Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 5, 2015

H. Tschofenig
ARM Limited

July 4, 2014

The OAuth 2.0 Bearer Token Usage over the Constrained Application Protocol (CoAP)
draft-tschofenig-ace-oauth-bt-00.txt

Abstract

This specification describes how to use OAuth 2.0 bearer tokens to access protected resources using the Constrained Application Protocol (CoAP). Any party in possession of a bearer token (a "bearer") can use it to get access to the associated resources (without demonstrating possession of a cryptographic key). To prevent misuse, bearer tokens need to be protected from disclosure in storage and in transport.

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 5, 2015.

Copyright Notice

Table of Contents

1. Introduction 2
2. Terminology 2
3. Introduction 3
4. Requests 3
5. Responses 4
6. Security Considerations 6
7. IANA Considerations 6
8. Acknowledgements 7
9. Normative References 7
Author's Address 7

1. Introduction

OAuth enables clients to access protected resources by obtaining an access token, which is defined in "The OAuth 2.0 Authorization Framework" [4] as "a string representing an access authorization issued to the client", rather than using the resource owner's credentials directly.

Tokens are issued to clients by an authorization server and the client uses the access token to access the protected resources hosted by the resource server. This specification describes how to make protected resource requests when the access token is a bearer token and conveyed from the client to the resource server using the Constrained Application Protocol (CoAP) [3]. To secure the communication exchange the Datagram Transport Layer Security Version 1.2 [1] is mandatory to use.

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" [5].

This document also re-uses terminology from RFC 6749 [4] and RFC 6750 [2].

3. Introduction

The abstract OAuth 2.0 flow illustrated in Figure 1 describes the interaction between the client, resource owner, authorization server, and resource server (described in [[4]). The following two steps are specified within this document:

(E) The client requests the protected resource from the resource server and authenticates by presenting the access token.

(F) The resource server validates the access token, and if valid, serves the request.

This document also imposes semantic requirements upon the access token returned in step (D).

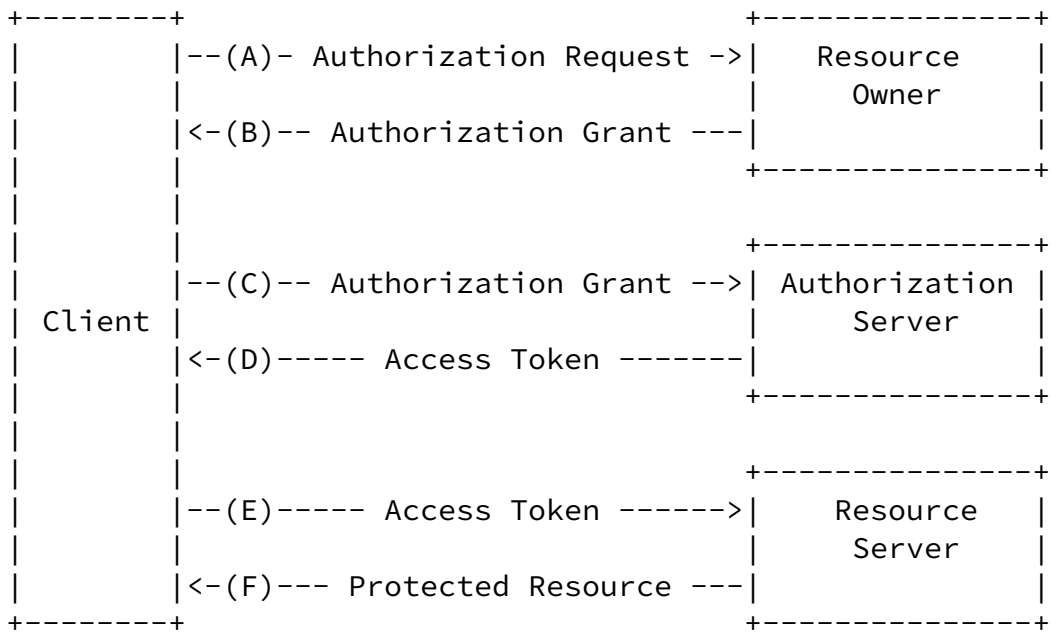

Figure 1: Abstract OAuth 2.0 Protocol Flow.

4. Requests

Access tokens are embedded in a CoAP message by the use of the "Bearer" option. The definition is shown in Figure 2.

No.	C	U	N	R	Name	Format	Length	Default
TBD	x				Bearer	opaque	var	(none)

Legend:
C=Critical, U=Unsafe, N=NoCacheKey, R=Repeatable

Figure 2: Bearer Option Definition.

Figure 3 shows an example request from the client to the resource server.

```
Header: GET (T=CON, Code=0.01, MID=0x7d34,
            Uri-Path:"resource", Bearer: "mF_9.B5f-4.1JqM")
```

Figure 3: Example CoAP Request with Bearer Token.

5. Responses

If the request does not contain an access token that enables access to the protected resource, the resource server MUST respond with a 4.01 "Unauthorized" error message.

QUESTION: Should the response also provide information about the

scheme (e.g., 'Bearer Tokens' vs. 'Proof-of-Possession Tokens')? Should it contain a "realm" attribute as well? Should a scope value be returned to provide some guidance about the available scopes at that resource server?

For example, in response to a protected resource request without a needed bearer token the error response shown in Figure 4 is sent.

Header: ACK (T=ACK, Code=4.01, MID=0x7d34)

Figure 4: Failed Request due to Missing Access Token.

To provide information back to the client about the failure of the request the following error codes are defined. These error codes are conveyed within the 'Error' option, which is defined in Figure 5.

No.	C	U	N	R	Name	Format	Length	Default
TBD					Error	uint	0-2	(none)

Legend:
C=Critical, U=Unsafe, N=NoCacheKey, R=Repeatable

Figure 5: Error Option Definition.

This specification defines the following error codes that are used with the 'Error' option:

invalid_request (0)

The request is missing a required parameter, includes an unsupported parameter or parameter value, repeats the same parameter, uses more than one method for including an access token, or is otherwise malformed. The resource server MUST respond with the 4.00 (Bad Request) status code.

invalid_token (1)

The access token provided is expired, revoked, malformed, or invalid for other reasons. The resource MUST respond with the 4.01 (Unauthorized) status code. The client MAY request a new access token and retry the protected resource request.

insufficient_scope (2)

The request requires higher privileges than provided by the access token. The resource server MUST respond with the 4.03 (Forbidden) status code.

QUESTION: Is the granularity of the error messages useful enough for client implementations to take actions?

As an example, in response to a request using an expired access token the following error is returned.

```
Header: ACK (T=ACK, Code=4.01, MID=0x7d34,
             Error="1")
```

Figure 6: Failed Request due Expired Access Token.

6. Security Considerations

The security threats of this specification are identical to those discussed in RFC 6750 since the encoding of the request does not change the security threats.

It is nevertheless worthwhile to replicate the security recommendation here for readers who do not want to consult another document.

Safeguard bearer tokens: Client implementations MUST ensure that bearer tokens are not leaked to unintended parties, as they will be able to use them to gain access to protected resources. This is the primary security consideration when using bearer tokens and underlies all the more specific recommendations that follow.

Validate TLS certificate chains: The client MUST validate the certificate chain, if the resource server is authenticated using a certificate-based ciphersuite in DTLS, when making requests to protected resources. Failing to do so may enable man-in-the-middle attacks.

Always use DTLS (coaps): Clients MUST always use DTLS [1] when making requests with bearer tokens. Failing to do so exposes the token to third parties and could consequently give attackers unintended access.

Issue short-lived bearer tokens: Authorization servers SHOULD issue short-lived bearer tokens. Using short-lived bearer tokens reduces the impact of them being leaked.

Issue scoped bearer tokens: Authorization servers MUST issue bearer tokens that contain an audience restriction, scoping their use to the intended relying party or set of relying parties.

7. IANA Considerations

This specification requests IANA to allocate two values, as shown below, in the 0..255 range of the CoAP option number registry established with RFC 7252.

Number	Name	Reference
TBD-61	Bearer	[this RFC]
TBD-62	Error	[this RFC]

TBD: Add a registry for error codes.

8. Acknowledgements

The author would like to thank Mike Jones and Dick Hardt for their work on RFC 6750. This document is heavily inspired by their work.

9. Normative References

[1] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.

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

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

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

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

Author's Address

Hannes Tschofenig
ARM Limited
Austria

Email: Hannes.Tschofenig@gmx.net
URI: http://www.tschofenig.priv.at