CoRE Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 3, 2018

Y. Liu
J. Zhu
Huawei

October 30, 2017

Mitigating delay attacks on Constrained Application Protocol draft-liu-core-coap-delay-attacks-01

Abstract

Various attacks including delay attack have become a topic in the security of Internet of Things (IoT), especially for the constrained nodes utilizing sensors and actuators which connect and interact with the physical world. [I-D.mattsson-core-coap-actuators] describes several serious delay attacks, discusses tougher requirements and then recommends mechanisms to mitigate the attacks. It also specifies some disadvantages with the mechanisms. This document proposes alternative mechanisms to address some of the disadvantages

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 https://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 May 3, 2018.

Copyright Notice

Table of Contents

1. Introduction 2
2. Terminology 3
3. Attacks 3
4. Solutions 3
4.1. The Repeat Option 3
4.2. The Enhanced Options 4
4.2.1. Simple Single Action Actuators 6
4.2.2. Multi-interrelated Actions 8
5. Security Considerations 9
6. IANA Considerations 9
6.1. Tables 10
7. Acknowledgements 10
8. References 10
8.1. Normative References 10
8.2. Informative References 11
Authors' Addresses 11

1. Introduction

Various attacks including delay attack have become a topic in the security of Internet of Things (IoT), especially for the resource-constrained nodes [RFC7252] utilizing sensors and actuators which connect and interact with the physical world. It is recommended to use the Constrained Application Protocol (CoAP) [RFC7252], which is designed for resource-constrained nodes, and message exchange between them. Also, it is required to use security protocols such as TLS [RFC5246], DTLS [RFC6347], TLS/DTLS profiles for the IoT [RFC7925], or OSCORE [I-D.ietf-core-object-security] to protect CoAP messages due to security and privacy. The security protocols can provide confidentiality, authentication and integrity protection of CoAP messages at both the application layer and the transport layer.

There are still issues related to delay attacks as descirbed in [I-D.mattsson-core-coap-actuators]. For example, [I-D.mattsson-core-coap-actuators] describes several serious attacks, discusses tougher requirements and then recommends solution to mitigate the attacks. The draft also indicates the disadvantage that CoAP messages need two round trips for the solution. This document will show alternative mechanisms which take CoAP messages only one round trip by utilizing the sending messages containing valid time window(s), Sequence Number and Response Policy.

2. Terminology

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

This specification requires readers to be familiar with all the terms and concepts that are discussed in [I-D.mattsson-core-coap-actuators] and [RFC7252].

3. Attacks

It is assumed that the reader is familiar with the following attacks as specified in section 2 of [I-D.mattsson-core-coap-actuators]:

- The Block Attack
- The Request Delay Attack
- The Response Delay and Mismatch Attack
- Relay Attack

4. Solutions

In order to mitigate the attacks as above, [I-D.mattsson-core-coap-actuators] provides a challenge-response mechanism for CoAP using a new CoAP Option "Repeat". This option is described below in section 4.1, which is originally specified in section 3 of the [I-D.mattsson-core-coap-actuators]. An editor's note indicates the disadvantages that the mechanism takes two round trips and provides two potential enhancements utilizing time.

Section 4.2 in this document describes another method which takes only one round trip CoAP messages which utilizes a "Valid Time Window" , "Sequence Number" and "Response Policy" on receiving messages to mitigate the delay attacks.

4.1. The Repeat Option

The Repeat Option is a challenge-response mechanism for CoAP, which is generated by the server and binded to an 4.03 forbidden response. The client bind the same Repeat Option value into a new request to echo the challenge. Then the server verifies the freshness of the original request. An example message flow is illustrated in Figure 1

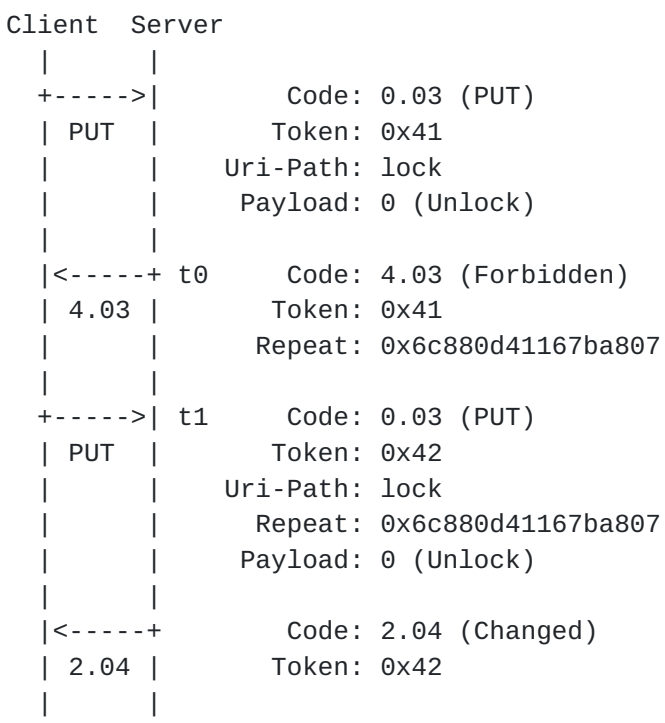

Figure 1: The Repeat Option

1) The client sends the original request without the Repeat Option to a resource on a server with freshness requirements. E.g. the client wants to unlock the door.

2) After receiving the original request, the server sends a 4.03 Forbidden response with a Repeat Option to challenge the client. The repeat Option value and the response transmit time 't0' are stored on the server.

3) The client SHOULD resend the original request with the same Repeat Option value contained in the previous response to echo the challenge. The server SHOULD store the request receive time 't1'.

4) The server firstly verifies that the Repeat Option value equals the previously sent one. Then the server calculates the round-trip time RTT = (t1 - t0). The server MUST only accept requests with a round-trip time below a certain threshold T, i.e. RTT < T. The threshold T is application specific.

4.2. The Enhanced Options

According to the method using a Repeat Option (see Section 4.1), there are still the following potential situations:

- If the RTT of the second message to the third message (see Figure 1) is larger than the certain threshold T, the server can determine that the request message from the client is delayed and then discard it.

- If the RTT of the second message to the third message (see Figure 1) is large but does not exceed the certain threshold T, the server treats these messages as valid and then processes them normally. But these messages may have become invalid especially in the situation where the request(s) containing actions relevant for actuators are required to be processed in a specific and limited period of time. For example, the actuator with the air conditioning may be required to keep it open in a specific time and temperature, which depends on some reasons such as user's preference and current room temperature. In other words, the specific time may be varied, it is possible that the server determines the request is valid by RTT < T but the potential specific time associated with the request is actually past.

- If the RTT of the third message to the fourth message (see Figure 1) is larger than the certain threshold T, it may cause that the client resends the request message but the actuator's actions associated with the previous message has already been processed.

- Regardless of whether the delay exists, the two round-trips increase the delay in overall processing of the original action (e.g. PUT)

In fact, the server cannot accurately know whether the messages are delayed in a reasonable period of time or not, because the reasons for the delay may be caused by the network itself and/or some attacks such as man-in-the-middle. In other words, how to set T value depends on many factors. Also, it is not enough to determine whether the delay happens.

Due to IoT covering different vertical domains actuators have different delay sensitivity requirements. Simple actuators (such as a smart switch) support a single action and may not be delay sensitive. There are others with complicated capabilities that are able to process multi-interrelated actions especially in Industrial Control and Production Systems. These actuators with multi-interrelated actions are usually associated with strict time requirements. Therefore, it is lack of a mechanism that assures them process multi-continuous actions addressed in different request(s) when the delay attack happens and even causes some mismatch/disorder.

4.2.1. Simple Single Action Actuators

For simple single action for the actuators, the Time Window Option is introduced as a new CoAP option and is to address the validity period of the request(s) from the client. The Time Window Option including T-start (i.e., a start valid time) and T-duration (i.e., a valid duration) of a request enables the server to accurately know the freshness of a request, determine how to process it, and thus achieve to mitigate the attacks described in Section 3.

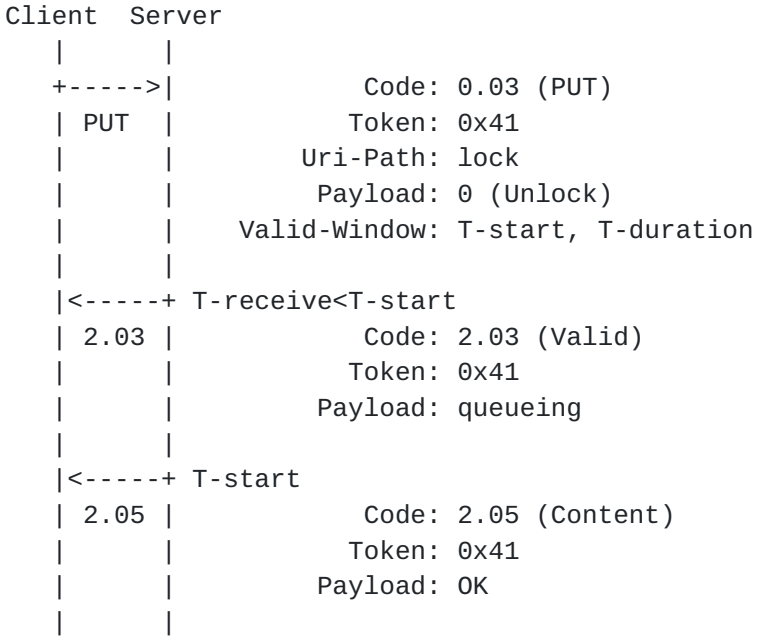

Figure 2: The Time Window Option(1)

Upon receiving a request containing the Time Window Option, the server extracts the T-start and T-duration from the first request from the client.

If T-receive (a reception time for the server receiving a request) < T-start as illustrated in Figure 2, it means that the server SHOULD not do the actions carried in the request until T-start is coming. The server SHOULD add this request to a waiting queue, and issues a temporarily response (e.g. 2.03) to the client with the payload indicating "queueing". When T-start is coming, the server gets the corresponding request from the processing buffer, executes the actions carried in the request, and sends a response 2.05 containing a payload indicating "OK".

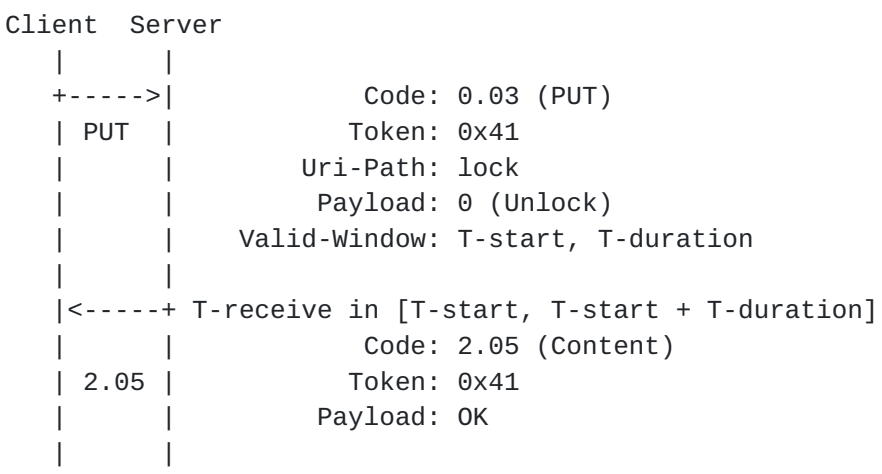

Figure 3: The Time Window Option(2)

If T-receive (i.e., a reception time for the server receiving a request) >= T-start and T-receive < (T-start + T-duration) as illustrated in Figure 3, it means that the request is just in the valid period of time. The server SHOULD process this request immediately, stores a payload indicating "OK" in a normal response for the client and returns this response with Code = 2.05.

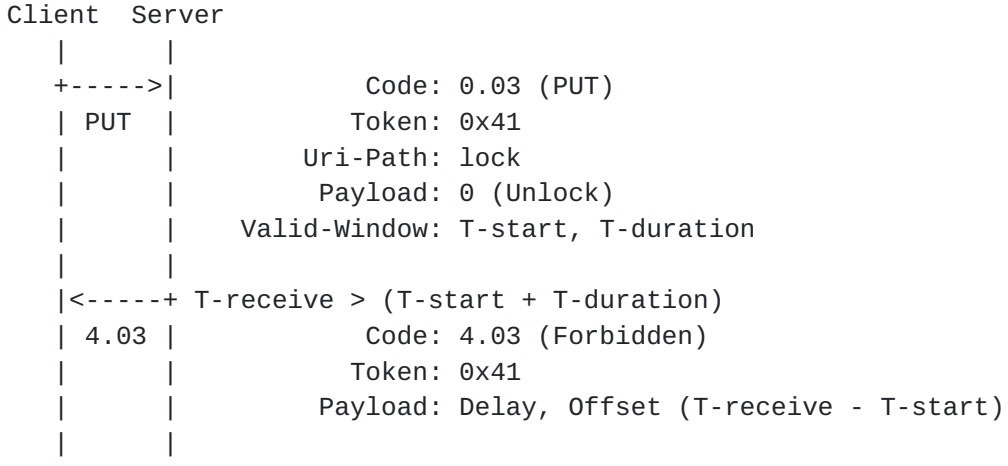

Figure 4: The Time Window Option(3)

If T-receive (i.e., a reception time for the server receiving a request) > (T-start + T-duration) as illustrated in Figure 4, it means that the request has become invalid. The server discards the request and sends a 4.03 error response to the client with a "Delay" payload indicating a time offset between T- receive and T-current. The offset helps the client to estimate the RTT between the client and the server, and thus set a more reasonable T-duration for the subsequent messages.

4.2.2. Multi-interrelated Actions

When some complicated actuators are able to support multi-interrelated actions with different request(s), it is desirable to be required give some indications to the server to make actions especially when there are delay caused by some attacks.

This document proposes the use of a Sequence Number CoAP Option to address the sending sequence of request(s) at the client side. It is used to provide some corresponding rules when the server recognizes that request(s) are disorder via the Sequence Number Options in these messages.

This document also proposes a new Response Policy CoAP Option which is valid with the Sequence Number Option. The Response Policy includes 3 modes - preemptive mode, sequential mode, and sequential with conditional discard mode. Also, the Response Policy may be pre-configured at the server side or may be specified in the requests at the client side. If the server cannot get the Response Policy, the server will select preemptive mode by default.

Upon receiving a request containing the Sequence Number Option, the server will do the following steps:

1) The server is aware that the Sequence Number value in current request (SNcur) is larger than the largest Sequence Number (SNmax) of all previous requests.

a. If the previous request with SNmax has already been normally responded and SNcur = (SNmax + 1) , the current request SHOULD be responded as specified in Section 4.2.1.

b. If the previous request with SNmax is still being queued, the server SHOULD respond the current request with SNcur according to the Response Policy and the validity period of the requests as below:

- Preemptive mode: If T-start of the current request is expired, the server SHOULD process the current request immediately, and then discard all the previous requests in the queue since SNcur > SNmax.

- Sequential mode: The server SHOULD respond to the requests orderly based on their Sequence Numbers. Although T-start of the current request is expired, the server SHOULD not respond to it until all the requests with Sequence Numbers less than SNcur have been responded, even if the request with a sequence number less than the SNcur has not been received by the server. Consequently, there is a possibility that the current request MAY not be

responded due to its Valid Time Window (T-start + T-duration) expiration.

- Sequential with conditional discard mode: The server SHOULD respond to the requests based on their Sequence Numbers as well as the Valid Time Window (T-start + T-duration) of the requests. Once the Valid Time Window of the current request expires, the sever SHOULD respond to the current request immediately, then discard all the requests with Sequence Numbers less than SNcur.

2) The server is aware that the Sequence Number value in current request (SNcur) is smaller than the largest Sequence Number(SNmax) of all previous requests.

- Preemptive mode: If the request with SNmax has already been processed, the server SHOULD discard the current request and respond an error indicating the delay. Otherwise, if the request with SNmax is still being queued, the server SHOULD add the current request to the queue and respond these queued requests in order based on Section 4.2.1 till the T-start of the request with SNmax.

- Sequential mode: The server SHOULD add the current request to the queue till its T-start.

- Sequential with conditional discard mode: The server SHOULD add the current request to the queue till its T-start. If the Valid Time Window (T-start + T-duration) of the SNmax request expires earlier than the T-start of the current request, the server SHOULD process the request with SNmax and discard the current request.

When some complicated actuators are able to support multi-interrelated actions with different requests, it is desirable to be required give some indications to the server to process actions, especially when there are delays caused by attacks.

Note: It is to be added figures to illustrate the above examples in the future.

5. Security Considerations

The whole document can be seen as security considerations for CoAP.

6. IANA Considerations

This document requests the registration of the following Option Number, whose value have been assigned to the CoAP Option Numbers Registry defined by [RFC7252].

6.1. Tables

Number	Name
30	Time Window
31	Sequence Number
32	Response Policy

Table 1

7. Acknowledgements

TBD.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <http://www.rfc-editor.org/info/rfc6347>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <http://www.rfc-editor.org/info/rfc7252>.

[RFC7925] Tschofenig, H. and T. Fossati, "Transport Layer Security(TLS)/ Datagram Transport Layer Security (DTLS) Profiles for the Internet of Things", RFC 7925, DOI 10.17487/RFC6347, December 2016, <http://www.rfc-editor.org/info/rfc7925>.

8.2. Informative References

[I-D.ietf-core-object-security]
Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security of CoAP", draft-ietf-core-object-security-06 (work in progress), October 2017.

[I-D.mattsson-core-coap-actuators]
Mattsson, J., Fornehed, J., Selander, G., and F. Palombini, "Controlling Actuators with CoAP", draft-mattsson-core-copa-actuators-02 (work in progress), November 2016.

Authors' Addresses

Yan Liu
Huawei
P.R.China

Email: scarlett.liuyan@huawei.com

Jintao Zhu
Huawei
P.R.China

Email: jintao.zhu@huawei.com