Sensors and Materials, Vol. 30, No. 8 (2018) 1817–1824
MYU Tokyo

S & M 1638

Load-balanced Resource Directory Architecture for Large-scale Internet of Things Local Networks

Jung-Hyok Kwon,[1] Jaehoon Park,[2] and Eui-Jik Kim[1*]

[1]Department of Convergence Software, Hallym University,
1 Hallymdaehak-gil, Chuncheon-si, Gangwon-do 24252, South Korea
[2]Department of Electronic Engineering, Hallym University,
1 Hallymdaehak-gil, Chuncheon-si, Gangwon-do 24252, South Korea

(Received March 30, 2017; accepted April 12, 2018)

Keywords: CoAP, Internet of things, IoT local network, resource directory, scalability

This paper presents a load-balanced resource directory (LB-RD) architecture based on the constrained application protocol (CoAP), and is aimed at addressing the scalability issue with the centralized resource discovery approach in large-scale Internet of Things (IoT) local networks. To balance the traffic load due to a large number of messages concentrated on the resource directory (RD), the LB-RD serves as an intermediary agent between the servers and the RD. Its architecture includes the following three essential functional blocks: server group management (SGM), message aggregation (MA), and message forwarding scheduling (MFS). The SGM manages the uniform resource identifier (URI) list for a group of servers associated with the LB-RD. The MA aggregates the registration/update messages received from the servers into a single aggregated message. Finally, the MFS determines the forwarding time for the LB-RD to send its aggregated message to the RD. To verify the effectiveness of the LB-RD architecture, an experimental simulation is conducted under various scenarios. The results show that the LB-RD architecture achieves better performance than the existing RD approach in terms of the number of lost messages and the message loss rate.

1. Introduction

Recently, interest in the Internet of Things (IoT) has increased rapidly and the technology is becoming more common. IoT is generally considered a key technology for large-scale monitoring applications such as smart cities, smart factories, and smart farms.(1) In these large-scale applications, the IoT local network is composed of a number of devices with constrained physical resources, such as limited batteries, memory, and CPU, and is similar to a wireless sensor network (WSN).(2) The constrained representational state transfer (RESTful) environment (CoRE) working group from the Internet engineering task force (IETF) has standardized the constrained application protocol (CoAP) as a lightweight web transfer protocol for resource-constrained devices in IoT local networks that maintain a small header size and operate on the user datagram protocol (UDP).(3,4)

[*]Corresponding author: e-mail: ejkim32@hallym.ac.kr
https://dx.doi.org/10.18494/SAM.2018.1897

ISSN 0914-4935 © MYU K.K.
http://myukk.org/

In the CoAP, resource discovery is an indispensable procedure through which the client obtains the necessary resources from the servers. The servers are various types of sensor devices having the resource, which is represented as its name, path, interface description, and resource type. The client should first discover a resource before requesting it from a particular server. There are two types of CoAP resource discovery: distributed and centralized.(5) The distributed resource discovery is where the client directly queries a specific server.(6) This method is not suitable for large-scale IoT local networks because the client has to know the Internet protocol (IP) address and host name of a multitude of servers for communication.(7) On the other hand, the centralized resource discovery uses the resource directory (RD), which maintains a set of web links known as RD entries on the servers, for energy efficiency. This centralized method using RD also suffers from low scalability when employed in large-scale IoT local networks. In a large-scale IoT local network, a large number of servers periodically send an update message to the RD to keep their resource entries updated. However, the RD cannot afford to receive all the update messages because of the limited bandwidth of the wireless link. As the traffic load increases heavily, the RD suffers from problems such as message loss, network congestion, and bottlenecks. Therefore, a load-balancing mechanism for the RD is needed to overcome the scalability issue in IoT local network environments.

To solve the above-mentioned problems, a number of studies on resource discovery and RD have been performed. Djamaa and Yachir proposed a proactive RD discovery (PRD) mechanism based on adaptive trickle timers.(8) This mechanism uses a trickle timer for scalable, reliable, and effective RD discovery and performs RD discovery by propagating a CoAP message called a directory advertisement (DA) to all devices in a wavelike pattern. However, it focuses only on the discovery of the RD and does not consider the traffic overhead and concentration on the RD caused by resource updates. Kim *et al.* proposed a hybrid mechanism to overcome the overburden on the RD.(5) This mechanism basically uses a centralized resource discovery method using the RD. When overload occurs in the RD, the client directly broadcasts a message by a distributed method. However, this mechanism cannot prevent the traffic load problem in the RD in advance.

In this paper, we propose a load-balanced resource directory (LB-RD) architecture based on the CoAP, which aims to address the scalability issue of a centralized resource discovery approach in large-scale IoT local networks. To balance the traffic load due to a large number of messages concentrated on the RD, the LB-RD plays the role of an intermediary agent between the servers and the RD, and its architecture includes the following three essential functional blocks: server group management (SGM), message aggregation (MA), and message forwarding scheduling (MFS). First, the SGM manages the uniform resource identifier (URI) list of a group of servers associated with the LB-RD. Second, the MA aggregates the registration/update messages received from the servers into a single aggregated message, which consists of payloads of multiple individual messages and a single header. Finally, the MFS determines the forwarding time for the LB-RD to send its aggregated message to the RD. Through this LB-RD architecture, the RD can efficiently reduce the traffic load concentrated on itself and manage a large amount of sensor data under large-scale IoT local network environment including a multitude of sensors. In order to verify the effectiveness of the LB-RD architecture, we conduct an experimental simulation to compare the number of lost messages and the message loss rate of the LB-RD architecture with that of the existing RD approach. The number of lost messages and the message loss rate are evaluated for the purpose of increasing the number of servers and bandwidth. The results show that the number of lost messages and the message loss rate for the LB-RD architecture are 12 and 11% lower, on average, than the values for the existing RD approach.

The rest of this paper is organized as follows. Section 2 describes the design and operation of the LB-RD architecture. The simulation setting and results for the LB-RD architecture are presented in Sect. 3. Finally, we conclude the paper in Sect. 4.

2. LB-RD

The LB-RD aims to aggregate update messages from the servers to reduce the traffic load concentrated on the RD. Similar to the RD, the LB-RD performs the following three operations sequentially: discovery, registration, and update. In the following subsections, we describe the design and operation of the LB-RD.

2.1 Design of LB-RD architecture

As mentioned earlier, the LB-RD serves as an intermediary agent between a group of servers associated with it and the RD. To mitigate the traffic load problem in the RD due to the concentration of update messages from the servers, the LB-RD conducts MA for the servers, then sends the aggregated message toward the RD. Figure 1 shows the architecture of the LB-RD, which is composed of three essential functional blocks: SGM, MA, and MFS.

- SGM: First, the SGM manages the URI list for a group of servers associated with the LB-RD. The URI list is constructed when the servers register their resource entries via

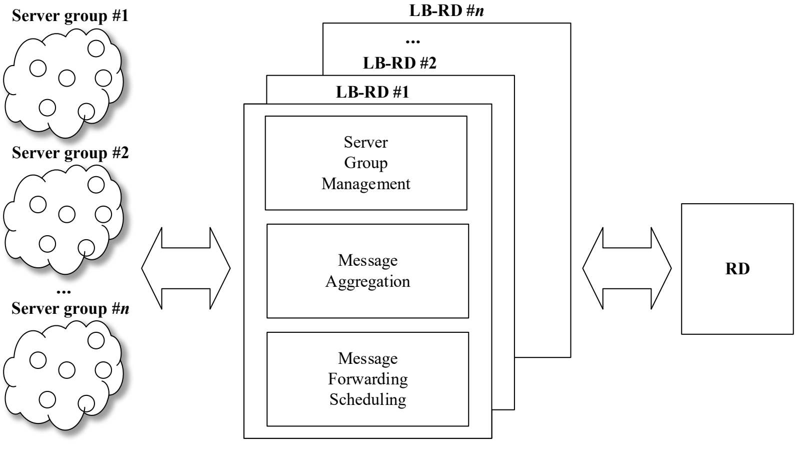

Fig. 1. LB-RD architecture.

registration messages and is used when the LB-RD aggregates the messages received from the servers via the MA functional block and multicasts the response message to the servers.

- MA: Second, the MA aggregates the registration/update messages received from the servers into a single aggregated message, which consists of the payloads of multiple individual messages and a single header.
- MFS: Finally, the MFS determines the forwarding time for the LB-RD to send its aggregated message to the RD. For this, we consider the maximum payload size of the CoAP message format. The size of the payload included in the aggregated message should be smaller than or equal to the maximum payload size of the CoAP message. This determines the number of registration/update messages that can be aggregated at one time. When the MA process for the MA block is completed, the MFS immediately forwards the aggregated message to the RD.

2.2 CoAP resource discovery with LB-RD

Because of the presence of the LB-RD, the CoAP resource discovery operations between the servers and RD should inevitably be modified. In this subsection, we describe the CoAP resource discovery operation including the LB-RD in detail. For convenience, we define the operation divided into the following three procedures: discovery, registration, and update.

Figure 2 shows the discovery procedure for the LB-RD. First, the RD obtains information about the LB-RDs (i.e., address and port) in response to the GET request message transmitted to the LB-RDs, and then, it assigns a different group of servers to each LB-RD, for which the RD maintains the LB-RDs' information. After that, each server sends a GET request message to the RD to request information about the LB-RD designated to it. The RD responds to this GET request message with the LB-RD information.

Figure 3 shows the registration procedure for the LB-RD. Each server sends a registration message, using the POST method, to the designated LB-RD, which contains the resource entry

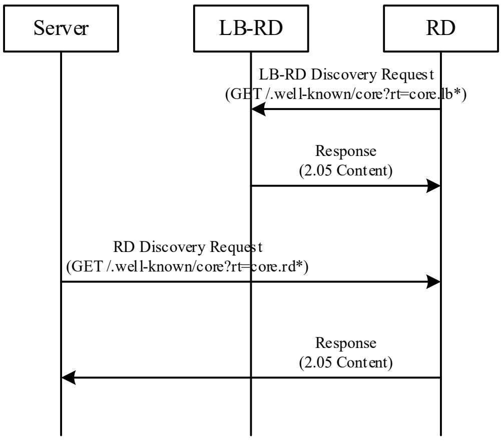

Fig. 2. Discovery procedure for LB-RD.

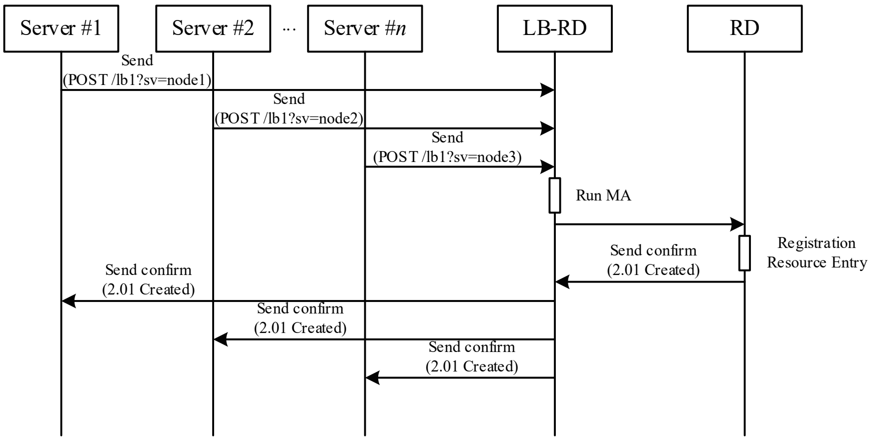
Server #1
Server #2
···
Server #*n*
LB-RD
RD
Send (POST /lb1?sv=node1)
Send (POST /lb1?sv=node2)
Send (POST /lb1?sv=node3)
Run MA
Registration Resource Entry
Send confirm (2.01 Created)
Send confirm (2.01 Created)
Send confirm (2.01 Created)
Send confirm (2.01 Created)

Fig. 3. Registration procedure for LB-RD.

for its resource. Note that in the LB-RD architecture, the servers register their resource entries in their RD via the LB-RD, whereas in the existing CoAP resource discovery, the servers register directly in their RD. The LB-RD counts the number of registration messages received from the servers, for the execution of the MA functional block, by which it sequentially aggregates the *k* registration messages. The value of '*k*' is determined by the payload size of the registration message (L_{Reg}) and the maximum payload size of the CoAP message (L_{CoAP}) using Eq. (1).

$$k = \left[\frac{L_{CoAP}}{L_{Reg}} \right] \tag{1}$$

Each LB-RD delivers the aggregated message immediately to the RD to prevent additional delays. Upon receiving the aggregated registration message, the RD registers the resource entries of *k* servers and responses with the LB-RD. Likewise, the LB-RD that receives a successful response sends the response message to its associated servers.

After the registration procedure, the servers periodically update the resource entries of their RD. Figure 4 shows the update procedure for the LB-RD; it ensures that the RD maintains the most recent resource entry for each server. This update procedure is the same as the registration procedure described above, and the update message is used instead of the registration message. Note that the proposed LB-RD architecture reduces the number of update messages that the RD receives via the use of an MA functional block, and its load-balancing effect is much more effective in large-scale IoT local networks including a large number of servers.

3. Performance Evaluation

To verify the effectiveness of the LB-RD architecture, we conduct an experimental simulation using MATLAB, and compare the simulation results of the LB-RD architecture

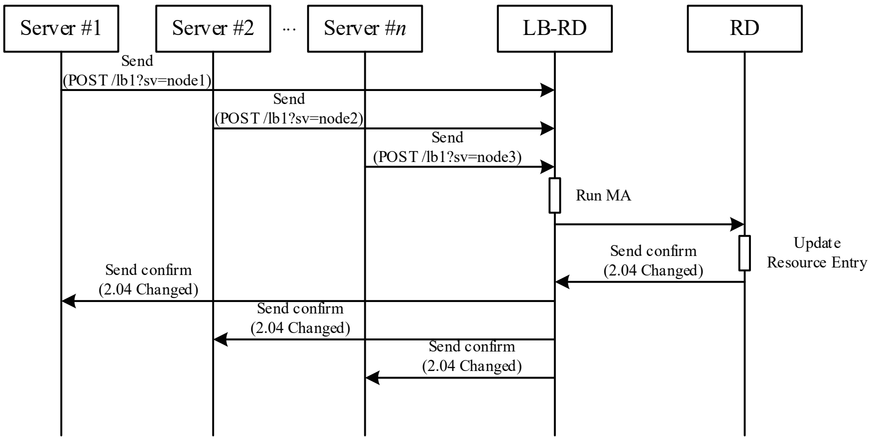

Fig. 4. Update procedure for LB-RD.

with that of the existing RD approach. More specifically, we check the performance of the LB-RD architecture in terms of the number of lost messages and the message loss rate through the simulation. To this end, the number of servers and LB-RDs are set to 100–1000 and 10, respectively. In the simulation, the number of update messages for MA is set to 5, so five update messages are combined into a single aggregated message by the LB-RD. The header size is 10 B and the payload size is 200 B. The bandwidth between servers and RD varies from 1 to 1.45 Mbps. Table 1 lists the simulation parameters.

Figure 5 shows the variation in the number of lost messages when the number of servers changes. In the simulation, the bandwidth between servers and RD is fixed to 1 Mbps. In the figure, the LB-RD architecture has fewer lost messages than the existing RD approach when the number of servers is over 500. In particular, when the number of servers is 600, only the existing RD approach loses update messages. This is because the LB-RD reduces the overhead caused by the header of update messages through aggregation. In other words, the LB-RD transmits more update messages through the 1 Mbps bandwidth than the existing RD approach. In both cases, the update messages are not lost until the number of servers is less than 500, since the bandwidth is sufficient for accommodating all update messages from the servers. Figure 6 shows the message loss rate for varying numbers of servers. The bandwidth is set to 1 Mbps, as in the case of Fig. 5. In the figure, the message loss rate for the LB-RD is lower than that of the existing RD approach, since the LB-RD combines multiple update messages into a single aggregated message. More specifically, the average message loss rates for the existing RD approach and the LB-RD architecture are 12 and 10%, respectively.

Figures 7 and 8 show the number of lost messages and the message loss rate when the bandwidth changes from 1 to 1.45 Mbps, respectively. In both simulations, the number of servers is set to 1000. In Fig. 7, the number of lost messages decreases as the bandwidth increases. This is because more update messages can be transmitted as the bandwidth increases in both cases. Overall, the LB-RD architecture has fewer lost messages than the existing RD approach because of the MA. Specifically, the LB-RD architecture has 11% fewer lost messages

Table 1
Simulation parameters.

Parameter	Value
Number of servers	100–1000
Number of LB-RDs	10
Number of update messages for aggregation	5
Header size	10 B
Payload size	200 B
Bandwidth	1–1.45 Mbps

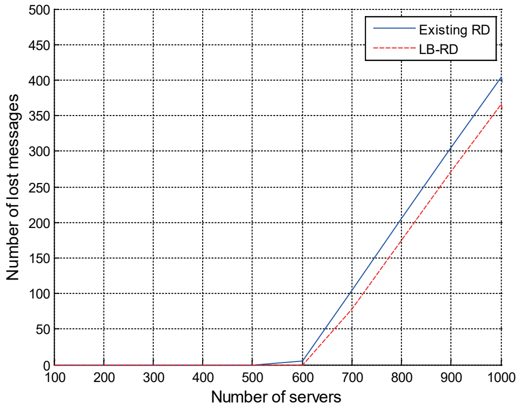

Fig. 5. (Color online) Number of lost messages per server.

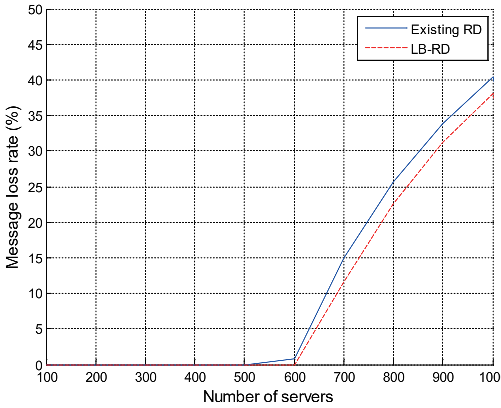

Fig. 6. (Color online) Message loss rate per server.

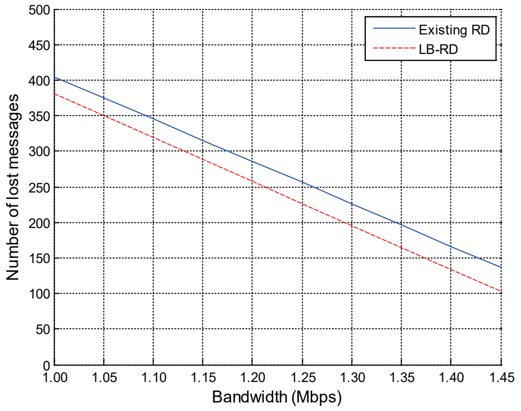

Fig. 7. (Color online) Number of lost messages per bandwidth.

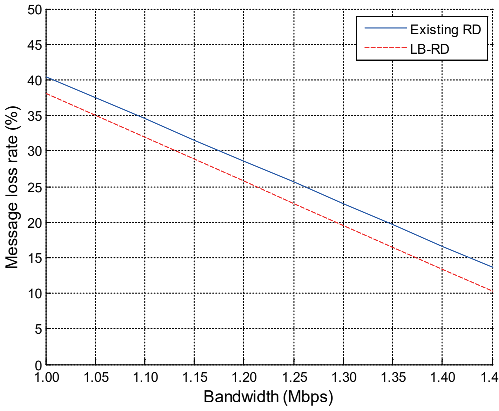

Fig. 8. (Color online) Message loss rate per bandwidth.

than the existing RD approach, on average. The message loss rate at various bandwidths is shown in Fig. 8. In the figure, the message loss rate of the LB-RD architecture is lower than that of the existing RD approach regardless of the bandwidth. This is for the same reason as in the case of Fig. 7. In this case, the message loss rate of the LB-RD architecture is 24%, which is 11% less than that of the existing RD approach.

4. Conclusions

In this paper, we presented an LB-RD architecture based on the CoAP, which addresses the scalability issue of a centralized resource discovery approach in large-scale IoT local networks. To balance the traffic load due to a large number of messages concentrated on the RD, the LB-RD plays the role of an intermediary agent between the servers and the RD, where the architecture includes three essential functional blocks: SGM, MA, and MFS. The SGM manages the URI list for a group of servers associated with the LB-RD. The MA aggregates the registration/update messages received from the servers into a single aggregated message. The MFS determines the forwarding time for the LB-RD to send its aggregated message to the RD. To verify the effectiveness of the LB-RD architecture, we conducted an experimental simulation under various scenarios using MATLAB. In the simulation, variations in the number of lost messages and the message loss rate were observed for various numbers of servers and bandwidths. The results showed that the LB-RD architecture has 12% fewer lost messages and an 11% lower message loss rate than the existing RD approach.

Acknowledgments

This work was supported by the Hallym Leading Research Group Support Program of 2017 (HRF-LGR-2017-0003).

References

1 I. Yaqoob, E. Ahmed, I. A. T. Hashem, A. I. A. Ahmed, A. Gani, M. Imran, and M. Guizani: IEEE Wireless Commun. **24** (2017) 10.

2 S. Cirani, L. Davoli, G. Ferrari, R. Léone, P. Medagliani, M. Picone, and L. Veltri: IEEE IoT J. **1** (2014) 508.

3 Z. Shelby, K. Hartke, and C. Bormann: The Constrained Application Protocol (CoAP) RFC 7252 (2014).

4 M. Majanen, P. Koskela, and M. Valta: Proc. 5th Int. Conf. Smart Grids, Green Communications and IT Energy-aware Technologies (ENERGY, 2015) 47.

5 Y. Kim, K.-H. Kim, T. Shon, and J.-H. Kim: Proc. 2015 Int. Conf. Information Networking (ICOIN) (IEEE, 2015) 407.

6 B. C. Villaverde, R. D. P. Alberola, A. J. Jara, S. Fedor, S. K. Das, and D. Pesch: IEEE Commun. Surv. Tutorials **16** (2014) 41.

7 Z. Shelby, M. Koster, and C. Bormann: CoRE Resource Directory draft-ietf-core-resource-directory-10 (2017).

8 B. Djamaa and A. Yachir: Procedia Comput. Sci. **83** (2016) 115.