CoRE Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 16, 2014

K. Hartke
Universitaet Bremen TZI

July 15, 2013

Liveliness and Cancellation of Separate Responses and Observations
draft-hartke-core-coap-liveliness-00

Abstract

This document extends the Constrained Application Protocol (CoAP) with a simple mechanism for a client to determine the "liveliness" of a request for which it is still expecting a response or a notification, and with a mechanism to request the cancellation of such a request.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on January 16, 2014.

Table of Contents

1. Requirements Notation

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 RFC 2119 [RFC2119].

2. Liveliness

When a CoAP server [I-D.ietf-core-coap] needs longer to process a request than it has time before sending back an acknowledgement, it can acknowledge the request message first and return the response at a later time (Separate Response). However, a client that does not receive any response for some time (despite the acknowledgement, which effectively is the server's promise to send a response) cannot know if the server simply hasn't finished processing the request yet or, for example, if the server had to reboot and thus couldn't finish the task.

Similarly, when a client observes a resource [I-D.ietf-core-observe] but hasn't received a notification for some time, it cannot know if the resource state simply did not change or if the server forgot the the client's interest in the resource. As long as the client still has a fresh representation of the target resource, this is not a problem. However, eventually the client must decide whether it needs to register its interest in the resource again or not (provided that it's still interested in the resource). If the client at this point makes the wrong decision, it ends up being in the list of observers of the resource either twice or not at all, both of which is not desirable.

This document extends CoAP with a simple mechanism for a client to determine the "liveliness" of a pending request. The liveliness check consists of the exchange of two CoAP messages: a "ping" and a "pong".

Ping: An Empty, Confirmable message that specifies the Token of a request for which the client is still expecting a response or a notification.

Pong: An Empty Acknowledgement message that is returned by the server if it recognises the token and wishes to reinforce the promise to send a response or further notifications; otherwise, an Empty Reset message.

The mechanism is thus just a small extension of the "CoAP ping" defined in [I-D.ietf-core-coap]. A short example is shown in Figure 1 below.

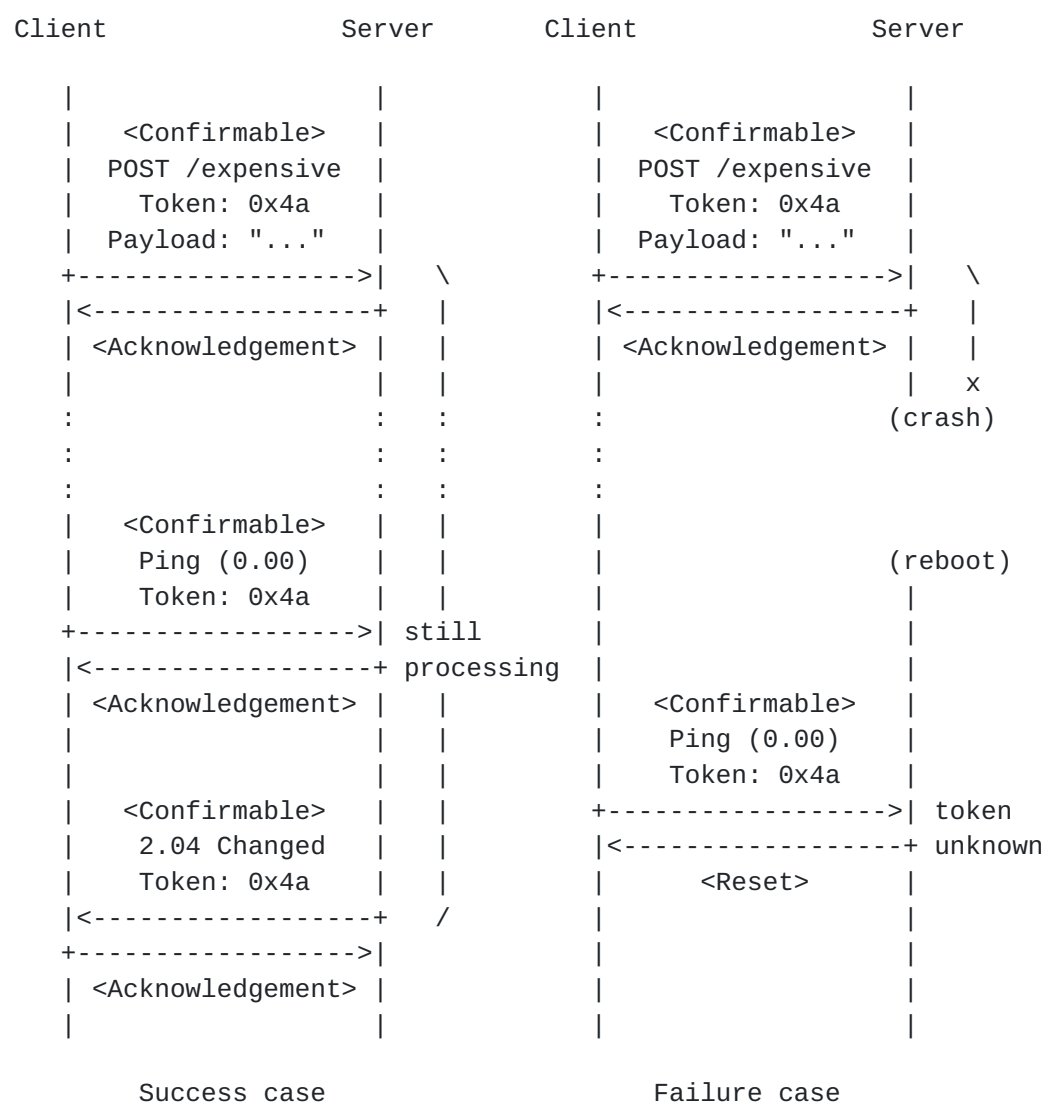

Figure 1: Liveliness check of a request with token 0x4a

A client that is expecting a response or notification SHALL NOT assume that the associated token is no longer in use until it has sent a Ping and received a matching Pong.

Furthermore, the client SHOULD NOT make a new resource-consuming request to the server until it has sent a Ping and received a matching Pong, such as a POST request taking significant processing time or a GET request with an Observe Option. The client MAY significantly increase the number of retransmit attempts (MAX_RETRANSMIT) for a Ping if necessary, and MAY truncate the retransmission timeout to a maximum of no less than 60 seconds.

Support for this mechanism is REQUIRED.

3. Cancellation

A CoAP client that is no longer interest in receiving a Separate Response or further notifications while observing a resource, can simply "forget" the pending request. When the server then sends the response or a notification, the client will not recognize the Token in the message and thus return a Reset message. This signals to the server the lost interest of the client and, in the case of an resource observation, will cause the server to remove the client from the list of observers. The resources allocated by the server to the request are effectively "garbage collected" by the server.

In some circumstances it may be desirable to cancel a pending request and release the resources allocated by the server more eagerly. This document extends CoAP with a mechanism for a client to request cancellation of a pending request.

A client MAY request the cancellation of pending request by sending a Confirmable or Non-Confirmable CoAP message with the Code field set to 7.31 and the Token field set to the token of the request to be cancelled.

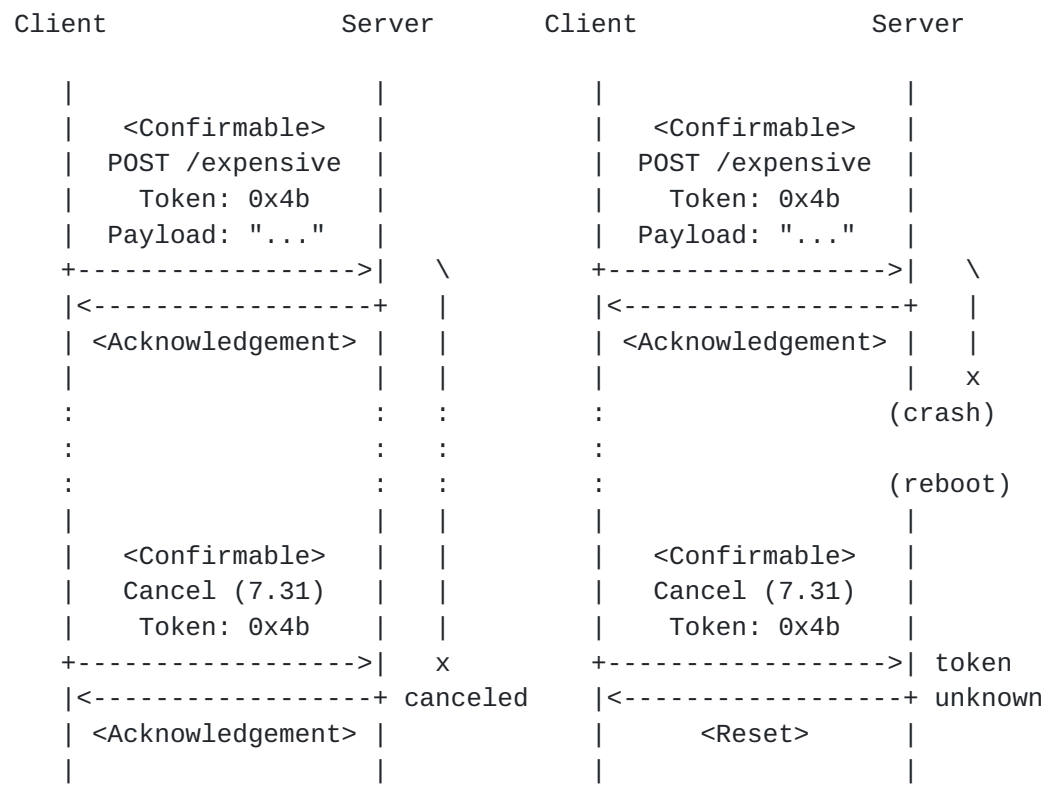

Figure 2: Cancellation of a request with token 0x4b

A server SHOULD NOT send any response or further notification in reply to the specified request after receiving such a message, and MUST remove the client from the list of observers of the target resource of the request.

The server SHOULD abort any ongoing tasks related to the request. However, if it is not possible to abort the tasks without leaving the system in an inconsistent state, it MAY continue to execute the tasks and just suppress the return of the resulting response.

An example is shown in Figure 2 above.

Support for this mechanism is REQUIRED.

4. Security Considerations

(TODO.)

5. IANA Considerations

This document makes no request to IANA.

6. Acknowledgements

Thanks to Matthias Kovatsch for helpful comments and discussions that have shaped the document.

7. Normative References

[I-D.ietf-core-coap]
Shelby, Z., Hartke, K., and C. Bormann, "Constrained Application Protocol (CoAP)", draft-ietf-core-coap-18 (work in progress), June 2013.

[I-D.ietf-core-observe]
Hartke, K., "Observing Resources in CoAP", draft-ietf-core-observe-09 (work in progress), July 2013.

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

Author's Address

Klaus Hartke
Universitaet Bremen TZI
Postfach 330440
Bremen D-28359
Germany

Phone: +49-421-218-63905
Email: hartke@tzi.org