

CoRE Group
Internet Draft
Intended status: Informational
Expires: December 29, 2010

A. Rahman
JC. Zuniga
G. Lu
InterDigital Communications, LLC
June 29, 2010

Sleeping and Multicast Considerations for CoAP
draft-rahman-core-sleeping-00.txt

Abstract

This document further analyzes the COAP requirements related to "sleeping nodes" and "multicast" through the use of examples and use cases. The goal of this document is to trigger discussions in the CoRE working group so that all relevant considerations are taken into account when designing CoAP.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>

This Internet-Draft will expire on December 29, 2010.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](http://trustee.ietf.org/license-info) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the BSD License.

Table of Contents

1.	Introduction.....	2
2.	Conventions and Terminology.....	3
3.	Handling Sleep Nodes.....	3
3.1.	General Behavior of Sleep Nodes.....	3
3.2.	Handling Sleeping Nodes within CoAP.....	4
3.3.	Use Cases.....	7
3.3.1.	Use Case 1: Smart Meter Reading.....	7
3.3.2.	Use Case 2: Smart Meter Firmware Upgrade.....	9
3.3.3.	Use Case 3: Obtaining Configuration.....	10
3.3.4.	Use Case 4: Constrained Node Sending Report.....	12
4.	Multicast Support in CoAP.....	13
5.	Conclusions.....	14
6.	Security Considerations.....	14
7.	IANA Considerations.....	14
8.	References.....	14
8.1.	Normative References.....	14
8.2.	Informative References.....	15
9.	Acknowledgments.....	15

[1.](#) Introduction

The CoRE working group is chartered to design and standardize a Constrained Application Protocol (CoAP) for resource constrained devices and networks [I-D.[draft-ietf-core-coap](#)]. The requirements for CoRE are well documented group [I-D.[draft-shelby-core-coap-req](#)].

In this draft, we focus and expand discussions on some of these key requirements pertaining to sleeping nodes and multicast support. Specifically, the requirements we analyze are:

REQ 3: The ability to deal with sleeping nodes. Devices may be powered off at any point in time but periodically "wake up" for brief periods of time.

REQ 4: Protocol must support the caching of recent resource requests, along with caching subscriptions to sleeping nodes.

REQ 9: CoAP will support a non-reliable IP multicast message to be sent to a group of Devices to manipulate a resource on all the Devices simultaneously. The use of multicast to query and advertise descriptions must be supported, along with the support of unicast responses.

[2.](#) Conventions and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [[RFC2119](#)].

The following are definitions of terminologies used in this draft.

Sleep Mode: Refers to the state when a device is not able to send and/or receive any messages due to power conservation from the perspective of an application protocol (i.e. CoAP).

Sleep Node: A device that may go to sleep mode from time to time.

Sleeping Node: A device that is in sleep mode.

[3.](#) Handling Sleep Nodes

3.1. General Behavior of Sleep Nodes

In this section we discuss different behaviors and scenarios of sleeping nodes. Such behaviors can affect the design of applications

(such as CoAP) and network topologies (such as proxies and caching).

Sleeping nodes can have various sleeping patterns. Sleep patterns can be predictable or totally unpredictable. For example, some nodes sleep at a fixed interval or upon certain triggers. Some nodes may sleep at irregular time intervals, or switch to sleep mode spontaneously. Some nodes stay in sleep mode and only wake up upon

certain event triggers. A network may thus not be well aware of the sleeping state of a node at a given time.

The duration of sleeping mode also varies largely, possibly from a few seconds to days. From the perspective of applications, it may not be affected by the sleeping period if it is very short. For example, a HTTP/TCP connection may still work (even if sub-optimally) if the sleep cycle is much shorter than the TCP retransmission timer. In contrast, if the sleeping period of a node exceeds a certain threshold it can impact an application. This threshold however, can be difficult to predict and often can vary from device to device and network to network. For example, this threshold can be very dependent on the topology of a constrained network especially for the case where a multi-hop path consists of multiple sleeping nodes. For this case, the cumulative effect of multiple sleeping nodes must be considered.

The network topology also affects how to handle sleeping nodes. For example, in a star shaped network, a proxy node (assuming to be not a sleeping node) can cache for the sleeping nodes within the network in a centralized manner. However, in a P2P or mesh network, especially when multi-hops are involved, caching can be difficult and delivering of messages can be largely delayed due to nodes' sleeping cycles. In this case distributed proxying and caching at intermediate nodes within the network (rather than just a single node such as the border node or sink) may make sense, if intermediate nodes are not sleeping nodes and have adequate resources to support caching.

3.2. Handling Sleeping Nodes within CoAP

Traffic generated for handling sleeping nodes should be minimized in

a constrained network. A constrained device can delegate its authority to the proxy node for operations on its behalf. Therefore operations between the constrained device and its counterpart (e.g. a computer on the Internet) become asynchronous due to the proxy.

Below we discuss the various aspects to consider for handling sleeping nodes with CoAP.

1) Exchange of sleeping schedule

A node can optionally include its sleep schedule and exchange such information with other nodes and/or proxies. This provides proxy(s) with better awareness of all resources that they are able to proxy

for as well as which devices sleep and their corresponding sleep cycle durations. Using this sleep information, the proxy(s) could choose which device resources it wants to create/maintain intercept caches for (e.g. support intercept caches only for sleeping devices), thus optimizing resource utilization within the proxy by selectively caching and also minimizing the impact on sleeping nodes by allowing them to sleep longer and not have to directly service incoming requests.

2) Proxy and device synchronization

A common synchronization scheme is to have the device always inform the proxy of its latest sleep state by checking-in [I-D.frank-6lowpan-chopan]. For example, the proxy can subscribe to a node and the node sends notification each time when it wakes up. However, this approach increases the traffic for managing the sleep state, especially when there are a large amount of nodes and they sleep frequently.

A proxy can benefit from being synchronized to its associated constrained device's sleep and awake state. Ideally, if the proxy and device are synchronized in time and if the device has a predictable sleep schedule, the proxy can know whether the device is awake or not without the device having to check-in so often. However, time synchronization may be difficult to achieve for constrained devices.

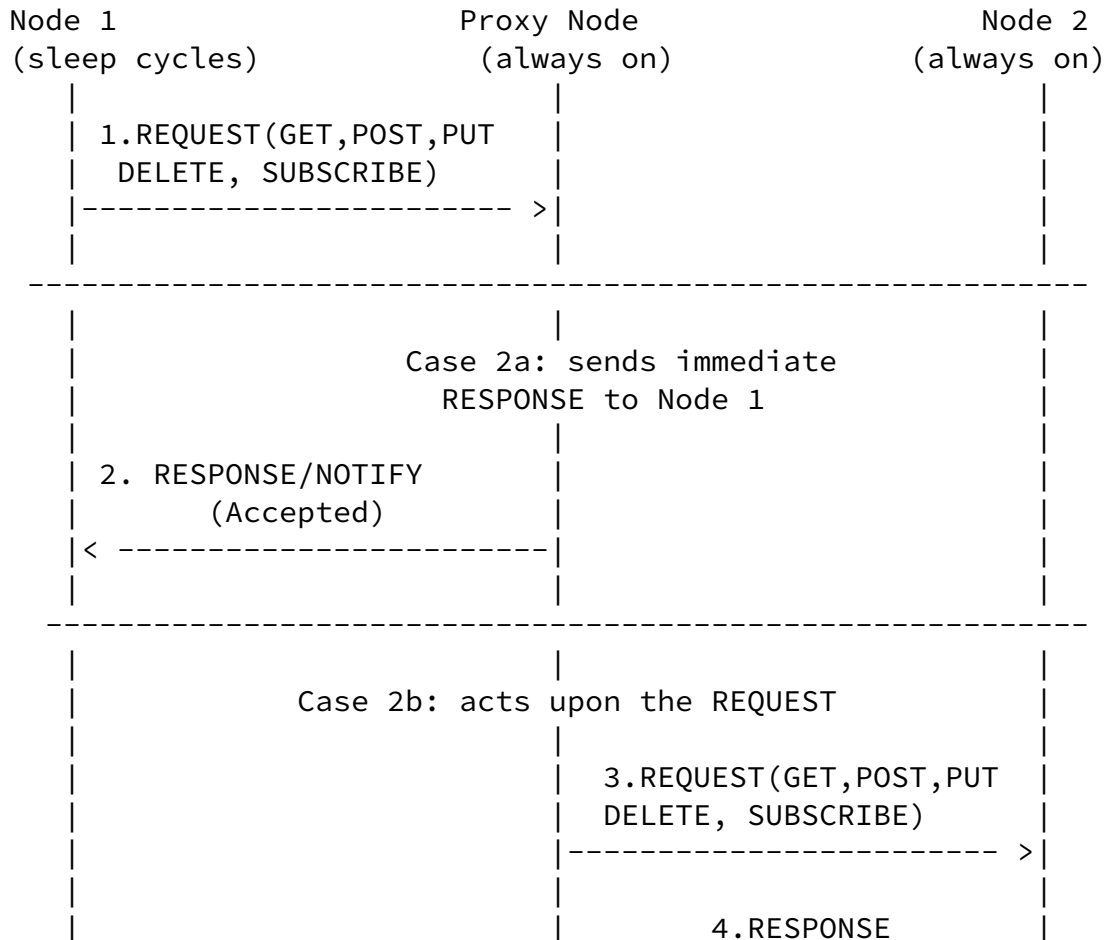
3) Caching and Proxying

It is assumed that the proxy node is awake all the time. A proxy can thus always cache for the constrained nodes, regardless of the node's sleep state. The proxy node would then act as an interception proxy. This is analogous to the concept of web caching (e.g. HTML pages,

images, etc.) between a Web server and a Web browser used in the general Internet.

Figure 1 shows a call flow when a constrained node (Node 1) sends a REQUEST to Node 2. The REQUEST can contain any method, GET, POST, PUT, DELETE, or SUBSCRIBE. The figure illustrates two scenarios. In Case 2a, the proxy sends a RESPONSE with intermediate status. After that, Node 1 may go to sleep mode, and receive the requested content in step 6 after it wakes up. In Case 2b, the proxy acts upon the REQUEST immediately. Node 1 may go into sleep mode at any time after sending the REQUEST, if it can handle delayed response. The proxy node is in sync with Node 1 for its sleep state and caches the

RESPONSE for it when it is in sleep mode. When Node 1 wakes up, it can notify the proxy with its sleep schedule (if changed) and send data for caching as the procedure shown in step 5.



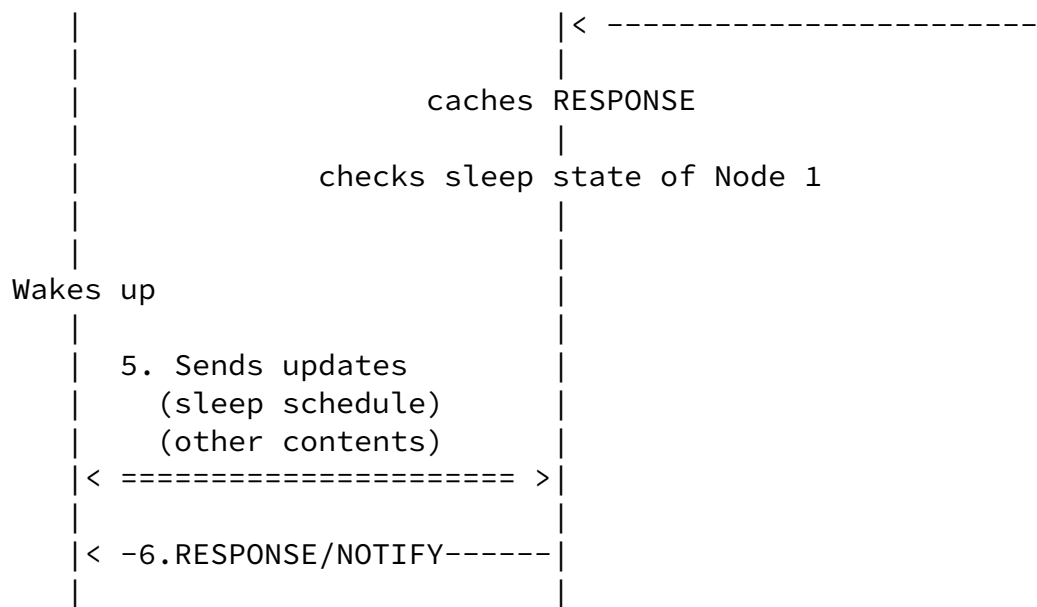


Figure 1 Caching and Proxying

3.3. Use Cases

In this section, different scenarios are illustrated to show the operations of a proxy node for GET and non-GET REQUESTs. The REQUEST can be from the constrained or non-constrained domain. For the following figures, Node 1 is a constrained node and it goes to sleep from time to time. Node 2 is a node in the non-constrained domain. It is assumed that the proxy node and Node 2 do not go to sleep. The protocol between Node 1 and the Proxy Node is CoAP, and the protocol between the Proxy Node and Node 2 is HTTP.

The figures show some key scenarios but are not exhaustive.

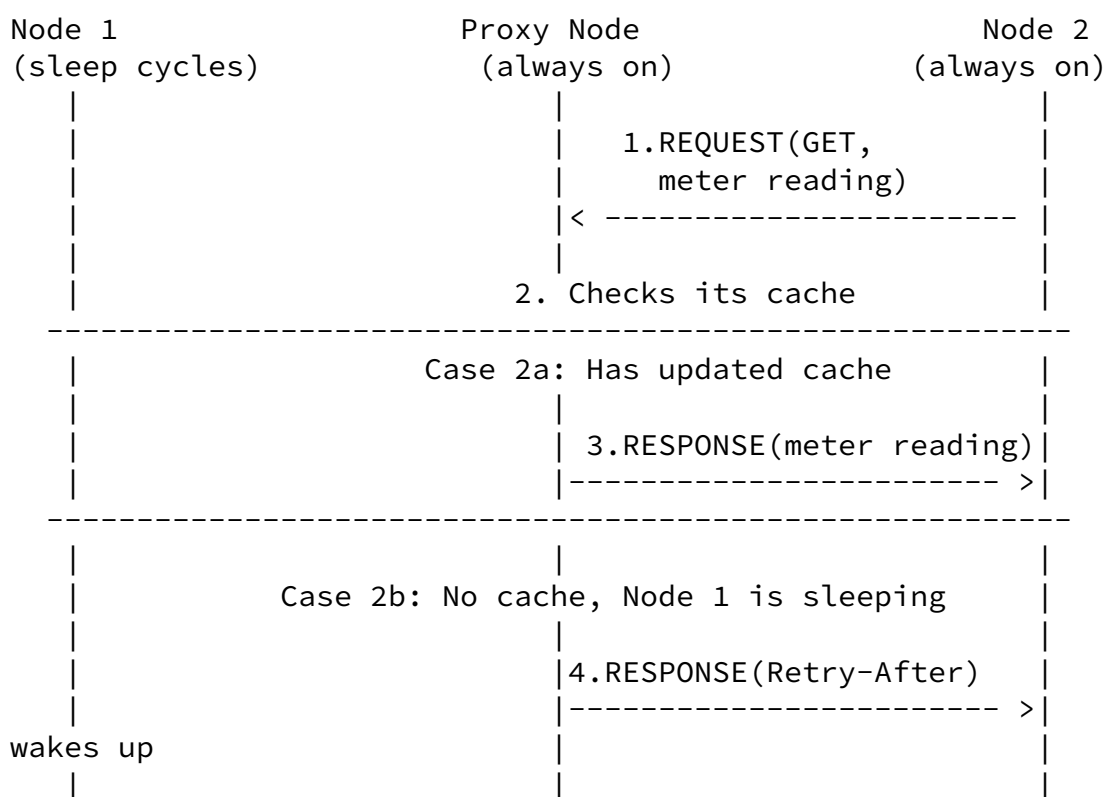
3.3.1. Use Case 1: Smart Meter Reading

In use case 1, Node 1 is a smart meter in the constrained network, and Node 2 in the non-constrained network wants to read the data from Node 1.

The proxy can enable asynchronous communication between Node 1 and Node 2. In situation 2a, since the proxy node has an updated cache for what Node 2 is requesting, the proxy can respond to Node 2, regardless of Node 1's sleep state.

In situation 2b, if the proxy does not have an updated cache, and Node 1 is sleeping, the proxy can send a RESPONSE to Node 2 with RETRY-AFTER information. The proxy would need to be synchronized with Node 1 and have Node 1's sleep schedule. Otherwise the proxy has to poll Node 1, which can cause a long delay and extra messaging. The synchronization is shown in step 5. When Node 1 wakes up, it can notify the proxy with its sleep schedule (if changed) and send data for caching.

Alternatively, as shown in step 2b', the proxy may buffer the REQUEST and only send a RESPONSE when it gets information from Node 1. This can reduce messaging for Node 2 to query again. The proxy should know the sleep schedule of Node 1 to make such a decision on how it reacts. If Node 1 is not waking up in a very long time, it is probably better for the proxy to send RESPONSE to Node 2 immediately. Otherwise the long delay can cause time-out and retry of Node 2.



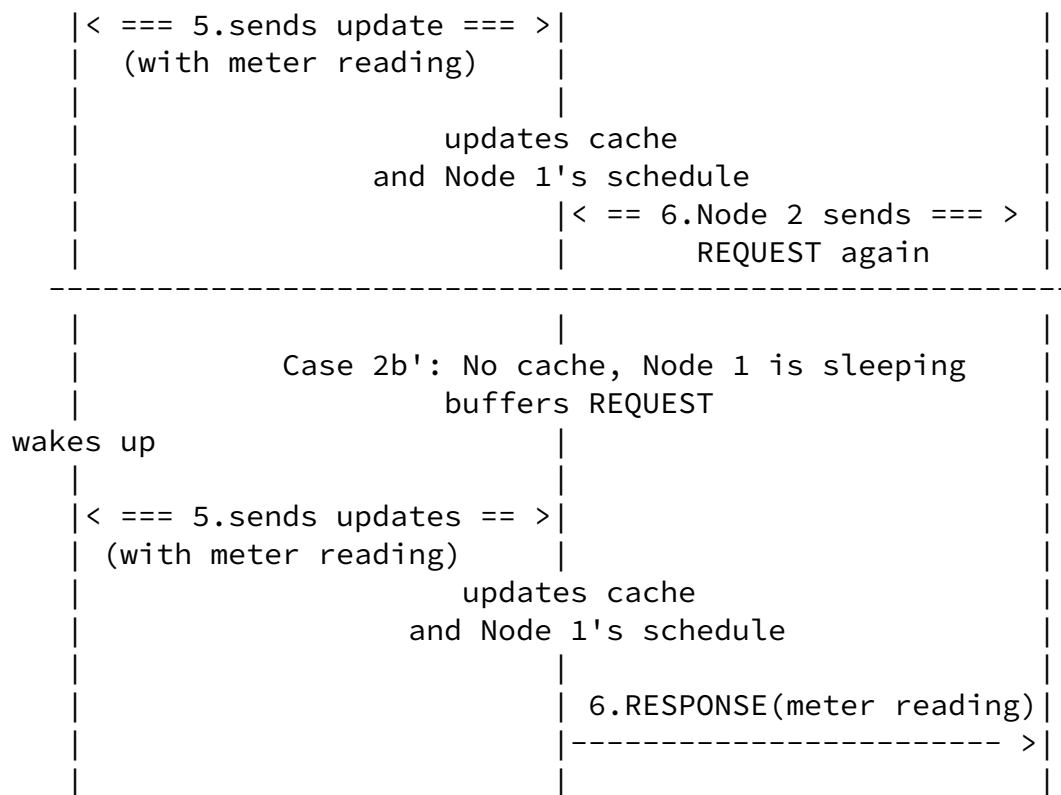
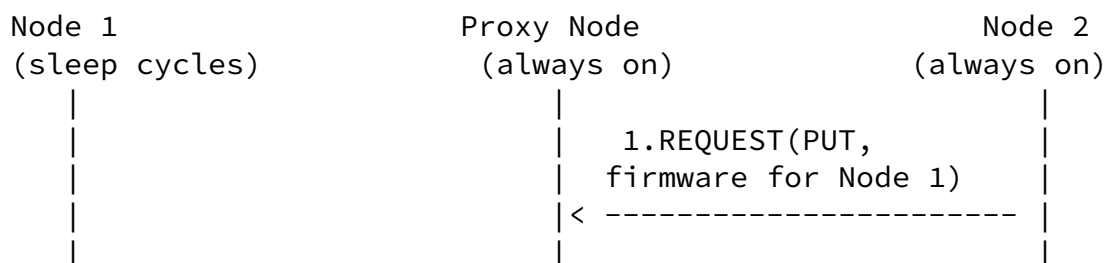


Figure 2 Use Case 1: Smart Meter Reading

3.3.2. Use Case 2: Smart Meter Firmware Upgrade

In use case 2, Node 1 is a smart meter in a constrained network and Node 2 upgrades Node 1's firmware using PUT. If the proxy knows that Node 1 is sleeping, it can either advise Node 2 to retry later, or the proxy can buffer the REQUEST till Node 1 wakes up.



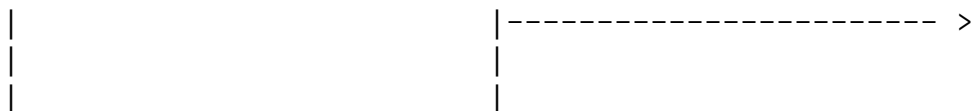
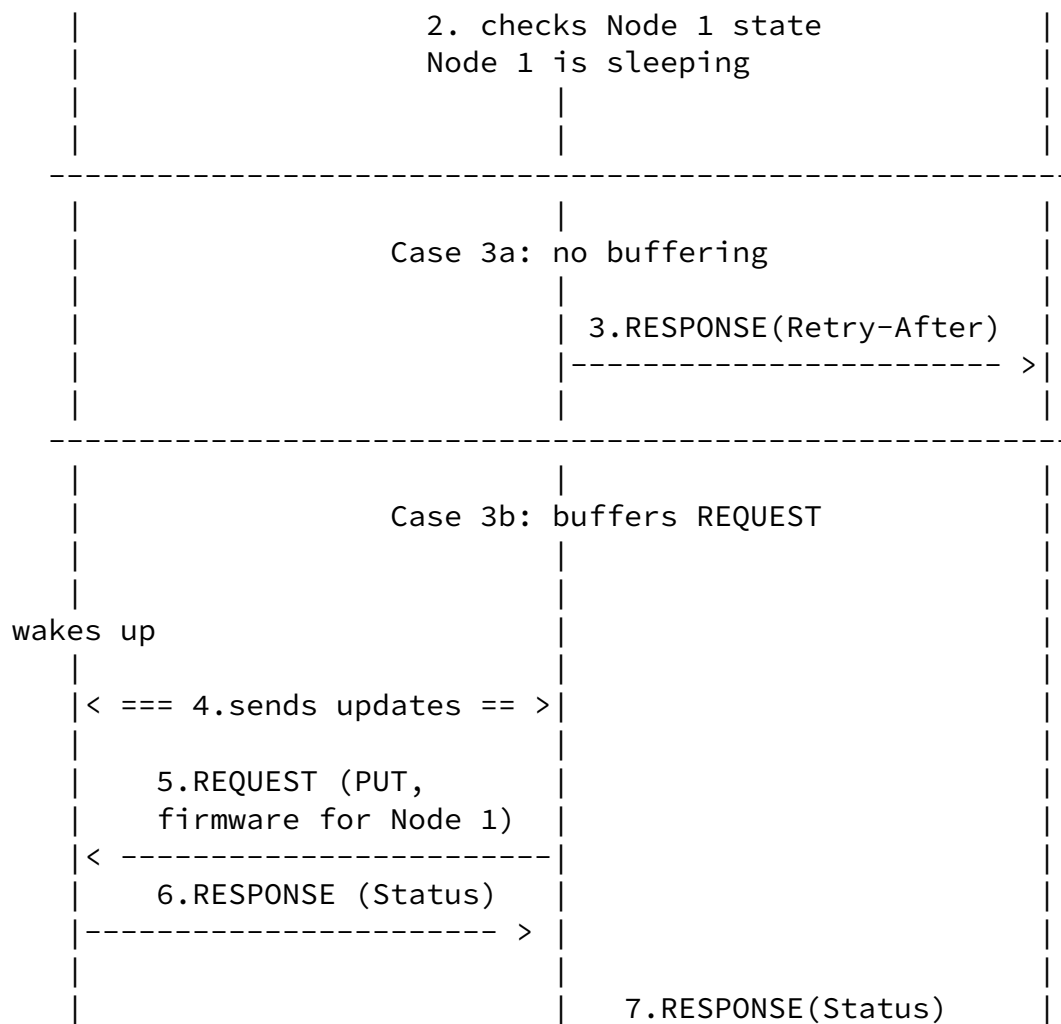
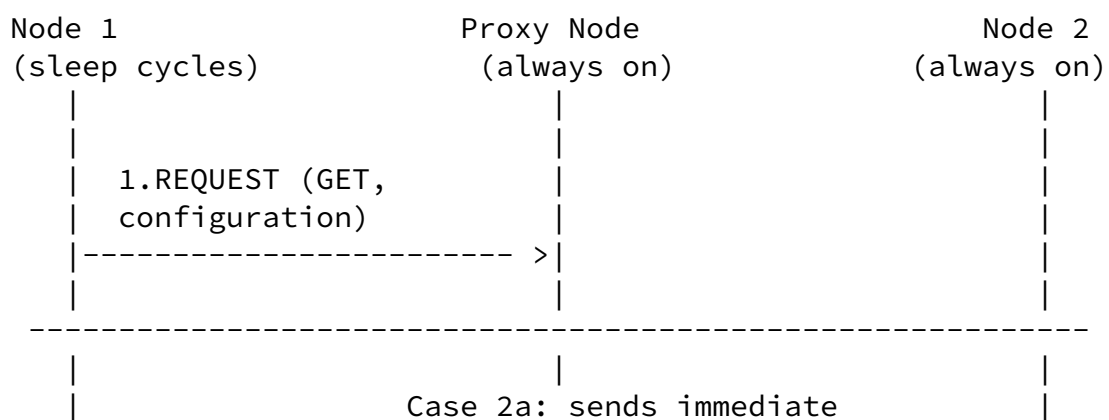


Figure 3 Use Case 2: Smart Meter Firmware Upgrade

3.3.3. Use Case 3: Obtaining Configuration

Use case 3 shows that Node 1 in a constrained network tried to obtain a configuration parameter from Node 2. Since the proxy is aware that Node 1 is a sleep node, the proxy sends RESPONSE to Node 1

immediately so that Node 1 can go to its sleep cycle. The proxy then requests and caches the configuration parameter for Node 1, and it can notify Node 1 when Node 1 wakes up.



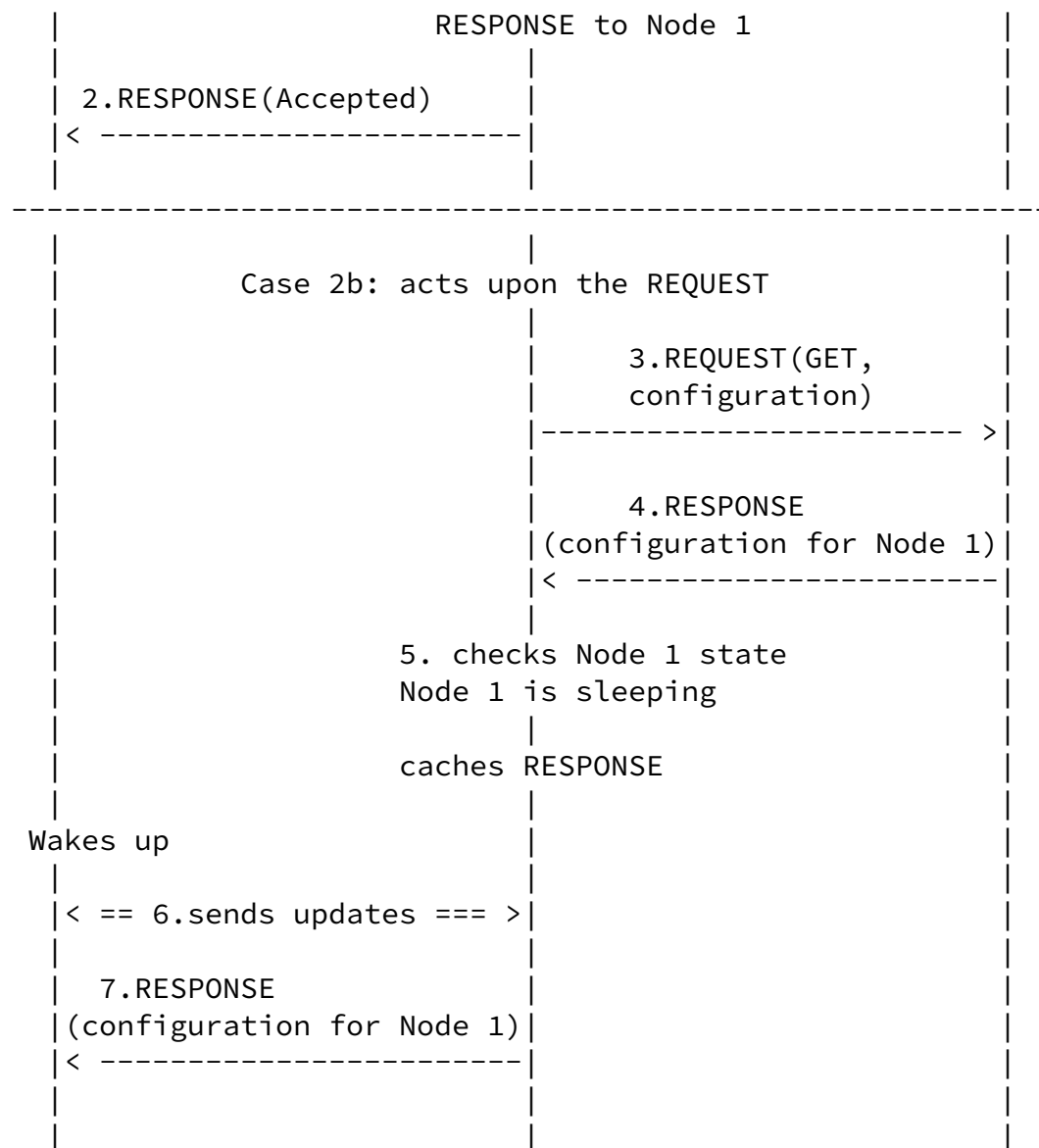


Figure 4 Use Case 3: Constrained Node Getting Configuration

3.3.4. Use Case 4: Constrained Node Sending Report

Use case 4 shows a smart meter Node 1 sends report to Node 2.

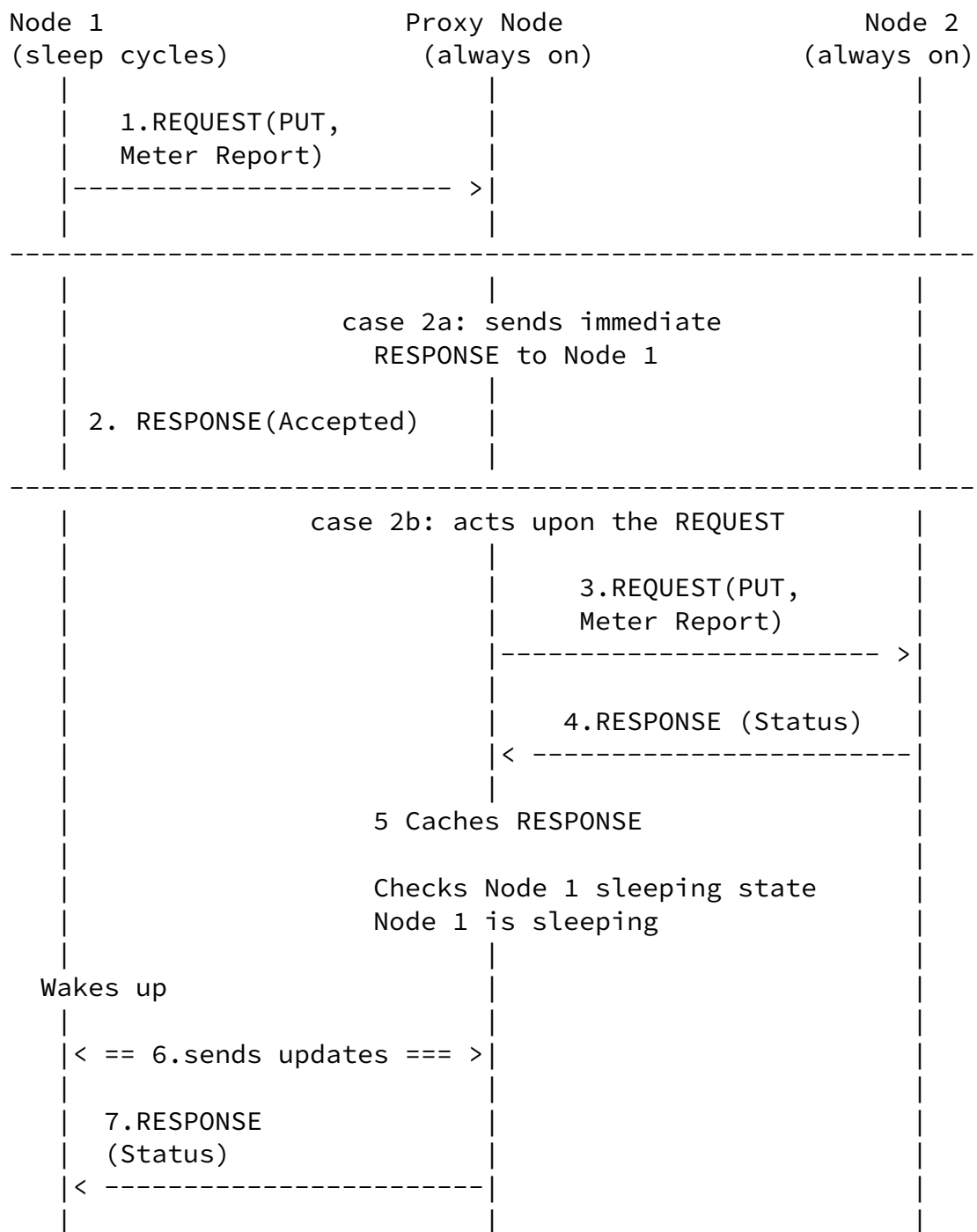


Figure 5 Use Case 4: Constrained Node Sending Report

4. Multicast Support in CoAP

One of the requirements of the CoAP design is to support a simple and unreliable multicast method. Supporting multicast poses additional requirements to the proxy node.

Within the constrained network, an application protocol (e.g. CoAP) usually runs over UDP for which IP multicast is supported. In a non-constrained network (i.e. general Internet), HTTP over TCP is typically used for which IP multicast is not naturally supported. This causes the scenario below:

- o The traffic within the constrained network is multicast (such as CoAP over UDP), and within the non-constrained network it is unicast (such as HTTP over TCP)

Therefore the proxy node needs to have functionality to support interworking of unicast and multicast. Specifically, CoAP protocol should include support for translating a HTTP/TCP unicast request into a CoAP/UDP multicast request as illustrated in Figure 6.

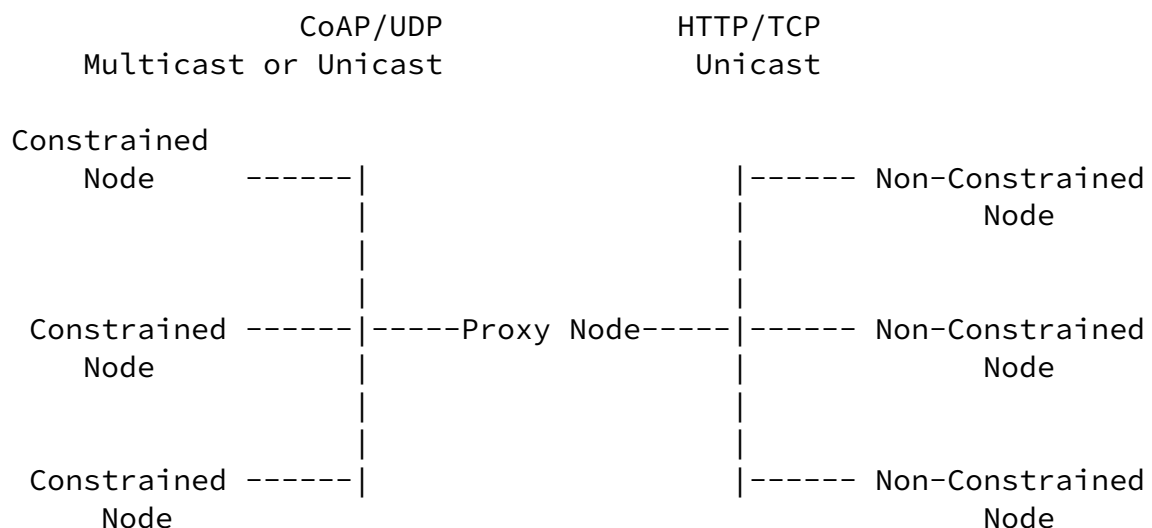


Figure 6 Multicast Support at Proxy Node

5. Conclusions

This draft analyzes three requirements identified in the CoAP design requirement document [I-D.[draft-shelby-core-coap-reg](#)] related to handling sleeping nodes and supporting multicast. The draft discusses some considerations for CoAP design and proxy operations to meet these requirements.

For a constrained network to handle sleeping nodes, a constrained node should be able to notify the proxy of its sleep schedule, and the proxy node should have an updated schedule for each sleep node. To utilize the sleep schedule, the sleep nodes and the proxy need to maintain synchronization to a certain extent. A proxy node thus enables asynchronous communications with the general Internet, since a proxy node can always cache for the constrained nodes.

CoAP is targeted to interact with HTTP/TCP on the general Internet. Since IP multicast does not support TCP, the proxy node in a constrained network needs to have functionalities to support interworking of multicast and unicast to fulfill this requirement.

6. Security Considerations

This draft discusses the design considerations and operations of CoAP and CoAP proxy in constrained networks. It does not introduce new security threats.

7. IANA Considerations

This document makes no request of IANA.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.

Internet-Draft

Sleeping and Multicast for CoAP

June 2010

8.2. Informative References

[I-D.[draft-ietf-core-coap](#)] Shelby, Z., Frank, B., and D. Sturek, "Constrained Application Protocol (CoAP)", [draft-ietf-core-coap-00](#) (Work in progress), June 7, 2010.

[I-D.[draft-frank-6lowpan-chopan](#)] Frank, B., "Chopan - Compressed HTTP Over PANs", [draft-frank-6lowpan-chopan-00](#) (Work in progress), June 15, 2009.

[I-D.[draft-shelby-core-coap-req](#)] Shelby, Z., Stuber, M., Sturek, D., Frank, B., and R. Kelsey, "CoAP Requirements and Features", [draft-shelby-core-coap-req-01](#) (Work in progress), April 20, 2010.

9. Acknowledgments

Thanks to Jean-Louis Gauvreau and Dale Seed for their useful comments and discussions.

Authors' Addresses

Akbar Rahman
InterDigital Communications, LLC
Email: Akbar.Rahman@InterDigital.com

Juan Carlos Zuniga
InterDigital Communications, LLC
Email: JuanCarlos.Zuniga@InterDigital.com

Guang Lu
InterDigital Communications, LLC
Email: Guang.Lu@InterDigital.com

