Prioritized Data Transmission Mechanism for IoT

Changsu Jung[1*]
[1] School of Computer Science and Engineering, Kyungpook National University,
Daegu, 41566, Republic of Korea
[e-mail: jungchangsu@gmail.com]
*Corresponding author: Changsu Jung

Received Septebmer 5, 2019; revised February 23, 2020; accepted March 23, 2020; published June 30, 2020

Abstract

This paper proposes a novel data prioritization and transmission mechanism to minimize the number of packets transmitted and reduce network overload using the constrained application protocol (CoAP) in resource-constrained networks. The proposed scheme adopts four classification parameters to classify and prioritize data from a sensor. With the packet prioritization scheme, the sensed data having the lowest priority is only delivered using the proposed keep-alive message notification to decrease the number of packets transmitted. The performance evaluation demonstrates that the proposed scheme shows the improvement of resource utilization in energy consumption, and bandwidth usage compared with the existing CoAP methods. Furthermore, the proposed scheme supports quality-of-service (QoS) per packet by differentiating transmission delays regarding priorities.

Keywords: Constrained Application Protocol (CoAP); 6LoWPAN; Bluetooth Low Energy; Prioritization, Internet of Things (IoT)

http://doi.org/10.3837/tiis.2020.06.002 **ISSN : 1976-7277**

1. Introduction

Internet of Things (IoT) and Machine-to-Machine (M2M) are prevalent technologies to link things and support the connectivity of the Internet to diverse devices. With those technologies, many applications extend to numerous applications such as environment and bosy sensor monitoring, smart home and smart city management, and the traffic monitoring [1]. Additionally, as forecasted by the development of IoT, trillions of devices are going to be linked to the Internet [2].

There are problems with IoT devices yet to be solved because of low channel capacity, limited bandwidth, inadequate computation ability, and exhaustible power, even though the aggressive growth of IoT connectivity. Numerous efforts have been conducted to solve these restricted environments. Particularly, IPv6 over low-power wireless personal area network (6LoWPAN) is already initiated to establish a low power wireless networks for resource-limited devices in IoT domains [3].

Initially, 6LoWPAN is suggested for IEEE 802.15.4. However, IPv6 over Networks of Resource-Constrained Nodes (6Lo) has broadened IPv6 connectivity to other resouce-limited networks recently [1]. Also, to aid Internet connectivity utilizing the IPv6 protocol, Internet Engineering Task Force (IETF) assists the IPv6 adaptation layer and Internet Protocol Support Profile (IPSP) 1.0 [4] for Bluetooth Low Energy (BLE) 4.1 [5].

Additionally, the Constrained Application Protocol (CoAP) established on Representational State Transfer (REST) composition is presently assimilated by the IETF Constrained RESTful Environments (CoRE) working group [6]. CoAP performs in the 6LoWPAN protocol and is a light application protocol with shorter header overhead than HyperText Transfer Protocol (HTTP) [7]. Hence, CoAP was designed to be compatible with HTTP and provide devices with RESTful functionality in M2M [8, 9, 10, 11, 12].

CoAP has a extension of observing a resource. After establishing observation relationship between a client and a server, a CoAP client constantly receives the notification of resource change by a CoAP server [13, 14]. The CoAP observation is useful to oversee a resource without a recurring request.

Fog computing, cloud computing, and edge computing technologies are introduced to process vast amounts of data as innumerable devices are linked to the Internet to measure the real world [15, 16, 17, 18, 19]. Although the mentioned devices regularly produce a modest number of sensed data from different kinds of sensors, cloud servers and IoT servers obtain extensive amounts of traffic [20]. Accordingly, more memory capacities, higher processing abilities, and bandwidth are required by the IoT systems to handle these data.

It is appreciable that a large number of packets will be generated from devices and traffic will expand immensely in recent years, considering the rapid rise of IoT devices. Furthermore, there will be an overflow in IoT networks distinctive in insufficient power, low bandwidth, and channel capacity, due to the excessively heightened traffic [1]. Therefore, managing huge amounts of data in IoT and M2M networks is a great challenge recently [21, 22, 23, 24].

This paper presents an innovative prioritized data transmission (PDT) system to provide a solution of the exponential growth of traffic problem in the resource-limited networks without additional mechanism and framework. The prime aim of this paper is to decrease the traffic overload and enhancing bandwidth availability without missing the essential data in the recurrent monitoring applications.

The proposed mechanism prioritizes data collected from sensors using parameters, that is, threshold values, a tolerance value, and a sudden change value.

The prospective system reduces the number of transmitted packets and provides packets with quality-of-service (QoS) by supplying differentiated transmission delay with the prioritized data. Additionally, the performance of the proposed framework is assessed in terms of power usage, supporting QoS, transmission latency, and packet transmission rate comparable to the current CoAP approaches.

The remainder of this paper consists of seven sections. Section 2 describes CoAP overview and Section 3 details related works. System architecture of the proposed scheme is depicted in Section 4. The schemes of packet prioritization and transmission are elaborated on Section 5. Section 6 demonstrates the evaluation environments and Section 7 evaluates the performance of the proposed mechanism compared with CoAP approaches. Section 8 presents the conclusion.

2. CoAP Overview

CoAP is designed as the Internet application protocol for devices and networks with minimal resources [25]. This protocol is essentially drafted for M2M applications with unsatisfactory computation power, insufficient bandwidth, and inadequate memory scope. CoRE working group has been building up CoAP as an alternative of HTTP for constrained networks and is still upgrading CoAP by IETF. Since, the broadly dispersed HTTP is irrelevant for constrained devices due to its long packet size, and connection-oriented communication attribute [7, 26].

CoAP provides a Confirmable message (CON) for a reliable message transmission scheme. At the time a client delivers a CON message with a Uniform Resource Identifier (URI) for a resource, a server replies a piggybacked response, an Acknowledge message (ACK) containing the requested resource with an identical message ID. Non-confirmable message (NON) is used for the best-effort delivery that a request does not require an acknowledgement and a notification from a server [27].

Moreover, CoAP supplies an extension to obtain the notification regularly after an interested resouce is enrolled in a CoAP server. Whenever a resource is modified, the CoAP observe extension permits a client to observe a resource in a CoAP server as illustrated in **Fig. 1**. However, CoAP specification [13] does not define the guideline for criteria or theresholds to observe the resouce change.

CoAP exhausts less power using tiny binary header and modest packet size comparable to HTTP so CoAP is a suitable solution for IoT applications. Additionally, The characteristics of CoAP supporting interoperability with HTTP and RESTful service are well associated with pervasive HTTP applications. Nevertheless, CoAP is inadequate in providing QoS, since only confirmable and non-confirmable messages are given for QoS. Moreover, message queues are not provided, so it is demanding to manage a huge amount of packets from large-scale devices.

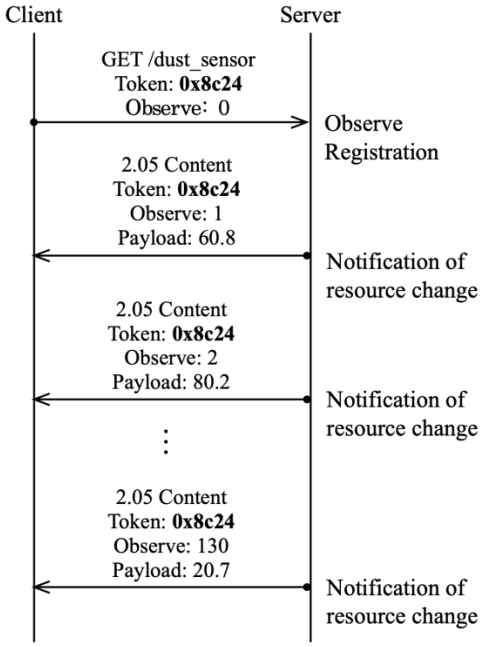


Fig. 1. Notification of CoAP observe scheme

3. Related Work

This section reviews researches regarding the architecture of a cross-protocol proxy and its operation to provide interoperability between HTTP and CoAP. Additionally, the performance of constrained devices and networks are analyzed.

The plan and implementation of a CoAP proxy in Wireless Sensor Networks (WSN) are proposed to aid in the correlation between CoAP and HTTP protocol [28]. The proxy has two protocol stacks for 6LoWPAN over IEEE 802.15.4 and IP over IEEE 802.3 to exchange CoAP and IP packets. The paper also proposed a new URI format for converting HTTP to CoAP and vice versa. The performance of long-term and short-term communication by HTTP and WebSocket is evaluated regarding latency and memory utilization. The outcome indicates that WebSocket protocol functions better in long-lived communications in memory usage and transmission latency. Also, there is no discrepancy in short-period communications

The paper discusses the cross-protocol proxy, a means of interpreting HTTP to CoAP conversely at a transitional node [29]. Both HTTP and CoAP are not switchable since they have dissimilar URI systems. As a result, the study denotes the standardized cross URI and the embedded cross method, as well as the entire URIs of two protocols in a particular URI for equivalent matching between them. Additionally, HTTP/IPv4 to CoAP/IPv6 translation is proposed.

Teklemariam et al [30] suggested a conditional data collection approach as a CoAP extension. The restrictive observation method implements a client-initiative methodology where a CoAP client transmits a request message with the standards of observation. A CoAP server only conveys notification messages, whenever the alteration in the CoAP server realizes the client's criteria. A number of criteria for the conditional observation are specified as response times and threshold values. The research enhanced battery usage and network lifetime than the existing CoAP approach by reducing the transmitted packets to a client using

CoAP. However, the proposed conditional observation approach demands the change of CoAP protocol.

In [31], the authors projected a system of managing numerous entries of CoAP observation with delivering an accumulated packet at a proxy in WSN. Several resource notifications are combined into a packet and sent to a destination. Also, the research designs an adaptive algorithm to find the most appropriate proxy throughout the registration and notification process.

A QoS support system was submitted to timely transmit packets using transmission priority in CoAP networks [32]. Based on the characteristic of resources, the proposed scheme categorizes notifications as urgent and non-urgent. A distinct degree of QoS related to critical or non-critical notice is demanded by a client, as the client (observer) needs a distinctive notification cutoff. Each of the four priority levels of the planned scheme is categorized based on the punctuality of delivery. Furthermore, to lessen the lag in transmission for the notification with higher priority, the server distinguishes the order of delivery, in line with the importance of the notification when a server conveys notification.

A mobile e-health system was designed and implemented using European Telecommunications Standards Institute (ETSI) open standardard [33]. The system is composed of Bluetooth wearable devices, smartphones and M2M middlewares. They measured the latency of end-to-end devices including intermediate devices in the constrained IoT appliction platform using timestamp and round-trip time. The contribution of the research is to identify the most critical components causing the largest delay in the delay sensitive e-health application.

A research evaluated the performance of CoAP and Message Queuing Telemetry Transport (MQTT) protocols over narrowband IoT (NB-IoT) link [34]. Notably, CoAP and MQTT user User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) in the transmission layer respectively. The research measured the throughput and service availability of the two protocols with different packet sizes. MQTT shows lower throughput than CoAP due to the three-way handshake of TCP in MQTT. Moreover, lower throughput causes more energy consumption in MQTT protocol. The service availability meant the delay of delivery within 20 seconds from a device to the cloud. CoAP represents a higher percentage of servicce availability than MQTT as the traffic loads increases because of TCP retransmission overhead in MQTT.

4. System Architecture

A description of the prospective framework comprising of CoAP servers, CrossProxy, and web applications is exhibited in **Fig. 2**. The responsibilities of a CoAP server is to measure the dust density and classify packets for assigning a priority to the sensed data. Meanwhile, CrossProxy servers as a proxy and a delegate to forward request and response messages between a web application and sensor nodes. When BLE connections are established with the 6LoWPAN adaptation layer, CoAP servers are in communication with CrossProxy. When an HTTP request is delivered to CrossProxy by a web application, CrossProxy responds with the demanded resource utilizing HTTP. The succeeding sections describe more thorough information concerning the CoAP server andCrossProxy.

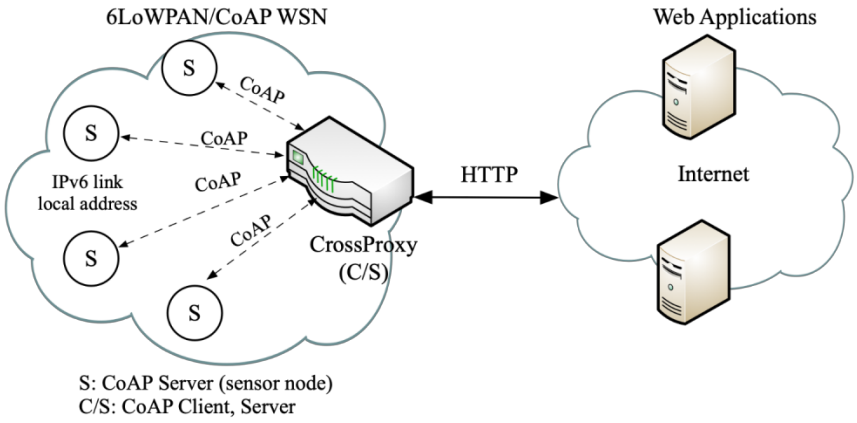


Fig. 2. Network architecture

4.1 Protocol Stacks

This study utilizes CoAP for an application protocol in sensor nodes because of low power exhaustion and its protocol simplicity. 6LoWPAN over BLE is applied to support IPv6 connectivity between sensor nodes and CrossProxy.

CrossProxy contains two different protocol stacks and has BLE and IEEE 802.3 (Ethernet) network interfaces as illustrated in **Fig. 3**. For the interaction between CrossProxy and CoAP servers, BLE interface is utilized, while the communication between a web application and CrossProxy uses IEEE 802.3 interface.

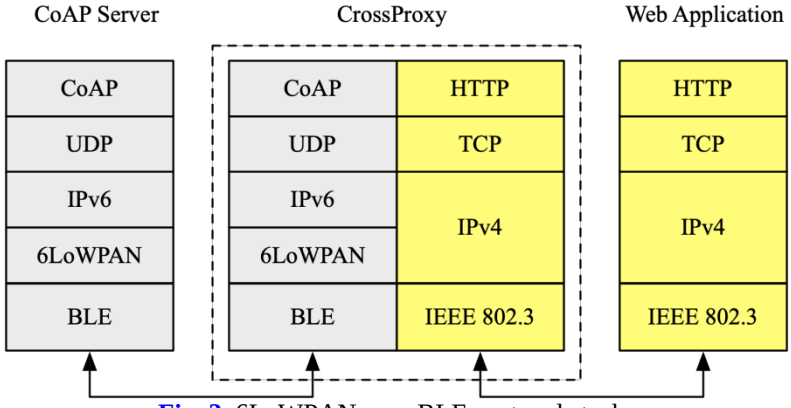


Fig. 3. 6LoWPAN over BLE protocol stacks

4.2 CrossProxy

The roles of CrossProxy are a gateway and edge router of sensor nodes. CrossProxy translates protocol between HTTP and CoAP and forwards packets from sensor nodes when an application sends a request message. Also, CrossProxy consists of five functional modules; these are CoAP Proxy Clients, priority queues, CoAP Proxy Server, Protocol translation, and Proxy HTTP Server as depicted in **Fig. 4**.

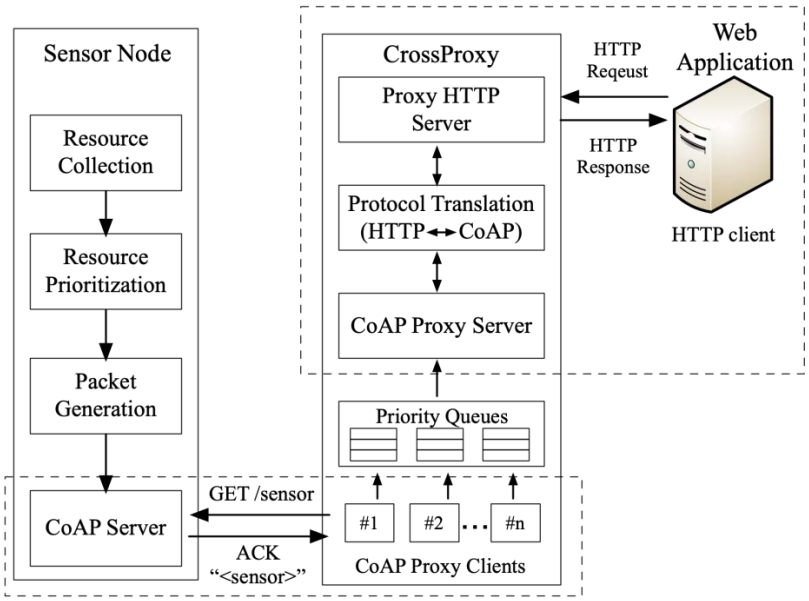


Fig. 4. System architecture

Proxy HTTP Server manages HTTP inquiries and replies messages to web applications. When a web application delivers an HTTP request to the Proxy HTTP server, the request packet is dispatched to the Protocol Translation module for translation. As presented in **Fig. 5**, an HTTP request comprises of HTTP URI, HTTP port number, CoAP URI, CoAP port number, and a resource name.

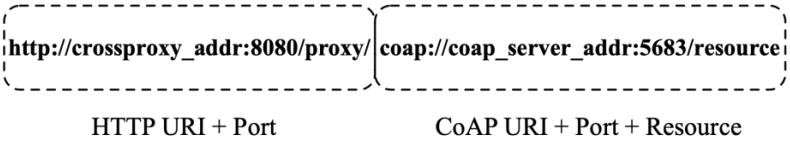


Fig. 5. HTTP and CoAP URIs

Protocol translation module probes a CoAP URI as well as the resource name from the HTTP request and distributes the analyzed CoAP URI to the CoAP Proxy Server module. Also, CoAP Proxy Server module verifies the priority queues to enqueue packets coming from sensor nodes. If the priority queues have packets, CoAP Proxy Server acquires packets from the top-priority queue to the lowest-priority queue. CoAP Proxy Server attempts to dequeue from lower-priority queues if the topmost-priority queue is unoccupied. When the CoAP Proxy Server has packets to transfer, the packets are sent to Proxy HTTP Server through the Protocol translation module. Finally, utilizing the HTTP protocol, Proxy HTTP Server responds to the web application with packets produced by sensor nodes.

A CoAP Proxy Client in CrossProxy makes observations of a registered resource in a sensor node. For observing registration, a CoAP Proxy Client sends a GET request message with an observe option. The CoAP Proxy Client is handled as an observer of the requested resource by a CoAP Server in a sensor node, which also receives the registration request if a sensor node has the requested resource from a CoAP Proxy Client.

Each time the resource modifies its state, a CoAP Server in a sensor node notifies the registered observer of the changed state. When a CoAP Proxy Client in CrossProxy accepts notifications involving sensor value, priority, and timestamp, the notices are enqueued into the priority queues corresponding to its sensor node-designated priority.

4.3 Sensor node (CoAP Server)

Resource collection, resource prioritization, packet generation, and CoAP server modules are the four efficient modules that constitute the sensor node. Sensor-generated periodic data is obtained by the resource collection module. A sensed data is categorized by the resource prioritization module using threshold values, a tolerance range, and a former sensor value to designate a priority when received. s shown in **Fig. 6**, the packet generation module, following the prioritization of the sensed data, creates a packet with its BLE device ID, sensor type, sensor value, priority, and a timestamp. Using CoAP observe notification, CoAP server module transmits response messages to a CoAP Proxy Client that is already registered as an observer in CoAP Server.

Device ID	Sensor Type	Sensor Value	Priority	Timestamp
6 bytes	1 byte	4 bytes	1 byte	8 bytes

Fig. 6. Packet format

5. Packet Prioritization and Transmission Scheme

5.1 Packet prioritization scheme

This study proposes a novel priority assignment scheme to decrease the number of packets to be sent to the resource-constrained network. Two threshold values and a tolerance value are adopted to categorize packets with priorities. First, this paper classifies a sensed value using two threshold values ($Threshold_{high}$, $Threshold_{low}$).

If a packet from a sensor belongs to the external range of the two threshold values, the highest priority (priority 0) is allocated to the packet as described in **Fig. 7**. Contrarily, a packet has a normal priority (priority 1) if the measured value is included in the inner range of the two threshold values.

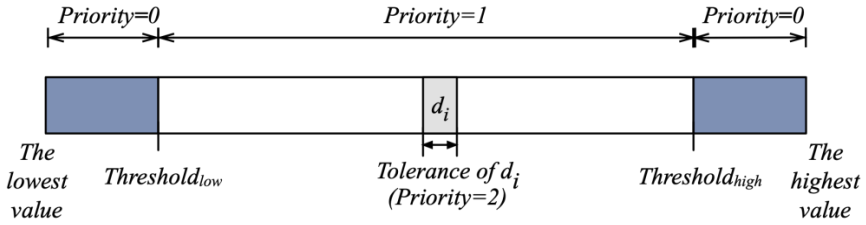


Fig. 7. Packet prioritization scheme

In addition, a tolerance range (T_i) for the prioritization of sensed data is utilized as sensors have an acceptable measurement data range. If the difference of the previously measured data (d_i) and the newly measured data (d_{i+1}) is within the tolerance range, the proposed scheme considers the newest data as the same with the previously measured value and assigns the lowest priority (priority 2) to the data. The lowest priority packets are transmitted by the proposed keep-alive message notification scheme to minimize packets delivered in IoT networks.

Furthermore, the proposed scheme adopts another parameter (*sudden_change*) to discern extraordinary incidents. If there is a significant difference in two successive data, the proposed mechanism considers that an unusual event has happened in the real situation. Accordingly, the packet having a drastic change has the highest priority (priority 0) even though the sensed value is placed in the inner range of two threshold values as shown in **Fig. 8**.

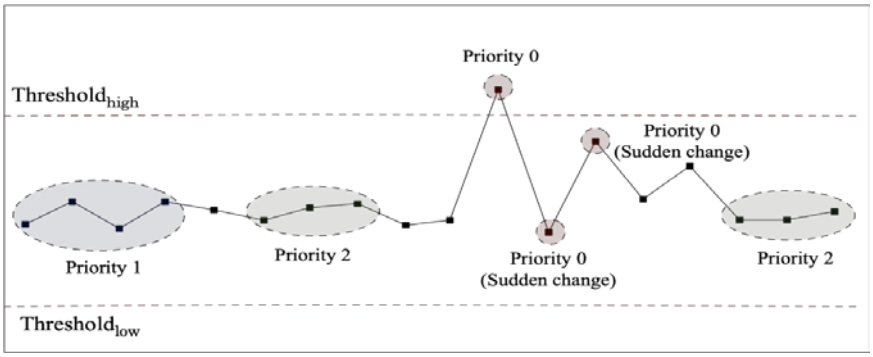


Fig. 8. Detection of sudden change data

Fig. 9 shows the detailed algorithm that assigns priorities to a sensed data and forwards the keep-alive message according to the priorities. After measuring value in a sensor node, the measured value is compared with two threshold values. If the value surpasses thresholds, priority 0 is assigned to the measured value.

Secondly, investigating an unusual incident is performed by calculating the absolute difference between two consecutive measurements. An irregular incident is considered happening when the difference is bigger than the sudden_change value and the proposed algorithm assigns priority 0 to the measurement.

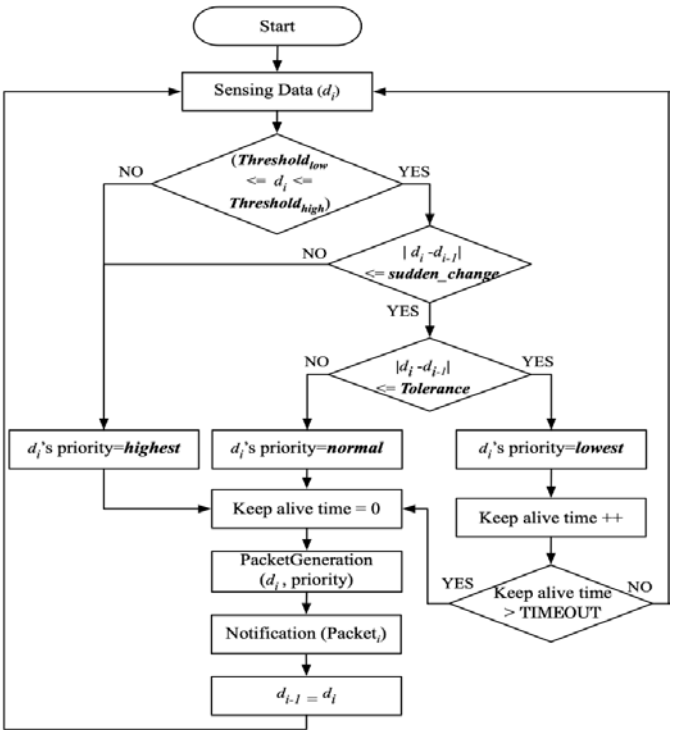


Fig. 9. Algorithm for assigning priorities to packets

Thirdly, the measured value is considered as normal and is assigned with priority 1 if the difference between two consecutive measurements is bigger than the tolerance value (T_i) and smaller than the sudden_change value. If a sensed value has priority 0 or 1, a packet is generated including other elements and delivered to the networks without delay.

5.2 Packet transmission scheme

The keep-alive message notification scheme is applied to transmit the lowest priority packets. If packets have priority 2 consecutively by the priority assignment algorithm, those packets are transmitted after the keep-alive time has elapsed as described in **Fig. 10**.

The keep-alive message notification is applied to decrease energy consumption and bandwidth usage in the resource-limited networks by minimizing the packet transmission. Moreover, the normal operation of a sensor node is informed to CrossProxy with the keep-alive message scheme. If a device stops its process by battery shortages and other failures, CrossProxy can recognize a node's failure.

Algorithm 1: Keep-alive message notification

```
1   KEEP_ALIVE_TIMEOUT = 3 seconds
2   while (1)
3     if ((sensed value == priority 0) or (sensed value == priority 1))
4       keep_alive_time = 0;
5       Packet_i = PacketGeneration (value, priority, device ID, timestamp);
6       Notification (Packet_i); /* send notification immediately */
7     else /* lowest priority */
8      if ((++keep_alive_time) >= KEEP_ALIVE_TIMEOUT)
9            keep_alive_count = 0;
10      Packet_i = PacketGeneration (value, lowest, device ID, timestamp);
11      Notification (Packet_i);
12        end if
13    end if
14  end while
```

Fig. 10. Algorithm for keep-alive message notification

Fig. 11 describes the proposed packet transmission mechanism. Packets with priority 0 or 1 are transmitted 100% as shown in case 1. However, some packets with priority 2 are skipped and not delivered to CrossProxy. If a series of packets have priority 2, the proposed packet transmission scheme only transmits every third packet as case 4. Therefore, the number of packets transmitted can be reduced using the proposed scheme.

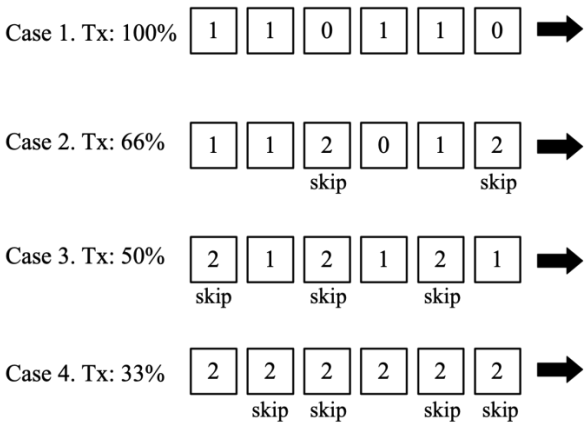


Fig. 11. Packet transmission scenarios of the proposed scheme

5.3 Packet classification in CrossProxy

CrossProxy's CoAP Proxy Clients receive packets from sensor nodes and analyze the priority from the transmitted packets. Then, each CoAP Proxy Client classifies a packet and enqueues it into the priority queue regarding the packet's priority as described in **Fig. 12**. However, if the associated priority queue is full of packets, packets are not queued. The packet drop number is counted in each priority queue by a CoAP Proxy Client.

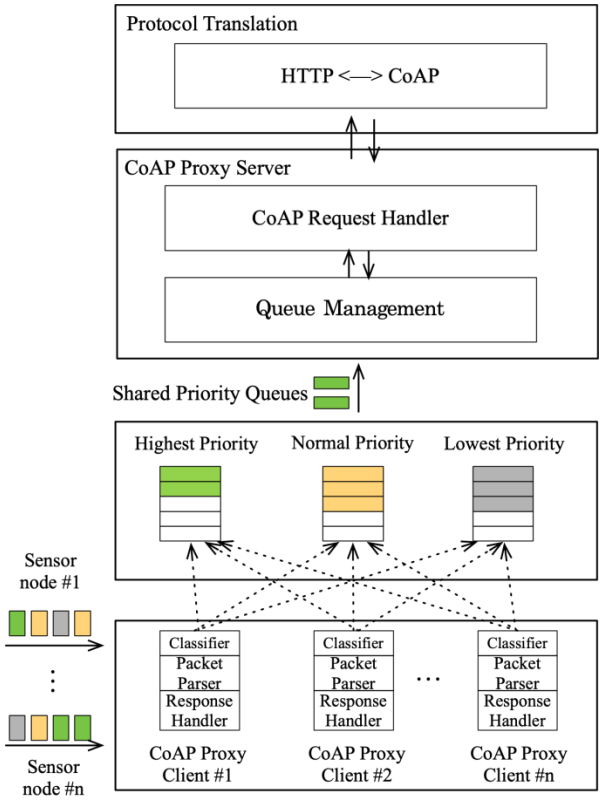


Fig. 12. Packet classification using priority queues in CrossProxy

CoAP Proxy Clients and a CoAP Proxy Server share the priority queues without establishing an extra connection to transmit packets to a web application. A CoAP Proxy Server examines the availability of priority queues and takes packets from the highest priority queue. CoAP Proxy Server transmits packets to the web application if priority queues contain packets.

6. Evaluation Environments

This section describes the evaluation environments of the PDT mechanism and the decision of evaluation parameters. A testbed consists of a CrossProxy using Raspberry Pi 3 Model B and six sensor nodes for CoAP servers with Raspberry Pi 3 Model B, Arduinos and dust sensors (GP2Y1010AU0F) as illustrated in **Fig. 13**.

A dust sensor measures a dust density at one-second interval and delivers data to a sensor node through a Universal Asynchronous Receiver Transmitter (UART). 6LoWPAN over BLE networks are established to communicate between sensor nodes and a CrossProxy.

Fig. 13. Testbed for performance evaluation

A CrossProxy exchanges IPv4 packets in communication request and response messages with a web application over the Internet. A web application sends an HTTP request message to CrossProxy every 300 milliseconds and analyzes its response to calculate transmission delay. For CoAP implementation, we used a Java-based open-source platform, that is, Californium (Cf) CoAP framework [35, 36]. CrossProxy can run on any computer systems in which Java Development Kit (JDK) is installed on it. Additionally, CrossProxy was implemented in Linux system.

Table 1 describes the characteristics of two existing approaches and the proposed scheme. The proposed approach classifies packets with threshold values ($Threshold_{high}$, $Threshold_{low}$), tolerance and a sudden_change value. In addition, the mechanisms of the packet reduction and QoS support per packet are provided with packet priority in the proposed scheme. CoAP observe and CoAP GET enqueue all packets into a single queue in CrossProxy without classifying packets. Therefore, packet prioritization and packet reduction schemes are not provided in the existing approaches.

Table 1. Evaluation approaches

No	Evaluation schemes	Packet classification with priorities	Packet reduction	Priority queues in CrossProxy	CoAP Scheme
1	Proposed scheme	Yes	Yes	Yes	Observe
2	CoAP Observe	None	None	None	Observe
3	CoAP GET	None	None	None	GET

6.1 Decision of evaluation parameters

The proposed scheme adopts four parameters for packet prioritization and packet reduction but it is difficult to determine the appropriate values of these parameters.

6.1.1 Tolerance value

Dust density data are collected 24 hours per second and an average absolute difference is computed between the previously measured value (d_{i-1}) and the current values (d_i) using equation (1). The average difference value represents the change of sensed value under a normal circumstance. Accordingly, the proposed scheme determines the difference value as the tolerance value. The tolerance value converges to 8.25 when the dust density is 80 during a clean day.

$$Tolerance = \frac{\sum_{i=1}^{N-1}|d_i - d_{i-1}|}{N-1} \quad (1)$$

Fig. 14 shows the mean dust density and mean tolerance values for a day. Although the values fluctuate at around 9 AM and 6 PM, the dust density stays constant throughout the day. Based on the experimental measurement, the tolerance value is selected as 8, which is used in this paper to evaluate performance.

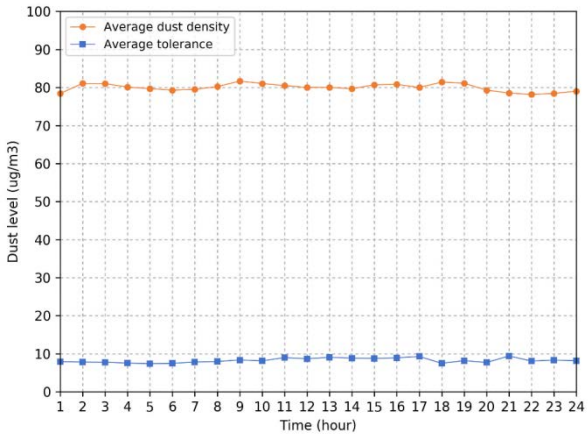


Fig. 14. Average dust density and tolerance values

6.1.2 Sudden change value

The sudden_change value is proposed to detect unexpected variations in this paper. The number of packets related to priorities is measured in a sensor node to determine the most suitable value for further performance evaluation. The sudden_change value is greater than the tolerance value and is associated with the packets having priority 0 and 1. As the sudden_change value changes from 8 to 24, the number of packets with zero priority drops

sharply as illustrated in **Fig. 15**. Conversely, when the value of sudden_change goes up from 8 to 24, the number of priority 1 packets augments significantly.

When the sudden_change value is small, a sensor node detects more packets with priority 0 in the slight variations of data and presumes that unusual events occur constantly.

However, infrequent abnormal events happen as the sudden_change value increases and a sensor node assigns priority 1 to more packets. As the sudden_change value changes from 32 to 48, the allocation of priority 0 and 1 to packets increases gradually. Therefore, 32 is chosen as the sudden_change value of the dust sensor in this paper.

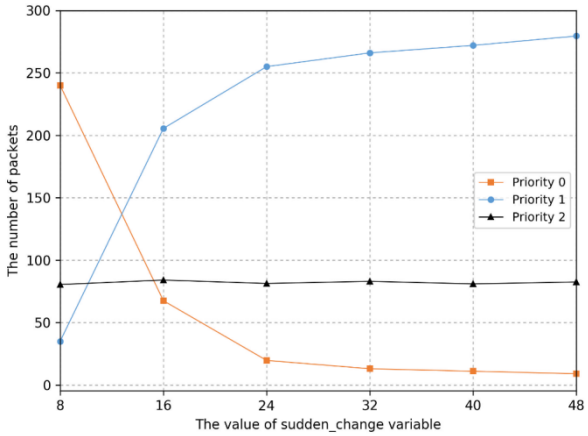


Fig. 15. The measurement for the sudden_change parameter

6.1.3 Threshold values

Two threshold values are adopted to assign priority to a packet in the proposed scheme. This paper referred to Air Quality Guide for Particle Pollution [37] and the Air Quality Index (AQI) from the European Environment Agency [38] to decide the threshold values. AQI is classified concerning the density of molecules less than 10μm (PM10). The index regards that the air quality is not healthy for sensitive people if PM10 density is higher than 100. The performance evaluation parameters are shown in **Table 2**.

Table 2. Evaluation parameters

Evaluation parameters	Values
$Threshold_{high}$	100
$Threshold_{low}$	0
Tolerance	8
Sudden change	32

7. Performance Evaluation

This paper evaluates packet transmission rate, energy consumption in a sensor node. The usage of bandwidth and packet drop rate are measured in the proxy. Moreover, the transmission delay from a sensor node to the web application is measured. When the packet drop rate is measured, three queues for each priority are adopted to the proposed mechanism.

Contrarily, only one queue with three times larger queue length is assigned to CoAP observe and CoAP GET methods for fairness of the performance evaluation because a priority is not assigned to a packet.

Moreover, when the transmission delay is measured on a clean day and a dusty day, the proposed scheme shows better performance in real weather conditions. This paper assumes that the resource URIs of sensor nodes are the same and the discovery procedure in CrossProxy has finished.

7.1 Packet transmission rate

The ratio of the total number of packet transmission (N_{TRANS}) over the total number of data sensed (N_{SENSED}) for a given period is defined as the packet transmission rate (T_{RATE}). The packet transmission rate is depicted in equation (2). A CoAP server collects data from a sensor every second and prioritizes a packet with threshold values, a tolerance value, and a sudden_change value in the proposed scheme. The prioritized packets are delivered to CrossProxy by the packet transmission scheme as described in Section 5.

$$T_{RATE} = \frac{N_{TRANS}}{N_{SENSED}} \tag{2}$$

The proposed scheme adopts the keep-alive message notification algorithm to minimize the transmission of the lowest-priority packets to decrease the packet transmission rate. A CoAP server of the proposed scheme sends 60% packets but the existing CoAP observe and GET approaches transmit 100% packets to CrossProxy as shown in **Fig. 16**.

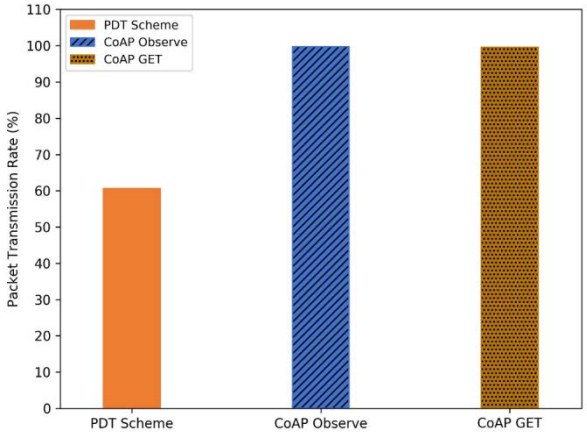
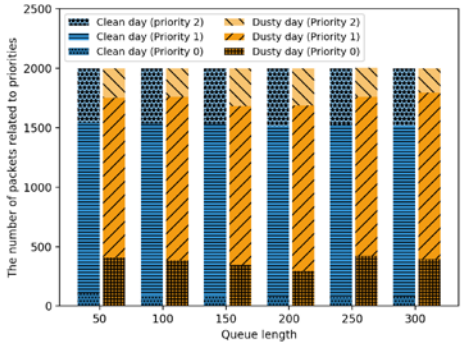


Fig. 16. Packet transmission rate of a CoAP server

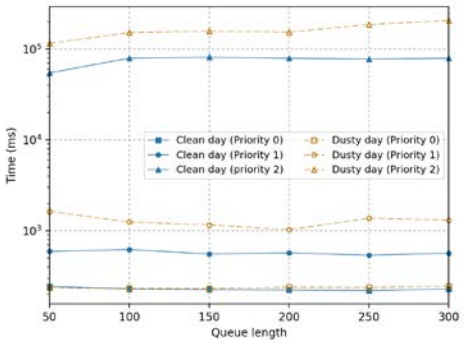
7.2 QoS support in the real situation

The proposed scheme verifies QoS support regarding the packet's priorities in the real weather conditions. The fine dust density levels of PM10 were measured using the dust sensor in the dust day and the clean day. The PM10's density level in the dust day was 120 ug/m^3 and the clean day's level was 58 ug/m^3. To demonstrate the enhanced performance in QoS support, the total number of arrival packets and the transmission latency in the web application are measured on a clean day and a very dusty day. For measuring transmission latency, the HTTP application sends a request message to CrossProxy every 300ms and CrossProxy delivers packets from CoAP servers to the HTTP application by altering its queue size from 50 to 300.

The total number of arrival packets and the transmission latency regarding priorities are compared in **Fig. 17(a)** and **Fig. 17(b)** respectively. The transmission latency of priority 0 in the dusty day shows a similar delay regardless of 4.3 times more packets with priority 0 than the clean day. However, the packets having priority 1 and 2 consume more time to arrive at the HTTP application in the dusty day because the highest priority packets are handled first in CrossProxy.



(a) The number of packets related to priorities



(b) Transmission delay from CoAP servers to a web application

Fig. 17. The peformance comparison in the real situation

As shown in **Fig. 17**, QoS support regarding transmission latency of the proposed scheme is demonstrated even in the severe weather conditions. Moreover, the proposed scheme differentiates transmission delay concerning a packet's priorities.

7.3 Energy consumption of a sensor node

Energy consumption for one hour of three schemes is measured in a sensor node. **Fig. 18** represents the average current consumption of the proposed scheme, CoAP observe, and CoAP GET, which are 0.59, 0.69 and 0.81 respectively.

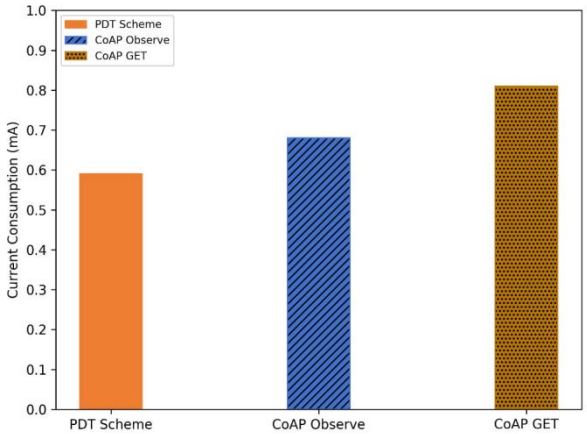


Fig. 18. Energy consumption of a sensor node

The proposed scheme using prioritized packet transmission with threshold values and a tolerance value has the least energy consumption than the existing CoAP methods. Although

the packet transmission rates of CoAP GET and CoAP observe scheme are almost the same, the energy consumption of CoAP GET approach is higher than CoAP observe scheme because CoAP GET exchanges longer messages with the URI of a CoAP server in a request message and an ACK message including the requested resource. When the proposed scheme and CoAP observe reply to a request, a shorter length of ACK message with vacant payload is delivered.

Therefore, the decreased number of transmission packet and the shorter message exchange result in the least energy consumption of the proposed scheme.

7.4 Bandwidth utilization in CrossProxy

Lower bandwidth consumption is a critical issue in the resource-constrained networks. Therefore, bandwidth usage in CrossProxy is measured to compare the performance of three approaches using Wireshark [39] while the number of CoAP servers increases to six. The proposed scheme shows the smallest bandwidth consumption as the number of CoAP increases.

The proposed scheme consumes less bandwidth at around 26% and 39% than CoAP observe and CoAP GET approaches with one CoAP server. When the number of CoAP server is six, the proposed scheme uses less bandwidth at around 43% of CoAP observe and 57% of CoAP GET as described in **Fig. 19**. CoAP GET approach exchanges longer messages than other methods so the bandwidth consumption of CoAP GET is the highest. As the number of CoAP servers grows, the difference in bandwidth consumption becomes bigger due to the packet reduction approach of the proposed scheme.

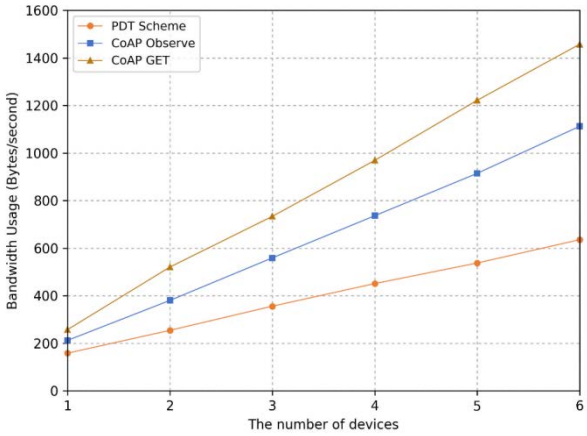


Fig. 19. Bandwidth utilization in CrossProxy

7.5 Packet loss rate in CrossProxy

The packet loss rate was calculated when CrossProxy receives a request message from a web application every 300ms. CrossProxy translates an HTTP request to a CoAP request message and forwards the request message to a sensor node. While CoAP Server in a sensor node sends response messages to CoAP Proxy Clients in CrossProxy, CrossProxy enqueues a packet into a priority queue concerning the priority. Packet loss occurs when a priority queue is filled with packets.

As can be seen from **Fig. 20**, the proposed scheme adopting the packet reduction mechanism exhibits the lowest packet loss rate and ends up at a zero rate when queue size becomes 150. In other approaches, as the queue length increases, the packet loss rates show

gradual decreases from around 40% to 32% because every packet from a sensor node is transmitted.

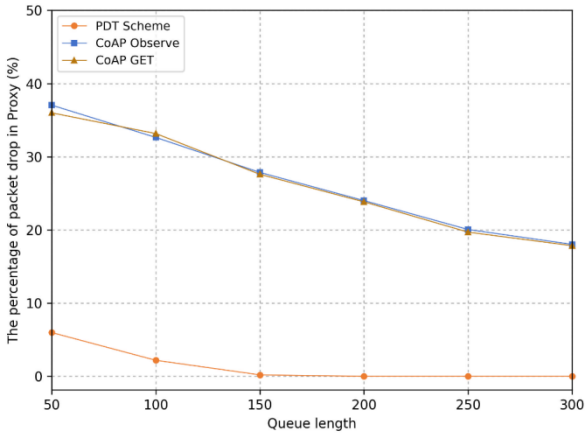


Fig. 20. Packet loss rate in CrossProxy

7.6 Transmission delay

Transmission latency from sensor nodes to a web application was measured by changing queue size in CrossProxy. Sensor nodes transmit packets to a web application when a web application requests every 300ms. The transmission delay of each priority is calculated in three approaches when a response packet from a sensor node arrives at a web application.

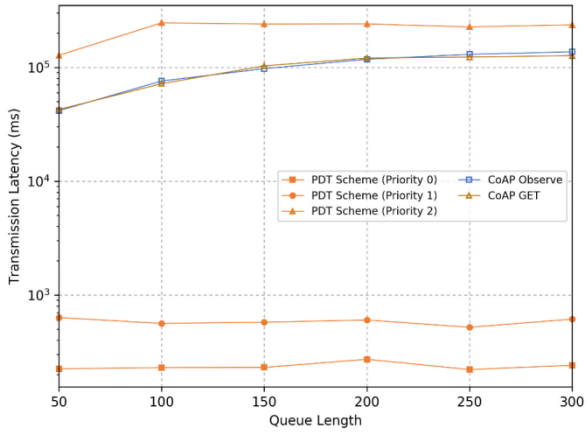


Fig. 21. Transmission delay in a web application

As illustrated in **Fig. 21**, the transmission delays of packets with priority 0 and 1 in the proposed scheme are much shorter than other approaches. However, packets with priority 2 in the proposed scheme have longer transmission latency than other approaches because the proposed scheme transmits packets with priority 2 after sensing three consecutive packets according to the packet transmission scheme as described in section 5.2.

8. Conclusion

This paper proposed a data prioritization and transmission mechanism to reduce the number of packets for better resource usage in the resource-limited networks. For the data prioritization scheme, three priority parameters are determined to allocate a priority to a packet. The prosed packet prioritization method reduces the packet transmission rate by 40% than the CoAP observe and CoAP GET. Moreover, this paper represents improved performance in bandwidth consumption, less energy usage, and packet drop rate due to the decreased packet transmission.

Furthermore, the proposed scheme supports QoS and differentiated transmission delay per packet using priority in CrossProxy. Comparing transmission latency in the real situation demonstrates the performance of QoS support. The performance evaluation in the real situation demonstrates that the highest priority packets have the same transmission delay even in a huge dust density environment.

Finally, the augmented resource utilization in the constrained network is the primary contribution of this paper. The proposed packet prioritization mechanism can be applied to the periodic monitoring applications under constrained resources.

References

[1] C. Gomez, J. Paradells, C. Bormann and J. Crowcroft, "From 6LoWPAN to 6Lo: Expanding the Universe of IPv6-Supported Technologies for the Internet of Things," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 148-155, 2017. Article (CrossRef Link)

[2] M. A. Feki, F. Kawsar, M. Boussard and L. Trappeniers, "The Internet of Things: The Next Technological Revolution," *Computer*, vol. 46, no. 2, pp. 24-25, 2013. Article (CrossRef Link)

[3] J. Kim, R. Haw, E. Cho, C. Hong and S. Lee, "A 6LoWPAN Sensor Node Mobility Scheme Based on Proxy Mobile IPv6," *IEEE Trans. Mob. Comput.*, vol. 11, no. 12, pp. 2060-2072, 2012. Article (CrossRef Link)

[4] T. Savolainen, K. Kerai, F. Berntsen, J. Decuir, R. Heydon, V. Zhodzishsky and E. Callaway, "Internet Protocol Support Profile," *Bluetooth Specification*, December, 2014. Article (CrossRef Link)

[5] G. Moritz, F. Golatowski, C. Lerche and D. Timmermann, "Beyond 6LoWPAN: Web Services in Wireless Sensor Networks," *IEEE Trans. Ind. Informatics*, vol. 9, no. 4, p. 1795–1805, 2013. Article (CrossRef Link)

[6] C. Bormann, A. Castellani and a. Z. Shelby, "CoAP: An Application Protocol for Billions of Tiny Internet Nodes," *IEEE Internet Comput.*, vol. 16, no. 2, pp. 62-67, Mar./Apr. 2012. Article (CrossRef Link)

[7] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang and W. Zhao, "A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1125-1142, 2017. Article (CrossRef Link)

[8] S. Raza, H. Shafagh, K. Hewage, R. Hummen and T. Voigt, "Lithe: Lightweight secure CoAP for the internet of things," *IEEE Sens. J.*, vol. 13, no. 10, p. 3711–3720, 2013. Article (CrossRef Link)

[9] A. Zanella, N. Bui, A. Castellani, L. Vangelista and M. Zorzi, "Internet of Things for Smart Cities," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 22-32, 2014. Article (CrossRef Link)

[10] A. Al-Fuqaha, A. Khreishah, M. Guizani, A. Rayes and M. Mohammadi, "Toward Better Horizontal Integration Among IoT Services," *IEEE Commun. Mag.*, vol. 53, no. 9, pp. 72-79, 2015. Article (CrossRef Link)

[11] J. Pradilla, R. Gonzalez, M. Esteve and C. Palau, "Sensor Observation Service (SOS)/Constrained Application Protocol (CoAP) proxy design," in *Proc. of 18th Mediterr. Electrotech. Conf. Intell. Effic. Technol. Serv. Citizen (MELECON), Limassol, Cyprus*, pp. 1-5, 2016. Article (CrossRef Link)

[12] M. Ruta, F. Scioscia, A. Pinto, F. Gramegna, S. Ieva, G. Loseto and E. D. Sciascio, "A CoAP-based framework for collaborative sensing in the Semantic Web of Things," *Procedia Computer Science*, vol. 109, pp. 1047-1052, 2017. Article (CrossRef Link)
[13] K. Hartke, "Observing Resources in the Constrained Application Protocol (CoAP)," *IETF RFC 7641*, 2015. Article (CrossRef Link)
[14] I. Ishaq, J. Hoebeke, I. Moerman and P. Demeester, "Observing CoAP groups efficiently," *Ad Hoc Networks*, vol. 37, pp. 368-388, 2016. Article (CrossRef Link)
[15] S. Zhao, L. Yu and B. Cheng, "An Event-Driven Service Provisioning Mechanism for IoT (Internet of Things) System Interaction," *IEEE Access*, vol. 4, p. 5038–5051, 2016. Article (CrossRef Link)
[16] X. Masip-Bruin, E. Marn-Tordera, G. Tashakor, A. Jukan and G. J. Ren, "Foggy clouds and cloudy fogs: A real need for coordinated management of fog-to-cloud computing systems," *IEEE Wireless Commun.*, vol. 23, no. 5, pp. 120-128, 2016. Article (CrossRef Link)
[17] Y. Nikoloudakis, S. Panagiotakis and E. Markakis, "A Fog-Based Emergency System for Smart Enhanced Living Environments," *IEEE Cloud Comput.*, vol. 3, no. 6, pp. 54-62, 2016. Article (CrossRef Link)
[18] W. Shi, J. Cao, Q. Zhang, Y. Li and L. Xu, "Edge Computing: Vision and Challenges," *IEEE Internet Things Journal*, vol. 3, no. 5, pp. 637-646, 2016. Article (CrossRef Link)
[19] A. V. Dastjerdi and R. Buyya, "Fog Computing: Helping the Internet of Things Realize Its Potential," *Computer*, vol. 49, no. 8, pp. 112-116, 2016. Article (CrossRef Link)
[20] A. Betzler, C. Gomez, I. Demirkol and J. Paradells, "CoAP Congestion Control for the Internet of Things," *IEEE Commun. Mag.*, vol. 54, no. 7, pp. 154-160, 2016. Article (CrossRef Link)
[21] A. Talaminos-Barroso, M. A. Estudillo-Valderrama, L. M. Roa, J. Reina-Tosina and F. Ortega-Ruiz, "A Machine-to-Machine protocol benchmark for eHealth applications - Use case: Respiratory rehabilitation," *Comput. Methods Programs Biomed.*, vol. 129, pp. 1-11, 2016. Article (CrossRef Link)
[22] M. Castro, A. J. Jara and A. F. Skarmeta, "Enabling end-to-end CoAP-based communications for the Web of Things," *Journal of Network and Computer Applications*, vol. 59, pp. 230-236, 2016. Article (CrossRef Link)
[23] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2347-2376, 2015. Article (CrossRef Link)
[24] W. Shi and S. Dustdar, "The Promise of Edge Computing," *Computer*, vol. 49, no. 5, pp. 78-81, 2016. Article (CrossRef Link)
[25] Z. Shelby, K. Hartke and C. Bormann, "The Constrained Application Protocol (CoAP)," *IETF RFC 7252*, 2014. Article (CrossRef Link)
[26] W. Gao, J. Nguyen, W. Yu, C. Lu and D. Ku, "Assessing Performance of Constrained Application Protocol (CoAP) in MANET Using Emulation," in *Proc. of ACM Int. Conf. Rel. Convergent Syst. (RACS), Odense, Denmark*, pp. 103-108, 2016. Article (CrossRef Link)
[27] A. Betzler, C. Gomez, I. Demirkol and M. Kovatsch, "Congestion Control for CoAP Cloud Services," in *Proc. of 19th IEEE Int. Conf. Emerg. Technol. Fact. Autom. ETFA*, pp. 1-6, 2014. Article (CrossRef Link)
[28] A. Ludovici, A. Calveras and A. Calveras, "A Proxy Design to Leverage the Interconnection of CoAP Wireless Sensor Networks with Web Applications," *Sensors*, vol. 15, no. 1, pp. 1217-1244, 2015. Article (CrossRef Link)
[29] A. Castellani, T. Fossati and S. Loreto, "HTTP-CoAP Cross Protocol Proxy: An Implementation Viewpoint," in *Proc. of 2012 IEEE 9th Int'l. Conf. Mobile Adhoc and Sensor Systems (MASS)*, pp. 1-6, 2012. Article (CrossRef Link)
[30] G. K. Teklemariam, J. Hoebeke, I. Moerman and P. Demeester, "Facilitating the creation of IoT applications through conditional observations in CoAP," *EURASIP Journal on Wireless Communications and Networking*, vol. 2013, no. 1, pp. 1-19, 2013. Article (CrossRef Link)

[31] N. Correia, D. Sacramento and G. Schutz, "Dynamic Aggregation and Scheduling in CoAP/Observe-Based Wireless Sensor Networks," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 923-936, 2016. Article (CrossRef Link)
[32] A. Ludovici, E. Garcia, X. Gimeno and A. C. Augé, "Adding QoS support for timeliness to the observe extension of CoAP," in *Proc. of Int. Conf. Wirel. Mob. Comput. Netw. Commun*., pp. 195-202, 2012. Article (CrossRef Link)
[33] C. Pereira, A. Pinto, D. Ferreira, and A. Aguiar, "Experimental Characterization of Mobile IoT Application Latency," *IEEE Internet Things J*., vol. 4, no. 4, pp. 1082–1094, 2017. Article (CrossRef Link)
[34] A. Larmo, A. Ratilainen, and J. Saarinen, "Impact of CoAP and MQTT on NB-IoT System Performance," *Sensors*, vol. 19, no. 1, p. 7, 2019. Article (CrossRef Link)
[35] "The Eclipse Foundation Californium (Cf)," [Online]. Article (CrossRef Link)
[36] M. Kovatsch, M. Lanter and Z. Shelby, "Californium: Scalable Cloud Services for the Internet of Things with CoAP," in *Proc. of 4th Int. Conf. Internet of Things (IoT'14)*, pp. 1-6, October 2014. Article (CrossRef Link)
[37] "Air Quality Guide for Particle Pollution – Environmental Protection Agency (EPA)," May 2018. [Online]. Article (CrossRef Link)
[38] "European Air Quality Index," [Online]. Article (CrossRef Link)
[39] "Wireshark 2.6.0," [Online]. Article (CrossRef Link)

Changsu Jung received Ph.D. in the School of Computer Science and Engineering from Kyungpook University in 2018. Currently, he is working as a visiting professor at the School of Computer Science and Engineering in Kyungpook National University. His research interests include architecture of IoT, machine-to-machine communication, and edge computing.