Network Working Group	B. Campbell, Ed.
Internet-Draft	Ping Identity Corp.
Intended status: Standards Track	C. Mortimore
	Salesforce.com
	Aug 2011

SAML 2.0 Bearer Assertion Profiles for OAuth 2.0
draft-ietf-oauth-saml2-bearer-07

Abstract

This specification defines the use of a SAML 2.0 Bearer Assertion as means for requesting an OAuth 2.0 access token as well as for use as a means of client authentication.

Status of this Memo

Copyright Notice

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 allows identity and security information to be shared 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 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 Assertion's subject for security related purposes.
The OAuth 2.0 Authorization Protocol *[I-D.ietf.oauth-v2]* provides a method for making authenticated HTTP requests to a resource using an access token. Access tokens are issued to third-party clients by an authorization server (AS) with the (sometimes implicit) approval of the resource owner. In OAuth, an authorization grant is an abstract term used to describe intermediate credentials that represent the resource owner authorization. An authorization grant is used by the client to obtain an access token. Several authorization grant types are defined to support a wide range of client types and user experiences. OAuth

also allows for the definition of new extension grant types to support additional clients or to provide a bridge between OAuth and other trust frameworks. Finally, OAuth allows the definition of additional authentication mechanisms to be used by clients when interacting with the authorization server.

The OAuth 2.0 Assertion Profile *[I-D.ietf.oauth-assertions]* is an abstract extension to OAuth 2.0 that provides a general framework for the use of assertions as client credentials and/or authorization grants with OAuth 2.0. This specification profiles the OAuth 2.0 Assertion Profile *[I-D.ietf.oauth-assertions]* to define an extension grant type that usues a SAML 2.0 Bearer Assertion to request an OAuth 2.0 access token as well as for use as client credentials. The format and processing rules for the SAML Assertion defined in this specification are intentionally similar, though not identical, to those in the Web Browser SSO Profile defined in SAML Profiles *[OASIS.saml-profiles-2.0-os]*. This specification is reusing, to the extent reasonable, concepts and patterns from that well-established Profile.

This document defines how a SAML Assertion can be used to request an access token when a client wishes to utilize an existing trust relationship, expressed through the semantics of (and digital signature calculated over) the SAML Assertion, without a direct user approval step at the authorization server. It also defines how a SAML Assertion can be used as a client authentication mechanism. The use of an Assertion for client authentication is orthogonal and separable from using an Assertion as an authorization grant and can be used either in combination or in isolation.

The process by which the client obtains the SAML Assertion, prior to exchanging it with the authorization server or using it for client authentication, is out of scope.

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. HTTP Parameter Bindings for Transporting Assertions

The OAuth 2.0 Assertion Profile *[I-D.ietf.oauth-assertions]* defines generic HTTP parameters for transporting assertions during interactions with a token endpoint. This section defines the values of those parameters for use with SAML 2.0 Bearer Assertions.

2.1. Using SAML Assertions as Authorization Grants

To use a SAML Bearer Assertion as an authorization grant, use the following paramter values and encodings.

The value of "grant_type" parameter MUST be "urn:ietf:params:oauth:grant-type:saml2-bearer"
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 are set to zero. 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. Using SAML Assertions for Client Authentication

To use a SAML Bearer Assertion for client authentication grant, use the following paramter values and encodings.
The value of "client_assertion_type" parameter MUST be "urn:ietf:params:oauth:client-assertion-type:saml2-bearer"
The value of the "client_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 are set to zero. 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.

3. Assertion Format and Processing Requirements

In order to issue an access token response as described in The OAuth 2.0 Authorization Protocol *[I-D.ietf.oauth-v2]* or to rely on an assertion for client authentication, the authorization server MUST validate the Assertion according to the criteria below. Application of additional restrictions and policy are at the discretion of the authorization server.

* The Assertion's <Issuer> element MUST contain a unique identifier for the entity that issued 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 token endpoint URL of the authorization server MAY be used as an acceptable value for an <Audience> element. The authorization server MUST verify that it is an intended audience for the Assertion.

* The Assertion MUST contain a <Subject> element. The subject MAY identify the resource owner for whom the access token is being

requested. For client authentication, the Subject MUST be the client_id of the OAuth client. When using assertions as an authorization grant, the Subject SHOULD identify an authorized accessor for whom the access token is being requested (typically the resource owner, or an authorized delegate). Additional information identifying the subject/principal of the transaction MAY be included in an <AttributeStatement>.

*The Assertion MUST have an expiry that limits the time window during which the it can be used. The expiry can be expressed either as the NotOnOrAfter attribute of the <Conditions> element or as the NotOnOrAfter attribute of a suitable <SubjectConfirmationData> element.

If the Assertion has a NotOnOrAfter attribute on the <Conditions> element, the authorization server MUST verify that the NotOnOrAfter instant has not passed, subject to allowable clock skew between systems. The authorization server SHOULD reject assertions with an expiry instant that is unreasonably far in the future.

If the Assertion does not have a NotOnOrAfter attribute on the <Conditions> element, then the Assertion's <Subject> element MUST contain at least one <SubjectConfirmation> element that allows the authorization server to confirm it as a Bearer Assertion. Conditions for bearer subject confirmation are described below.

- The <SubjectConfirmation< MUST have a Method attribute with a value of "urn:oasis:names:tc:SAML:2.0:cm:bearer" and MUST contain a <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 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 authorization server MAY reject assertions with a NotOnOrAfter instant that is unreasonably far in the future.

- 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 presenter act autonomously on behalf of the subject, an <AuthnStatement> SHOULD NOT be included. The presenter SHOULD be identified in the <NameID> or similar element, the <SubjectConfirmation> element, or by other available means like [OASIS.saml-deleg-cs].

* Other statements, in particular <AttributeStatement> elements, MAY be included in the Assertion.

* 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], such as (but not limited to) evaluating all content within the Conditions element including the NotOnOrAfter and NotBefore attributes, rejecting unknown condition types, etc.

3.1. Authorization Grant Processing

If present, the authorization server MUST also validate the client credentials.
Authorization servers SHOULD issue access tokens with a limited lifetime and require clients to refresh them by requesting a new access token using the same assertion, if it is still valid, or with a new assertion. The authorization server SHOULD NOT issue a refresh token.
If the Assertion is not valid, or its subject confirmation requirements cannot be met, 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":"Audience validation failed"
}
```

3.2. Client Authentication Processing

If the client Assertion is not valid, or its subject confirmation requirements cannot be met, 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_client" 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.

4. Authorization Grant Example (non-normative)

Though non-normative, the following examples illustrate what a conforming 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="2010-10-01T20:07:34.619Z"
  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="2010-10-01T20:12:34.619Z"
     Recipient="https://authz.example.net/token.oauth2"/>
   </SubjectConfirmation>
  </Subject>
  <Conditions>
    <AudienceRestriction>
      <Audience>https://saml-sp.example.net</Audience>
    </AudienceRestriction>
  </Conditions>
  <AuthnStatement AuthnInstant="2010-10-01T20:07:34.371Z">
    <AuthnContext>
      <AuthnContextClassRef>
        urn:oasis:names:tc:SAML:2.0:ac:classes:X509
      </AuthnContextClassRef>
    </AuthnContext>
  </AuthnStatement>
</Assertion>
```

To present the Assertion shown in the previous example as part of an access token request, for example, the client might make 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=urn%3Aietf%3Aparams%3Aoauth%3Agrant-type%3Asaml2-
bearer&assertion=PEFzc2VydGlvbiBJc3N1ZUluc3RhbnQ9IjIwMTEtMDU
[...omitted for brevity...]aG5TdGF0ZW1lbnQ-PC9Bc3NlcnRpb24-
```

5. Security Considerations

No additional considerations beyond those described within the OAuth 2.0 Protocol Framework *[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]*.

6. IANA Considerations

6.1. Sub-Namespace Registration of urn:ietf:params:oauth:grant-type:saml2-bearer

This is a request to IANA to please register the value grant-type:saml2-bearer in the registry urn:ietf:params:oauth established in [I-D.ietf.oauth-urn-sub-ns]

*URN: urn:ietf:params:oauth:grant-type:saml2-bearer

*Common Name: SAML 2.0 Bearer Assertion Grant Type Profile for OAuth 2.0

*Change controller: IETF

*Description: [[this document]]

6.2. Sub-Namespace Registration of urn:ietf:params:oauth:client-assertion-type:saml2-bearer

This is a request to IANA to please register the value client-assertion-type:saml2-bearer in the registry urn:ietf:params:oauth established in [I-D.ietf.oauth-urn-sub-ns]

*URN: urn:ietf:params:oauth:client-assertion-type:saml2-bearer

*Common Name: SAML 2.0 Bearer Assertion Profile for OAuth 2.0 Client Authentication

*Change controller: IETF

*Description: [[this document]]

7. References

7.1. Normative References

[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.

[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.
[I-D.ietf.oauth-assertions]	Mortimore, C., Campbell, B., Jones, M. and Y. Goland, "OAuth 2.0 Assertion Profile", ID draft-ietf-oauth-assertions-00 (work in progress), July 2011.
[I-D.ietf.oauth-urn-sub-ns]	Campbell, B. and H. Tschofenig, "An IETF URN Sub-Namespace for OAuth", ID draft-ietf-oauth-urn-sub-ns-00 (work in progress), Aug 2011.
[I-D.ietf.oauth-v2]	Hammer-Lahav, E., Recordon, D. and D. Hardt, "The OAuth 2.0 Authorization Protocol", ID draft-ietf-oauth-v2-16 (work in progress), May 2011.

7.2. Informative References

[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]	Raggett, D., Hors, A. and I. Jacobs, "HTML 4.01 Specification", World Wide Web Consortium Recommendation REC-html401-19991224, December 1999.
[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-deleg-cs]	Cantor, S., "SAML V2.0 Condition for Delegation Restriction", Nov 2009.

Authors' Addresses

Brian Campbell editor Campbell Ping Identity Corp. EMail: brian.d.campbell@gmail.com

Chuck Mortimore Mortimore Salesforce.com EMail: cmortimore@salesforce.com