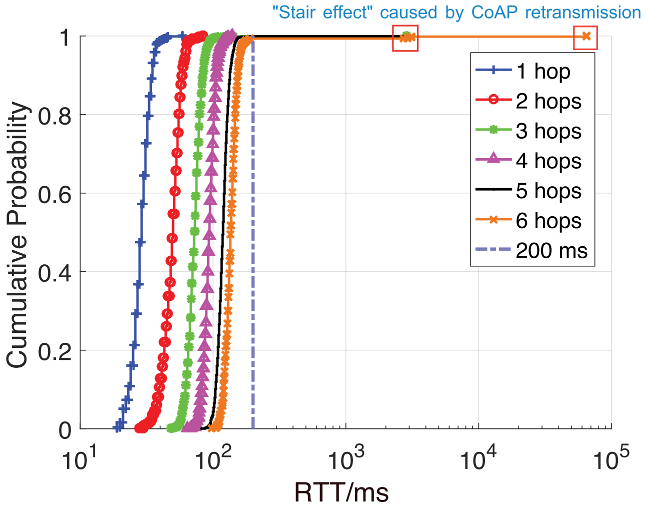


FIGURE 9. Experimental results on Cumulative Distributed Function versus Round Trip Time for Thread mesh network.

Recall that the latency of the Thread mesh network can be calculated using the Eq. (1). For h-hop networks, the transmitting delay in MAC layer of all nodes can be obtained using mathematical convolutions of a single hop probability distribution of MAC service time (T_{MAC0}) by h times . The other delays besides MAC TX can be referred in Section V. Thus, combining the delay of MAC layer and other layer delays, the results of the system model are showing in Fig. 10, both analytical and experimental results on the probability distribution of the latency of the Thread mesh network. It should be noticed that the packets with long latency caused by CoAP retransmission were not plotted in Fig. 10 for the sake of keeping the focus of the modeling part. If these packets were plotted, we would see a few dots with very low amplitude at the far right end of the X axis. The limitation of not considering the application layer retransmission will be investigated in the future work. With such constraint of the model, the latency model fits well with the testing result with

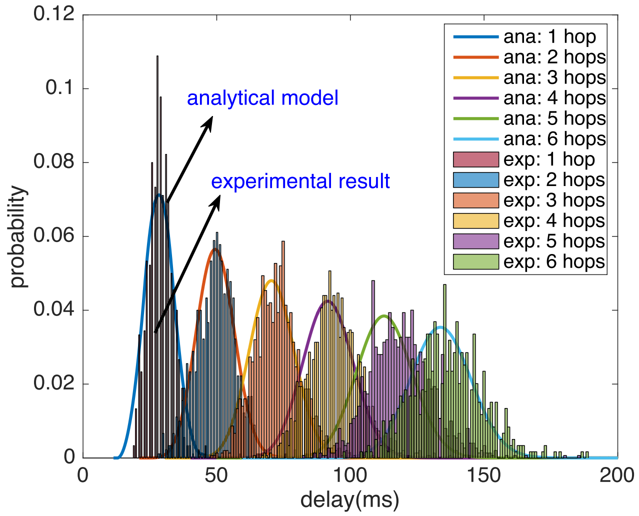
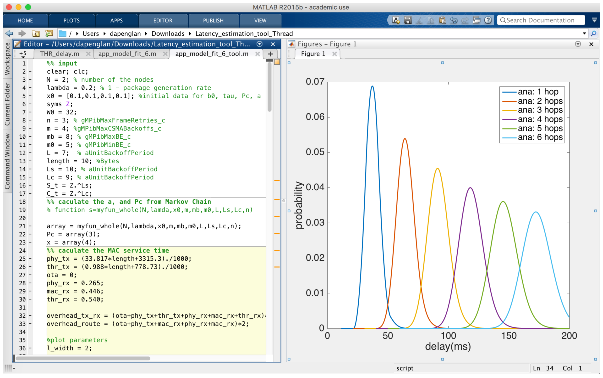


FIGURE 10. **Analytical and experimental results on the probability distribution of the latency of Thread mesh network.**



```
%% input
clear; clc;
N = 2; % number of the nodes
lambda = 0.2; % 1 - package generation rate
x0 = [0.1,0.1,0.1,0.1]; %initial data for b0, tau, Pc, a
syms Z;
W0 = 32;
n = 3; % gMPibMaxFrameRetries_c
m = 4; %gMPibMaxCSMABackoffs_c
mb = 8; % gMPibMaxBE_c
m0 = 5; % gMPibMinBE_c
L = 7;  % aUnitBackoffPeriod
length = 10; %Bytes
Ls = 10; % aUnitBackoffPeriod
Lc = 9; % aUnitBackoffPeriod
S_t = Z.^Ls;
C_t = Z.^Lc;
%% caculate the a, and Pc from Markov Chain
% function s=myfun_whole(N,lamda,x0,m,mb,m0,L,Ls,Lc,n)

array = myfun_whole(N,lambda,x0,m,mb,m0,L,Ls,Lc,n);
Pc = array(3);
x = array(4);
%% caculate the MAC service time
phy_tx = (33.817*length+3315.3)./1000;
thr_tx = (0.988*length+778.73)./1000;
ota = 0;
phy_rx = 0.265;
mac_rx = 0.446;
thr_rx = 0.540;

overhead_tx_rx = (ota+phy_tx+thr_tx+phy_rx+mac_rx+thr_rx)
overhead_route = (ota+phy_tx+mac_rx+phy_rx+mac_rx)*2;

%plot parameters
l_width = 2;
```

FIGURE 11. **Software tool to estimate the latency of Thread mesh network. Left side is used to configure the parameters and the right side shows the analytical results.**

respect to different hops. It can be noticed that the distribution of RTT is mainly decided by MAC back-off time.

D. SOFTWARE TOOL TO ESTIMATE THE LATENCY OF THREAD MESH NETWORK

In this section, a software tool to estimate the latency of the Thread mesh network is implemented using MATLAB R2015B. As Fig. 11 shows, the parameters N, λ, m, m_b, m_0, L, Ls, and Lc can be configured according to the setup of the Thread mesh network. The right side is the figure of the estimation of RTT according to the parameter configuration. It depicts the probability distribution of RTT from one to six hops.

Characterizing latency in smart home and BA applications is an essential aspect of a sound engineering design. Therefore, one important goal is to guarantee QoS in terms of message latency, such as switching the lights or door control within 200 milliseconds (ms). ProSe developers generally lack the information of network delays when developing services, which significantly increases the cost of development and time to market [29]. This tool can solve the issue above. First, this tool reveals the relationships between MAC parameters setting and the latency of the Thread mesh network. Second, this tool gives guidelines for developers to design high quality services and develop user-friendly ProSe.

VI. CONCLUSION AND FUTURE WORKS

In this paper, we investigated the wireless network protocol called Thread for ProSe in smart home and BA, and proposed a theoretical system level model to investigate its latency. Markov chain process and generating functions are used to model the behavior of backoff process in CSMA/CA and to derive the probability distribution of the MAC layer service time. In addition, an experimental validation of the model is carried out showing a good match with the system level model. Based on system level model, we have developed a friendly software tool for estimating latency of Thread mesh network so that the developers can have more information about the communication infrastructures, especially latency, when developing user-friendly ProSe.

However, the system model and experiment results have some limitations. The model does not consider the application layer retransmissions. The future work will include considering this factor in the model. Investigating latency of the multicast communication will also be part of the future work. Also, more complete survey about wireless networks with improved responsiveness for ProSe will also be the future job.

REFERENCES

[1] Y. Wang, L. Wei, A. V. Vasilakos, and Q. Jin, "Device-to-device based mobile social networking in proximity (MSNP) on smartphones: Framework, challenges and prototype," *Future Gener. Comput. Syst.*, vol. 74, pp. 241–253, Sep. 2017.

[2] Y. Wang, A. V. Vasilakos, Q. Jin, and J. Ma, "Survey on mobile social networking in proximity (MSNP): Approaches, challenges and architecture," *Wireless Netw.*, vol. 20, no. 6, pp. 1295–1311, 2014.

[3] S. Mumtaz, A. Alsohaily, Z. Pang, A. Rayes, K. F. Tsang, and J. Rodriguez, "Massive Internet of Things for industrial applications: Addressing wireless IIoT connectivity challenges and ecosystem fragmentation," *IEEE Ind. Electron. Mag.*, vol. 11, no. 1, pp. 28–33, Mar. 2017.

[4] V. K. L. Huang, Z. Pang, C.-J. A. Chen, and K. F. Tsang, "New trends in the practical deployment of industrial wireless: From noncritical to critical use cases," *IEEE Ind. Electron. Mag.*, vol. 12, no. 2, pp. 50–58, Jun. 2018.

[5] S. Majumder *et al.*, "Smart homes for elderly healthcare—Recent advances and research challenges," *Sensors*, vol. 17, no. 11, p. 2496, 2017.

[6] *Thread Group*. Accessed: Dec. 27, 2018. [Online]. Available: http://www.threadgroup.org/

[7] Y. Liu, Z. Pang, G. Dán, D. Lan, and S. Gong, "A taxonomy for the security assessment of IP-based building automation systems: The case of thread," *IEEE Trans. Ind. Informat.*, vol. 14, no. 9, pp. 4113–4123, Sep. 2018.

[8] E. Azoidou, Z. Pang, Y. Liu, D. Lan, G. Bag, and S. Gong, "Battery lifetime modeling and validation of wireless building automation devices in thread," *IEEE Trans. Ind. Informat.*, vol. 14, no. 7, pp. 2869–2880, Jul. 2018.

[9] X. Jiang *et al.*, "Low-latency networking: Where latency lurks and how to tame it," *Proc. IEEE*, vol. 30, no. 99, pp. 1–27, Aug. 2018.

[10] M. Doudou, D. Djenouri, and N. Badache, "Survey on latency issues of asynchronous MAC protocols in delay-sensitive wireless sensor networks," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 2, pp. 528–550, 2nd Quart., 2013.

[11] M. Anwar, Y. Xia, and Y. Zhan, "TDMA-based IEEE 802.15.4 for low-latency deterministic control applications," *IEEE Trans Ind. Informat.*, vol. 12, no. 1, pp. 338–347, Feb. 2016.

[12] P. K. Sahoo, S. R. Pattanaik, and S.-L. Wu, "Design and analysis of a low latency deterministic network MAC for wireless sensor networks," *Sensors*, vol. 17, no. 10, p. 2185, 2017.

[13] G. Daneels, B. Spinnewyn, S. Latré, and J. Famaey, "ReSF: Recurrent low-latency scheduling in IEEE 802.15.4e TSCH networks," *Ad Hoc Netw.*, vol. 69, pp. 100–114, Feb. 2018.

[14] F. Despaux, Y.-Q. Song, and A. Lahmadi, "On the gap between mathematical modeling and measurement analysis for performance evaluation of the 802.15.4 MAC protocol," in *Proc. 12th Int. Workshop Real-Time Netw. (RTN)*, Paris, France, Jul. 2013, pp. 7–8.

[15] F. Österlind and A. Dunkels, "Approaching the maximum 802.15. 4 multi-hop throughput," in *Proc. 5th ACM Workshop Embedded Netw. Sensors (HotEmNets)*, 2008, pp. 5–6.

[16] F. Despaux, "Modeling and evaluation of the end-to-end delay in wireless sensor networks," Ph.D. dissertation, Dept. Inf. Technol., Univ. Lorraine, Nancy, France, 2015.

[17] P. Park, P. Di Marco, C. Fischione, and K. H. Johansson, "Delay analysis of slotted IEEE 802.15. 4 with a finite retry limit and unsaturated traffic," in *Proc. IEEE Global Commun. Conf.*, 2009, p. 18.

[18] P. Park, P. D. Marco, P. Soldati, C. Fischione, and K. H. Johansson, "A generalized Markov chain model for effective analysis of slotted IEEE 802.15.4," in *Proc. IEEE Int. Conf. Mobile Adhoc Sensor Syst.*, Oct. 2009, pp. 130–139.

[19] P. Park, P. D. Marco, C. Fischione, and K. H. Johansson, "Modeling and optimization of the IEEE 802.15.4 protocol for reliable and timely communications," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 3, pp. 550–564, Mar. 2013.

[20] J. Misic and V. Misic, *Wireless Personal Area Networks: Performance, Interconnection and Security With IEEE 802.15.4*, vol. 1. Hoboken, NJ, USA: Wiley, 2008.

[21] A. Faridi *et al.*, "Comprehensive evaluation of the IEEE 802.15.4 MAC layer performance with retransmissions," *IEEE Trans. Veh. Technol.*, vol. 59, no. 8, pp. 3917–3932, Oct. 2010.

[22] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 3, pp. 535–547, Mar. 2000.

[23] H. Zhai, Y. Kwon, and Y. Fang, "Performance analysis of IEEE 802.11 MAC protocols in wireless LANs: Research articles," *Wireless Commun. Mobile Comput.*, vol. 4, no. 8, pp. 917–931, 2004.

[24] P. D. Marco, P. Park, C. Fischione, and K. H. Johansson, "Analytical modeling of multi-hop IEEE 802.15.4 networks," *IEEE Trans. Veh. Technol.*, vol. 61, no. 7, pp. 3191–3208, Sep. 2012.

[25] Y. Wang, M. Vuran, and S. Goddard, "Cross-layer analysis of the end-to-end delay distribution in wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 20, no. 1, pp. 305–318, Feb. 2012.

[26] H. Yan, Y. Zhang, Z. Pang, and L. D. Xu, "Superframe planning and access latency of slotted MAC for industrial WSN in IoT environment," *IEEE Trans Ind. Informat.*, vol. 10, no. 2, pp. 1242–1251, May 2014.

[27] J. Zheng and M. J. Lee, "Will IEEE 802.15.4 make ubiquitous networking a reality?: A discussion on a potential low power, low bit rate standard," *IEEE Commun. Mag.*, vol. 42, no. 6, pp. 140–146, Jun. 2004.

[28] *Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)*, IEEE Standard 802.15.4, 2006.

[29] Y. Wang, L. Wei, Q. Jin, and J. Ma, "AllJoyn based direct proximity service development: Overview and prototype," in *Proc. IEEE 17th Int. Conf. Comput. Sci. Eng. (CSE)*, Dec. 2014, pp. 634–641.

DAPENG LAN received the B.Eng. degree in microelectronics from Sun Yat-sen University, Guangzhou, China, in 2014, the M.Sc. degree in ICT innovation from the KTH Royal Institute of Technology, Stockholm, Sweden, in 2016, and the M.Sc. degree in innovation in information and communication technology from the Technical University of Berlin, Berlin, Germany, in 2017. In 2017, he was with InnoEnergy, where he was involved in smart building energy management in Sweden. He was a thesis student with the ABB Corporate Research Center, Västerås, Sweden, in 2016. He is currently a Ph.D. Research Fellow with the Department of Informatics, University of Oslo, Norway. His research interests include fog computing, the Internet of Things, and distributed systems.

ZHIBO PANG received the B.Eng. degree in electronic engineering from Zhejiang University, Hangzhou, China, in 2002, the M.B.A. degree in innovation and growth from the University of Turku, Turku, Finland, in 2012, and the Ph.D. degree in electronic and computer systems from the Royal Institute of Technology (KTH), Stockholm, Sweden, in 2013. He is currently a Principal Scientist on Wireless Communications with ABB Corporate Research, Västerås, Sweden, leading projects in digitalization solutions for smart buildings and homes; robotics and factories; healthcare and logistics; and power electronics and power systems. He is also an Affiliated Faculty with KTH. Before joining ABB, he was the Co-Founder and CTO of startups, such as Ambigua Medito AB. He is a Senior Member of the IEEE and the Co-Chair of the Technical Committee on Industrial Informatics. He received the 2016 Inventor of the Year Award from ABB Corporate Research Sweden. He is an Associate Editor of the IEEE Transactions on Industrial Informatics and the IEEE Journal of Biomedical and Health Informatics, and a Guest Editor of the Proceedings of the IEEE, the IEEE Internet of Things Journal, and the IEEE Reviews in Biomedical Engineering.

CARLO FISCHIONE received the Ph.D. degree in electrical and information engineering and the Laurea degree in electronic engineering *(summa cum laude)* from the University of L'Aquila, Italy, in 2005 and 2001, respectively. He was a Visiting Professor with the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2015, an Associate Professor with Harvard University, Cambridge, MA, USA, in 2015, and a Visiting Scholar, from 2004 to 2005, and a Research Associate, from 2007 to 2008, with the University of California at Berkeley, Berkeley, CA, USA. He is currently a tenured Associate Professor in electrical engineering and computer science with the Department of Network and Systems Engineering, Royal Institute of Technology, Stockholm, Sweden. He has also offered his advice as a consultant to numerous technology companies, such as ABB Corporate Research, the Berkeley Wireless Sensor Network Lab, Ericsson Research, Synopsys, and the United Technology Research Center. He is the Co-Founder and Scientific Director of MIND Music Labs. He has co-authored over 180 publications, including a book, book chapters, international journals and conferences, and international patents. His research interests include optimization with applications to networks, wireless and sensor networks, the Internet of Things, and digital systems. He is an Ordinary Member of the Italian Academy of the History Deputazione Abruzzese di Storia Patria. He received or co-received a number of awards, such as the IEEE Communication Society S. O. Rice Best Paper Award of 2018 for the best IEEE Transactions on Communications paper, the Best Paper Award of the IEEE Transactions on Industrial Informatics, in 2007, the Best Paper Awards at the IEEE International Conference on Mobile Ad-hoc and Sensor System 2005 and 2009, the Best Paper Award of the IEEE Sweden VT-COM-IT Chapter, in 2014, the Best Business Idea Awards from VentureCup East Sweden, in 2010, and from Stockholm Innovation and Growth Life Science in Sweden, in 2014, the Ferdinando Filauro Award from the University of LAquila, Italy, in 2003, the Higher Education Award from the Abruzzo Region Government, Italy, in 2004, the Junior Research Award from the Swedish Research Council, in 2007, and the Silver Ear of Wheat Award in history from the Municipality of Tornimparte, Italy, in 2012. He is an Editor of the IEEE Transactions on Communications and an Associate Editor of *IFAC Automatica*.

YU LIU received the B.Eng. degree in electronics science and technology from the Harbin Institute of Technology, Weihai, China, in 2014, the M.Sc. degree in computer science from the University of Trento, Trento, Italy, in 2016, and the M.Sc. degree in innovation in information and communication technology from the Technical University of Berlin, Berlin, Germany, in 2017. He is currently pursuing the Ph.D. degree with the Department of Science and Technology, Linköping University, Sweden. His research interests include the cloud-based Internet of Things solution, embedded systems, and wireless sensor networks.

AMIR TAHERKORDI received the Ph.D. degree from the Department of Informatics, University of Oslo, in 2011. He is a Researcher with the Networks and Distributed Systems Group, Department of Informatics, University of Oslo. His research interests include distributed computing and software engineering aspects of emerging technologies, such as the Internet of Things, clouds and fogs/edges, cyber-physical systems, and smart grids. His current research interests include software architectures, programming abstractions, service distribution, and middleware systems for the Internet of Things, as well as adaptation of middleware solutions for multicloud applications (EU H2020 MELODIC project). He has experience from several Norwegian and EU research projects and has published several articles in high-ranked conferences and journals. He was selected as a young talented researcher by the Norwegian Research Council, in 2017, to work on a novel IoT service computing model for future Fog-Cloud computing systems (NFR DILUTE Project).

FRANK ELIASSEN is currently a Professor with the Department of Informatics, University of Oslo. He is an experienced researcher and a project manager for several decades in the areas of distributed systems' middleware and the IoT/cyber-physical systems (CPS) with experience from national and EU level projects. His present research interests include service-oriented IoT/edge/fog computing and CPS middleware and programming models in application areas, including smart cities and smart grids, adaptive software systems, autonomic systems, peer-to-peer systems, and cooperative micro-grids.