B. Campbell, Ed.
Ping Identity Corp.
C. Mortimore
Salesforce.com

Internet-Draft
Intended status: Standards Track
Expires: January 28, 2011

July 27, 2010

SAML 2.0 Bearer Assertion Profile for OAuth 2.0
draft-campbell-oauth-saml-00

Abstract

This specification defines the use of a SAML 2.0 bearer assertion as means for requesting an OAuth 2.0 access token.

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 28, 2011.

Table of Contents

1. Introduction

The Security Assertion Markup Language (SAML) 2.0 [OASIS.saml-core-2.0-os], is an XML-based framework that provides a means for a subject to be identified across security domains. The SAML specification, while primarily targeted at providing cross domain web browser single sign-on, was also designed to be modular and extensible to facilitate use in other contexts. The Assertion, an XML security token, is a fundamental construct of SAML that is most often adopted for use in other protocols and specifications. An assertion is generally issued by an identity provider and consumed by a service provider who relies on its content to identify the subject for security related purposes.

OAuth 2.0 [I-D.ietf.oauth-v2] provides a method for making authenticated HTTP requests to a resource using an access token. Tokens are issued to third-party clients by an authorization server with the (sometimes implicit) approval of the resource owner. OAuth defines multiple profiles for obtaining access tokens to support a wide range of client types and user experiences. One such method is the use of an assertion which supports the case when a client wishes to exchange an existing security token for an access token. However the Oauth 2.0 leaves the specific format and validation of the assertion out of scope.

This specification profiles the specific use of a SAML 2.0 bearer assertion in requesting an access token using the assertion grant_type from OAuth 2.0. The format and processing rules for the SAML assertion defined in this specification are intentionally similar to those in the Web Browser SSO Profile defined in [OASIS.saml-profiles-2.0-os] with the goal of reusing, to the extent reasonable, concepts and patterns from that well established profile.

1.1. Notational Conventions

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].

Unless otherwise noted, all the protocol parameter names and values are case sensitive.

2. SAML Assertion Access Token Request

A SAML assertion is used to request an access token when a client wishes to utilize an existing trust relationship, expressed though the semantics of the SAML assertion, to establish authorization

without directly involving the resource owner's approval at the resource server.

The process by which the client obtains the assertion is out of scope.

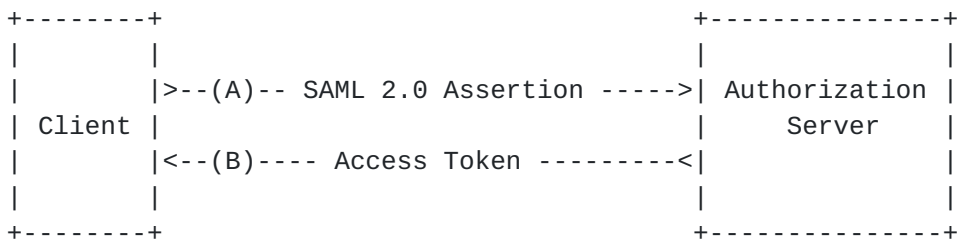

Figure 1: Assertion Access Token Request

The request/response flow illustrated in Figure 1 includes the following steps:

(A) The client sends an access token request to the authorization server and includes a SAML 2.0 assertion.

(B) The authorization server validates the assertion per the processing rules defined in this specification and issues an access token.

2.1. Client Requests Access Token

The client requests an access token by making an HTTP "POST" request to the token endpoint using an assertion as an access grant. The client makes an access token request, as defined in the OAuth, with the following parameter definitions taking precedence in the constructed URI:

assertion_type
: REQUIRED. The value of the assertion_format parameter MUST be "http://oauth.net/assertion_type/saml/2.0/bearer"

assertion
: REQUIRED. The value of the assertion parameter MUST contain a single SAML 2.0 Assertion. The SAML assertion XML data MUST be encoded using base64url, where the encoding adheres to the definition in Section 5 of RFC4648 [RFC4648] and where the padding bits set to zero. To to avoid the need for subsequent encoding steps (by "application/x-www-form-urlencoded"

[W3C.REC-html401-19991224], for example), the base64url encoded data SHOULD NOT be line wrapped and pad characters ("=") SHOULD NOT be included.

2.2. Assertion Format and Processing Requirements

The authorization server MUST validate the assertion according to the criteria below and, if valid, issues an access token response as described in [I-D.ietf.oauth-v2]. The access token SHOULD be issued only for the subject of the assertion

- The Assertion's <Issuer> element MUST contain a unique identifier for the entity that issued the assertion; the Format attribute MUST be omitted or have a value of "urn:oasis:names:tc:SAML:2.0:nameid-format:entity".

- The assertion MUST contain a <Subject> element that identifies the resource owner for whom the access token is being requested.

- The <Subject> element MUST contain a single <SubjectConfirmation> element and it MUST have a Method attribute with a value of "urn:oasis:names:tc:SAML:2.0:cm:bearer".

- The <SubjectConfirmation> element MUST contain a single <SubjectConfirmationData> element.

- The <SubjectConfirmationData> element MUST have a Recipient attribute with a value indicating the token endpoint URL of the authorization server. The authorization server MUST verify that the value of the Recipient attribute matches the token endpoint URL (or an acceptable alias) to which the assertion was delivered.

- The <SubjectConfirmationData> element MUST have a NotOnOrAfter attribute that limits the window during which the assertion can be confirmed. The authorization server MUST verify that the NotOnOrAfter instant has not passed, subject to allowable clock skew between systems.

- The <SubjectConfirmationData> element MUST NOT contain a NotBefore attribute.

- The <SubjectConfirmationData> element MAY also contain an Address attribute limiting the client address from which the assertion can be delivered. Verification of the Address is at the discretion of the authorization server.

- If the assertion issuer authenticated the subject, the assertion SHOULD contain a single <AuthnStatement> representing that

authentication event.

- If the assertion was issued with the intention that the client act autonomously on behalf of the subject, an <AuthnStatement> SHOULD NOT be included.

- Other statements, in particular, <AttributeStatement> elements MAY be included in the assertion.

- The assertion MUST contain an <AudienceRestriction> element with an <Audience> element containing a URI reference that identifies the authorization server, or the service provider SAML entity of its controlling domain, as an intended audience. The authorization server MUST verify that it is an intended audience for the assertion.

- The authorization server MAY ensure that bearer assertions are not replayed, by maintaining the set of used ID values for the length of time for which the assertion would be considered valid based on the NotOnOrAfter attribute in the <SubjectConfirmationData>.

- The assertion MUST be digitally signed by the issuer and the authorization server MUST verify the signature.

- Encrypted elements MAY appear in place of their plain text counterparts as defined in [OASIS.saml-core-2.0-os].

- The authorization server MUST verify that the assertion is valid in all other respects per [OASIS.saml-core-2.0-os].

2.3. Error Response

If the assertion is not valid, or its subject confirmation requirements cannot be met, the the authorization server MUST construct an error response as defined in [I-D.ietf.oauth-v2]. The value of the error parameter MUST be the "invalid_grant" error code. The authorization server MAY include additional information regarding the reasons the assertion was considered invalid using the error_description or error_uri parameters.

For example:

```
HTTP/1.1 400 Bad Request
Content-Type: application/json
Cache-Control: no-store

{
  "error":"invalid_grant",
  "error_description":"invalid signature"
}
```

2.4. Example (non-normative)

Though non-normative, the following examples illustrate what a conformant assertion and access token request would look like.

Below is an example SAML 2.0 Assertion (whitespace formatting is for display purposes only):

```
<Assertion IssueInstant="1970-01-01T00:00:00.000Z"
  ID="ef1xsbZxPV2oqjd7HTLRLIBlBb7"
  Version="2.0"
  xmlns="urn:oasis:names:tc:SAML:2.0:assertion">
 <Issuer>https://saml-idp.example.com</Issuer>
 <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  [...omitted for brevity...]
 </ds:Signature>
 <Subject>
  <NameID
    Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress">
   brian@example.com
  </NameID>
  <SubjectConfirmation
    Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
   <SubjectConfirmationData
    NotOnOrAfter="1970-01-01T00:05:00.000Z"
    Recipient="https://authz.example.net/token.oauth2"/>
   </SubjectConfirmation>
 </Subject>
 <Conditions>
  <AudienceRestriction>
   <Audience>https://saml-sp.example.net</Audience>
  </AudienceRestriction>
 </Conditions>
</Assertion>
```

Figure 2: Example SAML 2.0 Assertion

To present the assertion shown in the previous example as part of an access token request, for example, the client makes the following HTTPS request (line breaks are for display purposes only):

```
POST /token.oauth2 HTTP/1.1
Host: authz.example.net
Content-Type: application/x-www-form-urlencoded

grant_type=assertion&assertion_type=http%3A%2F%2Foauth.net%2Fasse
rtion_type%2Fsaml%2F2.0%2Fbearer&assertion=PEFzc2VydGlvbiBJc3N1ZU
[...omitted for brevity...]b24-PC9Db25kaXRpb25zPjwvQXNzZXJ0aW9uPg
```

Figure 3: Example Request

3. Security Considerations

No additional considerations beyond those described within the OAuth 2.0 Protocol [I-D.ietf.oauth-v2] and in the Security and Privacy Considerations for the OASIS Security Assertion Markup Language (SAML) V2.0 [OASIS.saml-sec-consider-2.0-os].

4. IANA Considerations

This document has no actions for IANA.

Appendix A. Contributors

The following people contributed wording and concepts to this document: Patrick Harding, Peter Motyka, Peter Saint-Andre and David Waite

5. References

5.1. Normative References

[I-D.ietf.oauth-v2]
Hammer-Lahav, E., Ed., Recordon, D., and D. Hardt, "The OAuth 2.0 Protocol", Jun 2010.

[OASIS.saml-core-2.0-os]
Cantor, S., Kemp, J., Philpott, R., and E. Maler,

"Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0", OASIS Standard saml-core-2.0-os, March 2005.

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

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006.

5.2. Informative References

[OASIS.saml-profiles-2.0-os]
Hughes, J., Cantor, S., Hodges, J., Hirsch, F., Mishra, P., Philpott, R., and E. Maler, "Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0", OASIS Standard OASIS.saml-profiles-2.0-os, March 2005.

[OASIS.saml-sec-consider-2.0-os]
Hirsch, F., Philpott, R., and E. Maler, "Security and Privacy Considerations for the OASIS Security Markup Language (SAML) V2.0", OASIS Standard saml-sec-consider-2.0-os, March 2005.

[W3C.REC-html401-19991224]
Hors, A., Raggett, D., and I. Jacobs, "HTML 4.01 Specification", World Wide Web Consortium Recommendation REC-html401-19991224, December 1999, <http://www.w3.org/TR/1999/REC-html401-19991224>.

Authors' Addresses

Brian Campbell (editor)
Ping Identity Corp.

Email: brian.d.campbell@gmail.com

Chuck Mortimore
Salesforce.com

Email: cmortimore@salesforce.com