Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 21, 2016

S. Moore
MITRE Corporation

July 20, 2015

# OAuth 2.0 over Constrained Application Protocol (CoAP) for GET with Observe Requests
## draft-moore-ace-oauth-observe-00

Abstract

This document describes a method for a client or resource server utilizing an OAuth 2.0 [RFC6749] authorization server when responding to a Constrained Application Protocol (CoAP) [RFC7252] GET request with the Observe option [I-D.ietf-core-observe]. CoAP's Observe option has the potential to reduce network traffic by allowing clients to get updates on protected resources as the protected resource's values change rather than polling the protected resource periodically. A client adding the Observe option to a CoAP GET request has the greatest impact on the device providing the protected resource since the resource server (RS) has to maintain a list of requestors and deal with tokens associated with those requests. Additionally an Authorization Server (AS) could potentially allow token introspection to happen over a GET request with the Observe option, further reducing network usage and heavy lifting on the part of the resource server doing the introspection.

Table of Contents



## 1. Introduction

The Constrained Application Protocol (CoAP) [RFC7252] supports a GET request with Observe option [I-D.ietf-core-observe]. When a CoAP client makes such a request the server responds as it normally would as well as with an update every time the requested CoAP resource changes. This behavior eliminates the need to poll the resource along with any additional network overhead such as acknowledgements cutting the traffic down to a third of what otherwise would be required.

[I-D.tschofenig-ace-oauth-iot], [I-D.tschofenig-ace-oauth-bt] describe how to use OAuth in CoAP requests but do not cover the case of GET requests that have the Observe option set. From a client perspective there isn't a change other than setting the Observe option, and handling the various responses to that type of request. From the Resource Server (RS) point of view though, there are addition steps that have to be taken to insure proper secure and authorized access to the resource.

Additionally, [I-D.wahlstroem-ace-oauth-introspection] describes how a RS can perform introspection of a token by sending it to an Authorization Server (AS) over a CoAP POST or GET request. Allowing the GET request to include the Observe option would ease network usage even more by reducing the number of introspections requests an RS would need to perform.

The method described in this document would allow any resource server to respond to a protected resource CoAP GET with Observe request and to perform token introspection over CoAP using the very same GET with Observe option, eliminating the need to validate the every token before sending out updates.

## 2. 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 "Key words for use in RFCs to Indicate Requirement Levels" [RFC2119].

This document also re-uses terminology from RFC 6749 [RFC6749] and RFC 6750 [RFC6750].

## 3. Observable CoAP OAuth Introspection Endpoint

[I-D.wahlstroem-ace-oauth-introspection] defines how a client or resource server can query an OAuth authorization server about an OAuth token using CoAP. For CoAP GETs without the Observe flag and POSTs this method is sufficient as described. Here we define additional behavior for the authorization server to support introspection utilizing CoAP GET with the Observe option (defined in [I-D.ietf-core-observe]).

### 3.1. Authorization Server with Observe Enabled

When an OAuth authorization server (AS) receives a properly formatted introspection request over CoAP GET that has the Observe flag set to 0 (register) and the AS supports Observe, the AS should respond with standard 2.xx response with the observe option set. Upon any change

in the token's state (i.e. expiration, revocation, etc.), the server should send a Notification response to the original requestor. If the token is still valid that response should be comprised of a 2.05 (Content) response code along with the updated meta-data for the token as content.

### 3.2. OAuth token no longer valid

If a token is no longer valid (or if the authentication of the original requestor is no longer valid for introspection), then a notification with the appropriate non-2.xx code should be sent back to the requestor, and the registration for the introspection removed from the authorization server's list of clients.

### 3.3. Authorization Server without Observe Enabled

If the AS does not allow for the introspection endpoint to be observable, or if the AS doesn't support it at all, the AS should behave as defined in [I-D.wahlstroem-ace-oauth-introspection], making sure that responses do not have the Observe flag set.

### 3.4. Unregistering an Introspection Request

When the AS receives a request from a previously registered client that has the Observe flag set to 1 (deregister), that client is indicating that it no longer wants updates about the accompanying token. If the client's credentials are valid, the AS should removed the client from the list of clients receiving updates for the given token. The CoAP particulars of this process are covered by [I-D.ietf-core-observe] 4.1.

## 4. OAuth Protected Resource CoAP GET with Observe

[I-D.tschofenig-ace-oauth-iot] and [I-D.tschofenig-ace-oauth-bt] describe OAuth 2.0 token usage over CoAP. When a resource server (RS) receives a GET request that has the Observe option set the RS should respond appropriately based on the request, and the state of the OAuth token received, as described below.

### 4.1. CoAP GET with Observer usage

When an OAuth protected RS receives a GET request that has the Observe flag set to 0 (register), the RS should first validate the token and then respond appropriately. If the token is valid and the RS supports observability, the RS should return a proper 2.xx response that has the Observe option set, indicating that the RS has added the client to its registered clients list. The RS should also add the client to the registered client list for the requested

resource. Whenever the associated resource changes, the RS should send a 2.05 (Content) response to all the clients in the registered clients list after re-validating the token (or by taking advantage of an Observable Introspection Endpoint as described in Section 3).

If the token is not valid at the time of request or update, then a notification with the appropriate non-2.xx code should be sent back to the requestor, and the registration for the introspection removed from the authorization server's list of clients.

### 4.2. Observer option not supported

If the RS has observable resources disabled, or if the RS doesn't support observable at all, the RS should behave as described in [I-D.tschofenig-ace-oauth-iot] and [I-D.tschofenig-ace-oauth-bt], making sure that responses do not have the Observe flag set indicating that the resource is not observable.

### 4.3. Unregistering an Observer Request

When a client wishes to no longer receive updates from a RS via a GET with Observe request, the client will send a request that has the Observe flag set to 1 (deregister). This request should be accompanied by a valid token. If the token is valid, then the RS should remove the client from the list of registered clients associated with the resource the request is for.

## 5. Security Considerations

TBD

## 6. IANA Considerations

TBD

## 7. Acknowledgments

The author would like to thank William Kim for valuable input, Hannes Tschofenig for his work on [I-D.tschofenig-ace-oauth-iot] and [I-D.tschofenig-ace-oauth-bt], and Erik Wahlstroem for his work on [I-D.wahlstroem-ace-oauth-introspection]. The author would also like to thank Justin Richer for his work on [I-D.richer-oauth-introspection] and general encouragement in the process.

# 8. References

## 8.1. Normative References

[I-D.ietf-core-observe]
Hartke, K., "Observing Resources in CoAP", draft-ietf-core-observe-16 (work in progress), December 2014.

[I-D.tschofenig-ace-oauth-bt]
Tschofenig, H., "The OAuth 2.0 Bearer Token Usage over the Constrained Application Protocol (CoAP)", draft-tschofenig-ace-oauth-bt-01 (work in progress), March 2015.

[I-D.tschofenig-ace-oauth-iot]
Tschofenig, H., "The OAuth 2.0 Internet of Things (IoT) Client Credentials Grant", draft-tschofenig-ace-oauth-iot-01 (work in progress), March 2015.

[I-D.wahlstroem-ace-oauth-introspection]
Wahlstroem, E., "OAuth 2.0 Introspection over the Constrained Application Protocol (CoAP)", draft-wahlstroem-ace-oauth-introspection-01 (work in progress), March 2015.

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

[RFC6749] Hardt, D., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012.

[RFC6750] Jones, M. and D. Hardt, "The OAuth 2.0 Authorization Framework: Bearer Token Usage", RFC 6750, October 2012.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, June 2014.

## 8.2. Informative References

[I-D.richer-oauth-introspection]
Richer, J., "OAuth Token Introspection", draft-richer-oauth-introspection-06 (work in progress), July 2014.

Author's Address

Stephen R Moore
MITRE Corporation
202 Burlington Rd.
Bedford, MA 01730

Email: srmoore@gmail.com