Sensors and Materials, Vol. 30, No. 3 (2018) 431–437
MYU Tokyo

S & M 1505

Energy-efficient Resource Directory Update Scheme for Constrained Application Protocol in Internet of Things

Sungmo Kim,[1] Jung-Hyok Kwon,[1] Sol-Bee Lee,[1]
Jaehoon Park,[2] Dongwan Kim,[3*] 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
[3]Department of Electronic Engineering, Dong-A University,
37 Nakdong-daero 550 beon-gil, Saha-gu, Busan 49315, South Korea

(Received August 1, 2017; accepted November 2, 2017)

Keywords: CoAP, dead node, resource directory, update period adaption, update message

In this paper, we present an energy-efficient resource directory update (ERDU) scheme for the constrained application protocol (CoAP) in Internet of Things (IoT), which is designed to prevent unnecessary energy consumption caused by dead nodes and the frequent transmission of update messages. The ERDU scheme improves the energy efficiency of centralized resource discovery procedures in the CoAP via two main mechanisms: (1) dead node detection and (2) update period adaptation. The former detects dead servers and removes their resource from resource directory (RD) entries to prevent the unnecessary retransmission of the client. The latter adjusts the update period of the server on the basis of its current energy level. Simulation results show that the energy efficiency of the ERDU scheme is better than that of the legacy RD update mechanism.

1. Introduction

Recently, the use of lightweight web-based transfer protocols has been strongly recommended for exchanging data in Internet of Things (IoT) environments since most IoT services use constrained devices and lossy networks. Specifically, the IoT devices with limited energy, memory, and computational capabilities communicate with each other via wireless sensor networks (WSNs) and thus, a lightweight application protocol is essential for this constrained condition of IoT services. The constrained RESTful environments (CoRE) working group of the Internet Engineering Task Force (IETF) has developed the constrained application protocol (CoAP) to allow reliable and efficient communications in resource-constrained environments.[1,2] The CoAP uses a client/server model. The server makes up a data structure (i.e., resource) for sensing data, which is identified by a uniform resource identifier (URI), and clients access the resource using CoAP methods (i.e., GET, PUT, POST, and DELETE). To

*Corresponding author: e-mail: dongwankim@dau.ac.kr
**Corresponding author: e-mail: ejkim32@hallym.ac.kr
http://dx.doi.org/10.18494/SAM.2018.1763

retrieve the sensor data from the server, the client needs to know the resources of the server; thus, resource discovery is indispensable.

The CoAP includes two types of resource discovery: distributed and centralized. In the former, the client directly sends a query to a number of servers to search their resources; thus, this method is not suitable for IoT environments where energy efficiency is highly critical.[3,4] In contrast, the latter uses the resource directory (RD) to minimize energy consumption accrued from resource discovery procedures, which is defined in the Internet-Draft of the CoRE working group.[5] Therefore, the centralized method is appropriate for various IoT applications. In centralized resource discovery, the RD maintains RD entries that refer to sets of web links on resources hosted by various servers. The servers register their resources directly in their RD, and the client can recognize the resources of a specific server by requesting for the RD. However, this centralized resource discovery approach using the RD has two problems. First, each server should periodically send an update message to keep the RD information up-to-date; however, this operation may cause additional signaling overhead and decrease the network lifetime. Second, if the server is abruptly terminated due to failure or power shutdown without reporting its status, the RD is unable to maintain an up-to-date resource information.[6–8]

In this paper, we propose an energy-efficient resource directory update (ERDU) scheme for the CoAP. The ERDU scheme addresses the above-mentioned problems of the existing RD update mechanism in the centralized resource discovery approach of the CoAP via two mechanisms: (1) dead node detection and (2) update period adaptation. The client initiates the detection of dead nodes when the RD cannot be updated owing to the failure or power shutdown of the server. If the server does not respond to the client's request, the client requests the server's resource information from the RD. If the RD responds with the same information as the previous resource entries for a predetermined number of times, the client determines that the server is dead. This mechanism prevents unnecessary message transmission. In update period adaptation, the servers adjust the transmission period of update messages depending on their energy level. This improves the energy efficiency of the servers by reducing the transmission of update messages. To evaluate the performance of the ERDU scheme, we have conducted an experimental simulation and compared the results with that of the existing RD update mechanism. The results show that the ERDU scheme has better performance in terms of energy consumption and network lifetime than the existing RD update mechanism.

The rest of this paper is organized into several sections. In Sect. 2, the detailed operation of the ERDU scheme is presented. Then, the simulation setting and results are presented in Sect. 3 followed by the conclusions in Sect. 4.

2. ERDU Operation

The ERDU scheme is designed to solve the energy inefficiency problem of the existing RD update mechanism in the centralized resource discovery approach of the CoAP. For this, the ERDU scheme functions via two mechanisms: dead node detection and update period adaptation. In the following subsections, the operation of the ERDU scheme is described in detail.

2.1 Dead node detection

In the existing RD update mechanism, the resource entries of the RD are alive only during the lifetime that is marked in the update message from the server. Therefore, in cases that the server is abruptly terminated, its RD entries cannot be updated. More specifically, if the server does not respond to the client's request, the client repeatedly sends the request to the server. Nevertheless, if there is no response to a given number of requests, the client requests the resource information of the server from the RD. Then, the RD searches for the resource of the corresponding server and sends it back to the client. However, in cases that the server is dead, the RD responds with an old resource of the server to the client as if the server were still alive, because the RD is unable to receive any update messages from the server. To address this problem, we propose the dead node detection mechanism, which allows the client to detect dead servers and, if detected, removes the resources of the corresponding server from the entry of the RD.

In the dead node detection mechanism, the client sends a request message to the server using the old resource received from the RD. This operation is repeated for a predefined number of times. If the number of operations exceeds the predefined number, the client determines that the server is dead. Then, it requests the RD to delete the old resource of the server. Finally, the RD deletes the old resource of the server.

Figure 1 shows an operational example of the dead node detection mechanism. As shown in the figure, the server is dead owing to failure or power shutdown; thus, it does not respond to the multiple requests of the client. In this case, the client sends requests to the RD to obtain the resource of the server. Upon receiving the request message, the RD responds to the client with

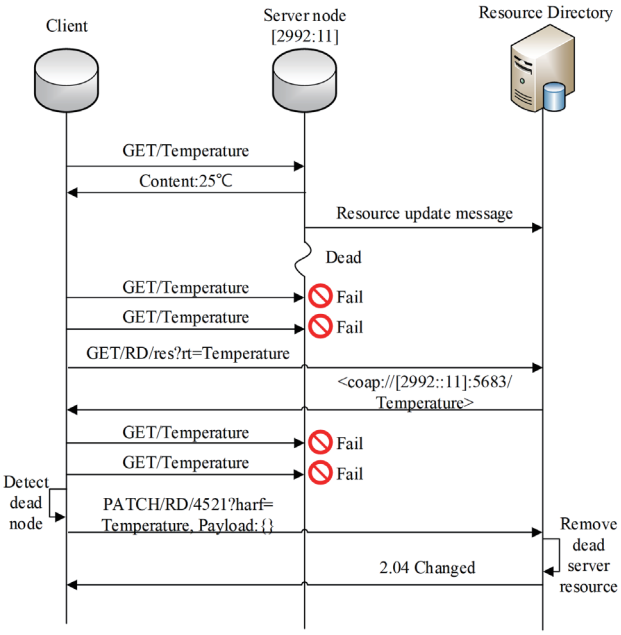

Fig. 1. (Color online) An example of dead node detection.

an old resource of the server. However, the client is still unable to receive the response message from the server since the server is dead. When the client does not receive the response from the server after two attempts, it determines that the server is dead and requests the RD to delete the resource. Finally, the RD deletes the resource and informs the client of the success in deleting the resource.

2.2 Update period adaptation

The RD periodically receives update messages from servers to maintain the latest resources. However, this operation may cause additional signaling overhead and decrease the network lifetime when an inadequate update message period is used. For example, if the update message period is set to very short, the server consumes considerable energy due to the frequent transmission of update messages. In opposite cases, the RD is unable to maintain the latest resources in its entry. To address this problem, the ERDU scheme includes an update period adaptation mechanism, which aims to improve the lifetime and energy efficiency of the servers and to maintain the latest resource in RD entries. To this end, the servers adjust their update message period depending on their energy level.

The server initially sets the multiplying factor to limit the maximum length of the update message period and then starts to check its remaining energy level. After that, to determine the new update message period, the server uses two parameters (i.e., multiplying factor and remaining energy level) with the initial update message period. The multiplying factor is a constant value that is used to set the maximum length of the update message period. For example, if the initial update message period is 10 ms and the multiplying factor is 5, the maximum length of the update message period is 50 ms. When the remaining energy level of server decreases, the update message period should be increased to conserve the energy of the server; thus, the remaining energy level is considered for the adaptation of the update message period. The new update message period is represented by Eq. (1):

$$UP_{new} = \{UP_{initial} \times (k-1) \times (1-REL)\} + UP_{initial}, \tag{1}$$

where UP_{new} is the new update message period, $UP_{initial}$ is the initial update message period, k is the multiplying factor, and REL is the remaining energy level of the server.

3. Performance Evaluation

An experimental simulation is conducted to evaluate the performance of the ERDU scheme using MATLAB. To verify the effectiveness of the dead node detection and update period adaptation mechanisms, we have performed the simulation under various scenarios and compared the results of the ERDU scheme with that of the legacy RD mechanism. In the following subsections, the simulation setting and results are described in detail.

3.1 Simulation setting

In this simulation, we employ a single client, RD, and server and assume that the servers can check their remaining energy levels. We set a server with a battery capacity of 1000 mA and to consume 11.6 mA for Tx, 12.3 mA for Rx, and 0.4 mA for sleep. The initial update message period is set to 10 ms. In addition, we set the minimum energy level so that the server is alive at 5%. In other words, if the remaining energy level of the server is less than 5%, the server is considered as discharged. The detailed simulation parameters are listed in Table 1.

3.2 Simulation results

Figure 2 shows the variation in the client's energy consumption as the simulation time increases. In the simulation, we investigate the effect of dead node detection by measuring the client's energy consumption every 10 ms. As shown in the figure, the ERDU scheme consumes less energy than the legacy RD mechanism when the simulation time reaches 80 ms. This difference may be attributed to the detection of the dead node at 80 ms by the client using the ERDU scheme; thus, it no longer sends the request message to the server. On the other hand, the client using the legacy RD mechanism does not know whether the server is dead. In this case, the client continues to send the request message even after 80 ms, resulting in unnecessary energy consumption.

Figure 3 shows the variation in the server's energy level during simulation. In the simulation, the effect of update period adaptation is investigated by measuring the server's energy level every 10 ms. As shown in the figure, the lifetime of the ERDU scheme is three times longer than that of the legacy RD mechanism since it adjusts its update message period according to the remaining energy. Specifically, the server lives until 2600 ms in the case of ERDU but lives only until 800 ms in the case of legacy RD.

Table 1
Simulation parameters.

Parameter	Value
Tx energy consumption	11.6 mA
Rx energy consumption	12.6 mA
Sleep energy consumption	0.4 mA
Initial update message period	10 ms
Maximum battery capacity	1000 mA
Minimum energy level	5%
Multiplying factor	5

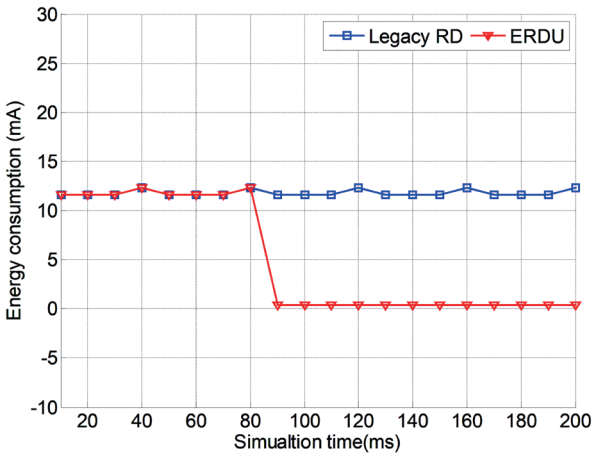

Fig. 2. (Color online) Variation in the energy consumption of the client.

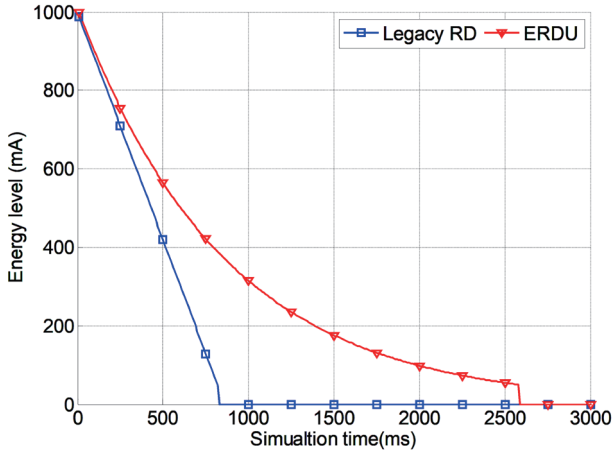

Fig. 3. (Color online) Variation in the energy level of the server.

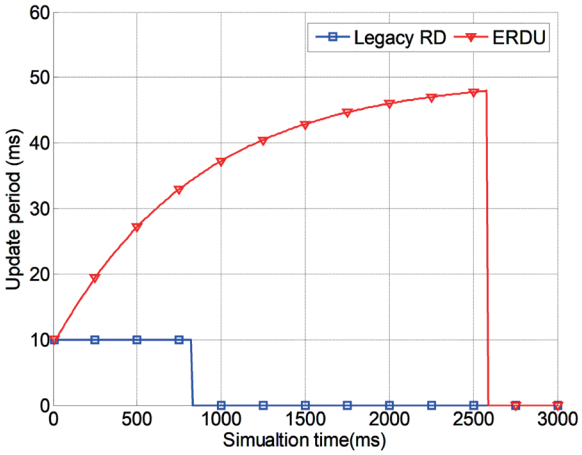

Fig. 4. (Color online) Variation in the update period of the server.

Figure 4 shows the change in the server's update message period according to simulation time. As shown in the figure, the update message period of the server using the ERDU scheme increases to approximately 50 ms; this is because the multiplying factor is set to 5. On the other hand, the server using the legacy RD mechanism maintains the update message period at 10 ms, which is the same as the initial update message period. Therefore, it consumes more energy than the server using the ERDU scheme because of the frequent transmission of update messages.

4. Conclusions

In this paper, we present an ERDU scheme for the CoAP, which prevents unnecessary energy consumption caused by dead nodes and the transmission of update messages. To improve the energy efficiency of servers, the ERDU scheme functions via two main mechanisms: (1) dead node detection and (2) update period adaptation. In the former mechanism, the ERDU scheme detects dead servers and removes their resource from RD entries. In the latter, the server adjusts the update message period on the basis of its remaining energy level. To verify the effectiveness of ERDU, an experimental simulation is conducted under various scenarios. In comparison with the legacy RD update mechanism, the ERDU scheme is able to reduce the client's energy consumption by 87% and increase the server's lifetime by 315%.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2017R1D1A1B03031055). This work was also supported by the Hallym Leading Research Group Support Program of 2017 (HRF-LGR-2017-0003).

References

1 Z. Shelby, K. Hartke, and C. Bormann: The Constrained Application Protocol (CoAP) RFC 7252 (2014).
2 C. Bormann, A. P. Castellani, and Z. Shelby: IEEE Internet Comput. **16** (2012) 62.
3 B. Djamaa, A. Yachir, and M. Richardson: J. Ambient Intell. Hum. Comput. **8** (2017) 357.
4 Y. Kim, K. H. Kim, T. Shon, and J. H. Kim: Proc. 2015 Int. Conf. Information Networking (IEEE, 2015) 407.
5 Z. Shelby, M. Koster, and C. Bormann: CoRE Resource Directory draft-ietf-core-resource-directory-11 (2017) (Internet-Draft).
6 B. C. Villaverde, R. P. Alberola, A. J. Jara, S. Fedor, S. K. Das, and D. Pesch: IEEE Commun. Surv. Tutorials. **16** (2014) 41.
7 M. Qasem, A. Al-Dubai, and M. B. Yassien: Proc. 2015 IEEE Int. Conf. Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (IEEE, 2015) 1118.
8 M. B. Yasin, Q. Abuein, A. B. Amer, and M. Qasem: Proc. Int. Conf. Engineering & MIS (IEEE, 2016) 1.