Security Assertion Markup Language (SAML) 2.0 Profile
for OAuth 2.0 Client Authentication and Authorization Grants

Abstract

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

Status of This Memo

   This is an Internet Standards Track document.

   This document is a product of the Internet Engineering Task Force
   (IETF). It represents the consensus of the IETF community. It has
   received public review and has been approved for publication by the
   Internet Engineering Steering Group (IESG). Further information on
   Internet Standards is available in Section 2 of RFC 5741.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   http://www.rfc-editor.org/info/rfc7522.

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 (SSO), 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. (Some examples include [OASIS.WSS-SAMLTokenProfile]
   and [OASIS.WS-Fed].) An Assertion is generally issued by an Identity
   Provider and consumed by a Service Provider that relies on its
   content to identify the Assertion's subject for security-related
   purposes.

   The OAuth 2.0 Authorization Framework [RFC6749] 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

Campbell, et al.   Standards Track   [Page 2]

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.

"Assertion Framework for OAuth 2.0 Client Authentication and
Authorization Grants" [RFC7521] 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 Assertion Framework [RFC7521] to
define an extension grant type that uses 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 the
SAML Profiles [OASIS.saml-profiles-2.0-os] specification. 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 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 to and separable from using an Assertion
as an authorization grant. They can be used either in combination or
separately. Client assertion authentication is nothing more than an
alternative way for a client to authenticate to the token endpoint,
and it must be used in conjunction with some grant type to form a
complete and meaningful protocol request. Assertion authorization
grants may be used with or without client authentication or
identification. Whether or not client authentication is needed in
conjunction with an assertion authorization grant, as well as the
supported types of client authentication, are policy decisions at the
discretion of the authorization server.

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.

1.2. Terminology

   All terms are as defined in the following specifications: "The OAuth
   2.0 Authorization Framework" [RFC6749], the OAuth Assertion Framework
   [RFC7521], and "Assertions and Protocols for the OASIS Security
   Assertion Markup Language (SAML) V2.0" [OASIS.saml-core-2.0-os].

2. HTTP Parameter Bindings for Transporting Assertions

   The OAuth Assertion Framework [RFC7521] defines generic HTTP
   parameters for transporting assertions during interactions with a
   token endpoint. This section defines specific parameters and
   treatments 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, the client
   uses an access token request as defined in Section 4 of the OAuth
   Assertion Framework [RFC7521] with the following specific parameter
   values and encodings.

   The value of the "grant_type" parameter is
   "urn:ietf:params:oauth:grant-type:saml2-bearer".

   The value of the "assertion" parameter contains a single SAML 2.0
   Assertion. It MUST NOT contain more than one 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 RFC 4648
   [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 MUST NOT be line wrapped and pad characters ("=") MUST
   NOT be included.

   The "scope" parameter may be used, as defined in the OAuth Assertion
   Framework [RFC7521], to indicate the requested scope.

   Authentication of the client is optional, as described in
   Section 3.2.1 of OAuth 2.0 [RFC6749] and consequently, the
   "client_id" is only needed when a form of client authentication that
   relies on the parameter is used.

   The following example demonstrates an access token request with an
   Assertion as an authorization grant (with extra line breaks for
   display purposes only):

     POST /token.oauth2 HTTP/1.1
     Host: as.example.com
     Content-Type: application/x-www-form-urlencoded

     grant_type=urn%3Aietf%3Aparams%3Aoauth%3Agrant-type%3Asaml2-bearer&
     assertion=PHNhbWxwOl...[omitted for brevity]...ZT4

2.2. Using SAML Assertions for Client Authentication

   To use a SAML Bearer Assertion for client authentication, the client
   uses the following parameter values and encodings.

   The value of the "client_assertion_type" parameter is
   "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 RFC 4648 [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.

   The following example demonstrates a client authenticating using an
   Assertion during the presentation of an authorization code grant in
   an access token request (with extra line breaks for display purposes
   only):

     POST /token.oauth2 HTTP/1.1
     Host: as.example.com
     Content-Type: application/x-www-form-urlencoded

     grant_type=authorization_code&
     code=n0esc3NRze7LTCu7iYzS6a5acc3f0ogp4&
     client_assertion_type=urn%3Aietf%3Aparams%3Aoauth
     %3Aclient-assertion-type%3Asaml2-bearer&
     client_assertion=PHNhbW...[omitted for brevity]...ZT

3. Assertion Format and Processing Requirements

   In order to issue an access token response as described in OAuth 2.0
   [RFC6749] 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.

   1.   The Assertion's <Issuer> element MUST contain a unique
        identifier for the entity that issued the Assertion. In the
        absence of an application profile specifying otherwise,
        compliant applications MUST compare Issuer values using the
        Simple String Comparison method defined in Section 6.2.1 of RFC
        3986 [RFC3986].

   2.   The Assertion MUST contain a <Conditions> element with an
        <AudienceRestriction> element with an <Audience> element that
        identifies the authorization server as an intended audience.
        Section 2.5.1.4 of "Assertions and Protocols for the OASIS
        Security Assertion Markup Language (SAML) V2.0"
        [OASIS.saml-core-2.0-os] defines the <AudienceRestriction> and
        <Audience> elements and, in addition to the URI references
        discussed there, the token endpoint URL of the authorization
        server MAY be used as a URI that identifies the authorization
        server as an intended audience. The authorization server MUST
        reject any Assertion that does not contain its own identity as
        the intended audience. In the absence of an application profile
        specifying otherwise, compliant applications MUST compare the
        Audience values using the Simple String Comparison method
        defined in Section 6.2.1 of RFC 3986 [RFC3986]. As noted in
        Section 5, the precise strings to be used as the Audience for a
        given authorization server must be configured out of band by the
        authorization server and the issuer of the Assertion.

   3.   The Assertion MUST contain a <Subject> element identifying the
        principal that is the subject of the Assertion. Additional
        information identifying the subject/principal MAY be included in
        an <AttributeStatement>.

        A. For the authorization grant, the Subject typically
            identifies an authorized accessor for which the access token
            is being requested (i.e., the resource owner or an
            authorized delegate), but in some cases, it may be a
            pseudonymous identifier or other value denoting an anonymous
            user.

        B. For client authentication, the Subject MUST be the
            "client_id" of the OAuth client.

4. The Assertion MUST have an expiry that limits the time window
    during which 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.

5. The <Subject> element MUST contain at least one
    <SubjectConfirmation> element that has a Method attribute with a
    value of "urn:oasis:names:tc:SAML:2.0:cm:bearer". If the
    Assertion does not have a suitable NotOnOrAfter attribute on the
    <Conditions> element, the <SubjectConfirmation> element MUST
    contain a <SubjectConfirmationData> element. When present, the
    <SubjectConfirmationData> element MUST have a Recipient
    attribute with a value indicating the token endpoint URL of the
    authorization server (or an acceptable alias). 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 <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.

6. The authorization server MUST reject the entire Assertion if the
    NotOnOrAfter instant on the <Conditions> element has passed
    (subject to allowable clock skew between systems). The
    authorization server MUST reject the <SubjectConfirmation> (but
    MAY still use the rest of the Assertion) if the NotOnOrAfter
    instant on the <SubjectConfirmationData> has passed (subject to
    allowable clock skew). Note that the authorization server may
    reject Assertions with a NotOnOrAfter instant that is
    unreasonably far in the future. 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 applicable
    NotOnOrAfter instant.

7. If the Assertion issuer directly 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 and the
    client presenting the Assertion SHOULD be identified in the
    <NameID> or similar element in the <SubjectConfirmation>
    element, or by other available means like "SAML V2.0 Condition
    for Delegation Restriction" [OASIS.saml-deleg-cs].

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

   9.   The Assertion MUST be digitally signed or have a Message
         Authentication Code (MAC) applied by the issuer. The
         authorization server MUST reject Assertions with an invalid
         signature or MAC.

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

   11.   The authorization server MUST reject an Assertion that is not
         valid in all other respects per [OASIS.saml-core-2.0-os], such
         as (but not limited to) all content within the Conditions
         element including the NotOnOrAfter and NotBefore attributes,
         unknown condition types, etc.

3.1. Authorization Grant Processing

   Assertion authorization grants may be used with or without client
   authentication or identification. Whether or not client
   authentication is needed in conjunction with an Assertion
   authorization grant, as well as the supported types of client
   authentication, are policy decisions at the discretion of the
   authorization server. However, if client credentials are present in
   the request, the authorization server MUST validate them.

   If the Assertion is not valid (including if its subject confirmation
   requirements cannot be met), the authorization server constructs an
   error response as defined in OAuth 2.0 [RFC6749]. 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 (including if its subject
   confirmation requirements cannot be met), the authorization server
   constructs an error response as defined in OAuth 2.0 [RFC6749]. 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

   The following examples illustrate what a conforming Assertion and an
   access token request would look like.

   The example shows an assertion issued and signed by the SAML Identity
   Provider identified as "https://saml-idp.example.com". The subject
   of the Assertion is identified by email address as
   "brian@example.com", who authenticated to the Identity Provider by
   means of a digital signature where the key was validated as part of
   an X.509 Public Key Infrastructure. The intended audience of the
   Assertion is "https://saml-sp.example.net", which is an identifier
   for a SAML Service Provider with which the authorization server
   identifies itself. The Assertion is sent as part of an access token
   request to the authorization server's token endpoint at
   "https://authz.example.net/token.oauth2".

   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>
```

                   Figure 1: 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 might make the
following HTTPS request (with extra line breaks 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-
```

                    Figure 2: Example Request

5. Interoperability Considerations

   Agreement between system entities regarding identifiers, keys, and
   endpoints is required in order to achieve interoperable deployments
   of this profile. Specific items that require agreement are as
   follows: values for the Issuer and Audience identifiers, the location
   of the token endpoint, the key used to apply and verify the digital
   signature over the Assertion, one-time use restrictions on
   Assertions, maximum Assertion lifetime allowed, and the specific
   Subject and attribute requirements of the Assertion. The exchange of
   such information is explicitly out of scope for this specification,
   and typical deployment of it will be done alongside existing SAML Web
   SSO deployments that have already established a means of exchanging
   such information. "Metadata for the OASIS Security Assertion Markup
   Language (SAML) V2.0" [OASIS.saml-metadata-2.0-os] specifies one
   common method of exchanging SAML-related information about system
   entities.

   The RSA-SHA256 algorithm, from [RFC6931], is a mandatory-to-implement
   XML signature algorithm for this profile.

6. Security Considerations

   The security considerations described within the following
   specifications are all applicable to this document: "Assertion
   Framework for OAuth 2.0 Client Authentication and Authorization
   Grants" [RFC7521], "The OAuth 2.0 Authorization Framework" [RFC6749],
   and "Security and Privacy Considerations for the OASIS Security
   Assertion Markup Language (SAML) V2.0"
   [OASIS.saml-sec-consider-2.0-os].

The specification does not mandate replay protection for the SAML
Assertion usage for either the authorization grant or for client
authentication. It is an optional feature, which implementations may
employ at their own discretion.

7. Privacy Considerations

A SAML Assertion may contain privacy-sensitive information and, to
prevent disclosure of such information to unintended parties, should
only be transmitted over encrypted channels, such as Transport Layer
Security (TLS). In cases where it is desirable to prevent disclosure
of certain information to the client, the Subject and/or individual
attributes of a SAML Assertion should be encrypted to the
authorization server.

Deployments should determine the minimum amount of information
necessary to complete the exchange and include only that information
in an Assertion (typically by limiting what information is included
in an <AttributeStatement> or by omitting it altogether). In some
cases, the Subject can be a value representing an anonymous or
pseudonymous user, as described in Section 6.3.1 of the OAuth
Assertion Framework [RFC7521].

8. IANA Considerations

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

This section registers the value "grant-type:saml2-bearer" in the
IANA "OAuth URI" registry established by "An IETF URN Sub-Namespace
for OAuth" [RFC6755].

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

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

   o Change Controller: IESG

   o Specification Document: RFC 7522

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

   This section registers the value "client-assertion-type:saml2-bearer"
   in the IANA "OAuth URI" registry established by "An IETF URN Sub-
   Namespace for OAuth" [RFC6755].

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

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

   o Change Controller: IESG

   o Specification Document: RFC 7522

9. References

9.1. Normative References

   [OASIS.saml-core-2.0-os]
               Cantor, S., Kemp, J., Philpott, R., and E. Maler,
               "Assertions and Protocols for the OASIS Security Assertion
               Markup Language (SAML) V2.0", OASIS Standard
               saml-core-2.0-os, March 2005, <http://docs.oasis-open.org/
               security/saml/v2.0/saml-core-2.0-os.pdf>.

   [OASIS.saml-deleg-cs]
               Cantor, S., Ed., "SAML V2.0 Condition for Delegation
               Restriction Version 1", Committee Specification 01,
               November 2009, <http://docs.oasis-open.org/
               security/saml/Post2.0/sstc-saml-delegation-cs-01.html>.

   [OASIS.saml-sec-consider-2.0-os]
               Hirsch, F., Philpott, R., and E. Maler, "Security and
               Privacy Considerations for the OASIS Security Assertion
               Markup Language (SAML) V2.0", OASIS Standard
               saml-sec-consider-2.0-os, March 2005,
               <http://docs.oasis-open.org/security/saml/v2.0/
               saml-sec-consider-2.0-os.pdf>.

   [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>.

   [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform
              Resource Identifier (URI): Generic Syntax", STD 66,
              RFC 3986, DOI 10.17487/RFC3986, January 2005,
              <http://www.rfc-editor.org/info/rfc3986>.

   [RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data
              Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006,
              <http://www.rfc-editor.org/info/rfc4648>.

   [RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework",
              RFC 6749, DOI 10.17487/RFC6749, October 2012,
              <http://www.rfc-editor.org/info/rfc6749>.

   [RFC6931] Eastlake 3rd, D., "Additional XML Security Uniform
              Resource Identifiers (URIs)", RFC 6931,
              DOI 10.17487/RFC6931, April 2013,
              <http://www.rfc-editor.org/info/rfc6931>.

   [RFC7521] Campbell, B., Mortimore, C., Jones, M., and Y. Goland,
              "Assertion Framework for OAuth 2.0 Client Authentication
              and Authorization Grants", RFC 7521, DOI 10.17487/RFC7521,
              May 2015, <http://www.rfc-editor.org/info/rfc7521>.

9.2. Informative References

   [OASIS.WS-Fed]
              Goodner, M. and A. Nadalin, "Web Services Federation
              Language (WS-Federation) Version 1.2", OASIS Standard, May
              2009, <http://docs.oasis-open.org/wsfed/
              federation/v1.2/os/ws-federation-1.2-spec-os.html>.

   [OASIS.WSS-SAMLTokenProfile]
              Monzillo, R., Kaler, C., Nadalin, T., Hallam-Baker, P.,
              and C. Milono, "Web Services Security SAML Token Profile
              Version 1.1.1", OASIS Standard, May 2012,
              <http://docs.oasis-open.org/wss-m/wss/v1.1.1/
              wss-SAMLTokenProfile-v1.1.1.html>.

   [OASIS.saml-metadata-2.0-os]
              Cantor, S., Moreh, J., Philpott, R., and E. Maler,
              "Metadata for the OASIS Security Assertion Markup Language
              (SAML) V2.0", OASIS Standard saml-metadata-2.0-os, March
              2005, <http://docs.oasis-open.org/security/saml/v2.0/
              saml-metadata-2.0-os.pdf>.

   [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,
             <http://docs.oasis-open.org/security/saml/v2.0/
             saml-profiles-2.0-os.pdf>.

   [RFC6755] Campbell, B. and H. Tschofenig, "An IETF URN Sub-Namespace
             for OAuth", RFC 6755, DOI 10.17487/RFC6755, October 2012,
             <http://www.rfc-editor.org/info/rfc6755>.

   [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,
             <http://www.w3.org/TR/1999/REC-html401-19991224>.

Acknowledgements

Authors' Addresses

   Brian Campbell
   Ping Identity

   EMail: brian.d.campbell@gmail.com

   Chuck Mortimore
   Salesforce.com

   EMail: cmortimore@salesforce.com

   Michael B. Jones
   Microsoft

   EMail: mbj@microsoft.com
   URI: http://self-issued.info/