Send comments to:
Phillip Hallam-Baker, Senior Author
401 Edgewater Place, Suite 280
Wakefield MA 01880
Tel 781 245 6996 x227
Email: pbaker@verisign.com

SAML Core Assertions

Straw-man Architecture

Phillip Hallam-Baker *VeriSign*

Draft Version 0.2: March 12th 2001

SAML Core Assertions

Version 0.2

Table Of Contents

Printed on Wednesday, March 14, 2001

Table of Figures

Executive Summary

We present concrete ‘bits on the wire’ examples of using SAML assertions to control access to network resources.

1 Example Messages

In the following examples

1.1 Web Browser Password Access

Alice is a customer of the business exchange; she needs to access a resource at Carol’s store that is restricted to members of the exchange.

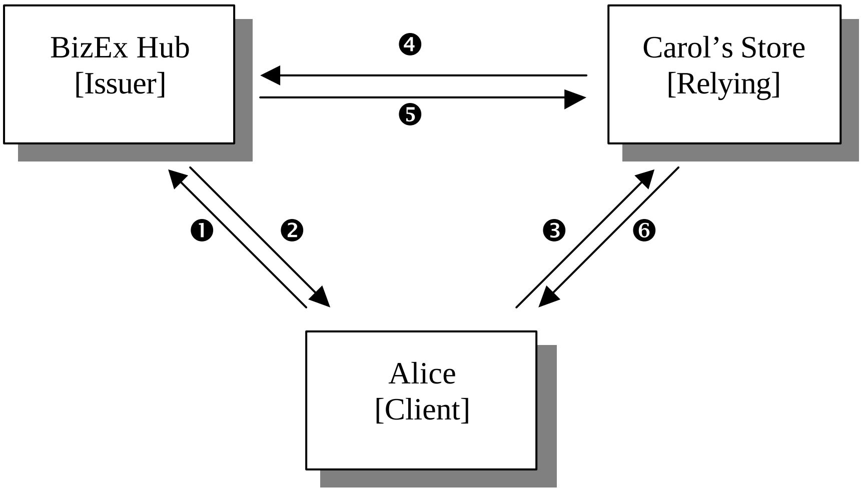


Figure 1: Web Server Log In

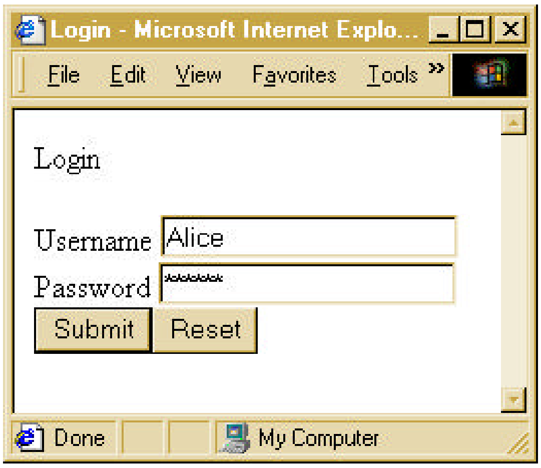
Message	Format	Data
❶ Login	HTTP/SSL Request	Username, Password
❷ Response	HTTP/SSL Response, Ticket (as HTML URL)	Ticket = Account, Validity, Assertion_ID Authenticator
❸ Access	HTTP/SSL Request	Ticket
❹ Pull Assertion	XP Request	Assertion_ID
❺ Assertion	XP Response	Assertion (see below)

Message	Format	Data
❻ Resource	HTTP/SSL Response	Resource Data

1.1.1 ❶ Login

The login data is posted in response to the following HTML form:

```
<form method="POST" action="https://login.bizex.test/login.asp">
  <p>Username <input type="text" name="username" size="20"><br>
  Password <input type="password" name="Password" size="20"><br>
  <input type="submit" value="Submit" name="B1"><input type="reset"
value="Reset" name="B2"></p>
</form>
```



Login

Username Alice

Password ******

Alice enters “Alice” as her username and “secret” as her password. This data is encoded as follows:

```
username=Alice&password=secret
```

1.1.2 ❷ Response

The business exchange service authenticates the username and password [resented by Alice and issues the ticket. The ticket contains the following data:

Item	Size	Data
Assertion_ID	7+2	`[10.20.1.123] AE 02 21`
Validity	4+2	`10-Mar-2001 12:00 for 24 hours`
Account	5+2	`“Alice”`
Authentication	20+2	`HMAC-SHA1 (Assertion_ID, Validity, Account)`
	44	

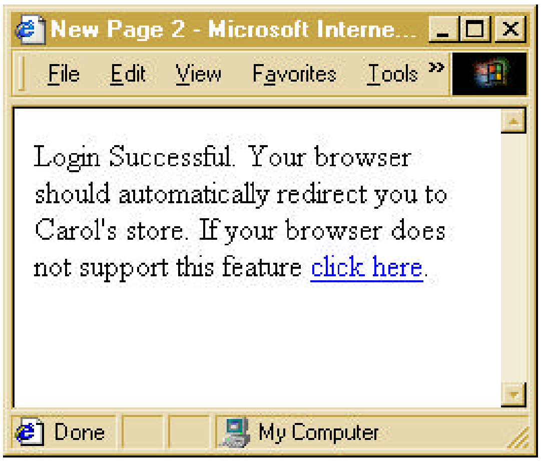
Using base64 encoding this results in a 60 byte string which is passed to Carol encoded as a URL:

```
<% Response.Redirect
"https://store.carol.test/finance/bizex.asp?ticket=jubafOqNEpcwR3RdFsT7
bCqnXPBe5ELh5u4VEy19MzxkXRgrMvavzyBpVR==" %>
<html>
<head><title>Carol’s Store</title</head>

<body>

<p>Login Successful. Your browser should automatically redirect you to
Carol's store. If your browser does not support this feature <a
href="https://store.carol.test/finance/bizex.asp?ticket=jubafOqNEpcwR3R
dFsT7bCqnXPBe5ELh5u4VEy19MzxkXRgrMvavzyBpVR==">click
here</a>.</p>

</body>
</html>
```



Login Successful. Your browser should automatically redirect you to Carol's store. If your browser does not support this feature click here.

1.1.3 ❸ Access

Alice’s Web browser is redirected to Carol’s Web site. The access ticket is encoded in the URL:

```
https://store.carol.test/finance/bizex.asp?ticket=jubafOqNEpcwR3RdFsT7b
CqnXPBe5ELh5u4VEy19MzxkXRgrMvavzyBpVR==
```

1.1.4 ❹ Pull Assertion

Carol’s store receives the URL and decodes the ticket. This tells the server that:

- The issuer of the ticket belongs to the domain `10.20.1.123`. The security policy of Carol’s store recognizes this domain identifier as ‘Bob’s Business Exchange”

- The HMAC value of the ticket agrees with the value calculated from a shared symmetric key exchanged out of band with Bob’s Business Exchange’.

- The ticket was issued to a party that the issuing server authenticated as “Alice”.
- The current time is within the validity interval of the ticket.
- More information may be obtained from the specified assertion.

In this case the assertion reference can be resolved directly since it encodes the IPv4 address of the assertion server and a unique assertion reference.

Alternate means of identifying the assertion

- Compact Locator: IPv4 Address + serial number
- Compact Locator: IPv6 Address + serial number
- Compressed URI [Use 6bit->8bit compression on ASCII URL]
- Compact Name: Large pseudo-unique number
- Serial number alone [does not work across domains]

Carol’s store has a policy of accepting a ticket from Bob’s Business Exchange as proof that a person is a member of the Business Exchange. Certain pages on Carol’s site MAY be accessible using locally managed authorization data and the authorization ticket.

The ticket is an assertion in its own right, typically the ticket encodes a subset of the data encoded in the full assertion.

Access to the resource requested by Alice in this instance requires specific authorization. Carol’s store therefore requests that the issuer supply the full assertion.

```
http://10.20.1.123/?assertion=AE0221
```

Alternatively the ticket might not be bound to a specific assertion and specify only the authenticated account (possibly pseudonymous).

```
<AssertionQuery>
   <Resources>
      <string>http://store.carol.test/finance
   <Subject>
      <Account>Alice
```

This mode of interaction is useful when the number of resources to which access is controlled is large and the Policy Enforcement Point (in this case Carol’s store) does not support de-referencing of higher-level abstractions such as rights.

1.1.5 ❺ Assertion

The assertion specifies the authorizations attached to the Alice account:

```
<SAML>
   <AssertionID>http://www.bizexchange.test/assertion/AE0221
   <Issuer>URN:dns-date:www.bizexchange.test:2001-01-03:19283
   <ValidityInterval>
      <NotBefore>
      <NotOnOrAfter>
   <Conditions>
      <Audience>http://www.bizexchange.test/rule_book.html
   <Subject>
      <Account>Alice
   <Resources>
      <string>http://store.carol.test/finance
      <string>URN:dns-date:www.bizexchange.test:2001-01-
04:right:finance
```

This assertion specifies that Alice is authorized to access two resources:

- The web pages in the tree `http://store.carol.test/finance`
- Resources mapped to the domain specific rights identifier “finance”

The assertion also specifies that it is addressed to a specific audience – informally members of the business exchange, more specifically it is the parties that agree to be bound by the exchange rule book.

1.1.6 ❻ Resource

Carol’s store receives back the assertion and authenticates it. The assertion may be authenticated by means of a secure transport layer, by and XML Signature Digital Signature or MAC, or other means.

The mapping from the resources specified in the assertion is under control of the resource owner. In this case the resource owner simply performs a direct mapping from the first resource identified in the assertion to the site. For a more comprehensive authorization decision see Section 1.3 .

To simplify further accesses Carol’s store issues two cookies to Alice’s browser marked as ‘ephemeral’, i.e. not to be saved to disk.

1. The ticket issued by Bob’s Business Exchange
2. An additional authenticated cookie issued by Carol that specifies authorizations extracted or derived from the assertion.

1.2 SSL Certificate Based Client Authentication

In this scenario Alice authenticates herself by means of a public key mechanism, this avoids the need to perform an initial authentication exchange with the business exchange prior to visiting Carol's store.

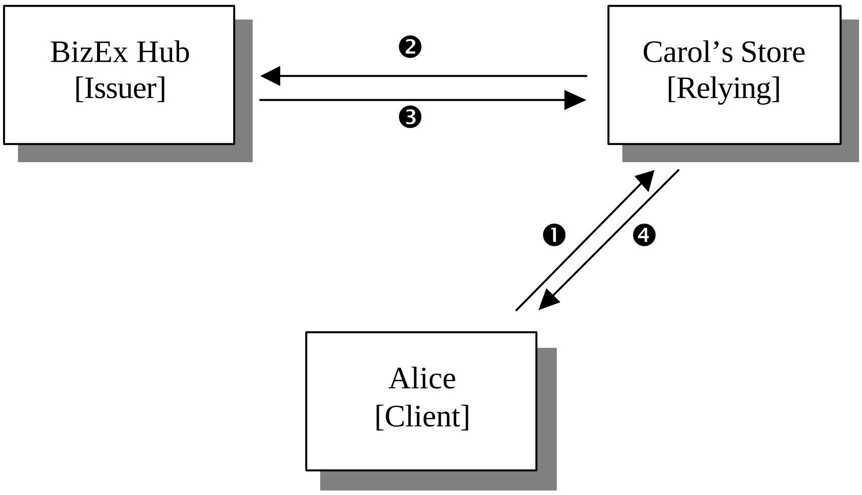


Figure 2: Certificate Based Client Auth

Message	Format	Data
❶ Request	HTTP/SSL Request (with certificate based client authentication)	Certificate, Resource
❷ Pull Assertion	XP Request	Certificate
❸ Assertion	XP Response	Assertion (see below)
❹ Resource	HTTP/SSL Response	Resource Data

1.2.1 ❶ Request

The client authenticates itself to Carol's store using a public key based challenge response scheme, in this case SSL certificate based client authentication. The details of this protocol are not visible to the SAML layer which receives only the result of the authentication, the resource request itself and the credential under which it was authenticated (in this case the certificate).

1.2.2 ❷ Pull Assertion

Carol's store requests authorization information from Bob's Business Exchange:

```
<AssertionQuery>
   <Resources>
      <string>http://store.carol.test/finance
   <Subject>
      <ds:KeyInfo>
         <ds:X509Data>...
```

1.2.3 ❸ Assertion

The business exchange responds that any party authenticating itself with the specified credentials is authorized to access the specified resources:

```
<SAML>
   <AssertionID>http://www.bizexchange.test/assertion/AE0221
   <Issuer>URN:dns-date:www.bizexchange.test:2001-01-03:19283
   <ValidityInterval>
      <NotBefore>
      <NotOnOrAfter>
   <Conditions>
      <Audience>http://www.bizexchange.test/rule_book.html
   <Subject>
      <ds:KeyInfo>
         <ds:X509Data>...
   <Resources>
      <string>http://store.carol.test/finance
      <string>URN:dns-date:www.bizexchange.test:2001-01-
04:right:finance
```

1.2.4 ❹ Resource

The resource is returned to Alice.

1.3 Server Authorization Delegation

In this example Carol's store uses SAML for internal exchange of authorization data. Authorization decisions are controlled by a central Policy Decision Point (PDP) which is consulted by the store server that receives the access request from Alice, the Policy Enforcement Point.

In the interests of completeness the Policy Decision Point consults a separate Policy Store to obtain the access policy for the resource in question. This is however, an extreme example. Few applications would require this degree of granularity. In a more typical example the functions of the Policy Decision Point would be combined with those of either the Policy Store or the Policy Enforcement Point.

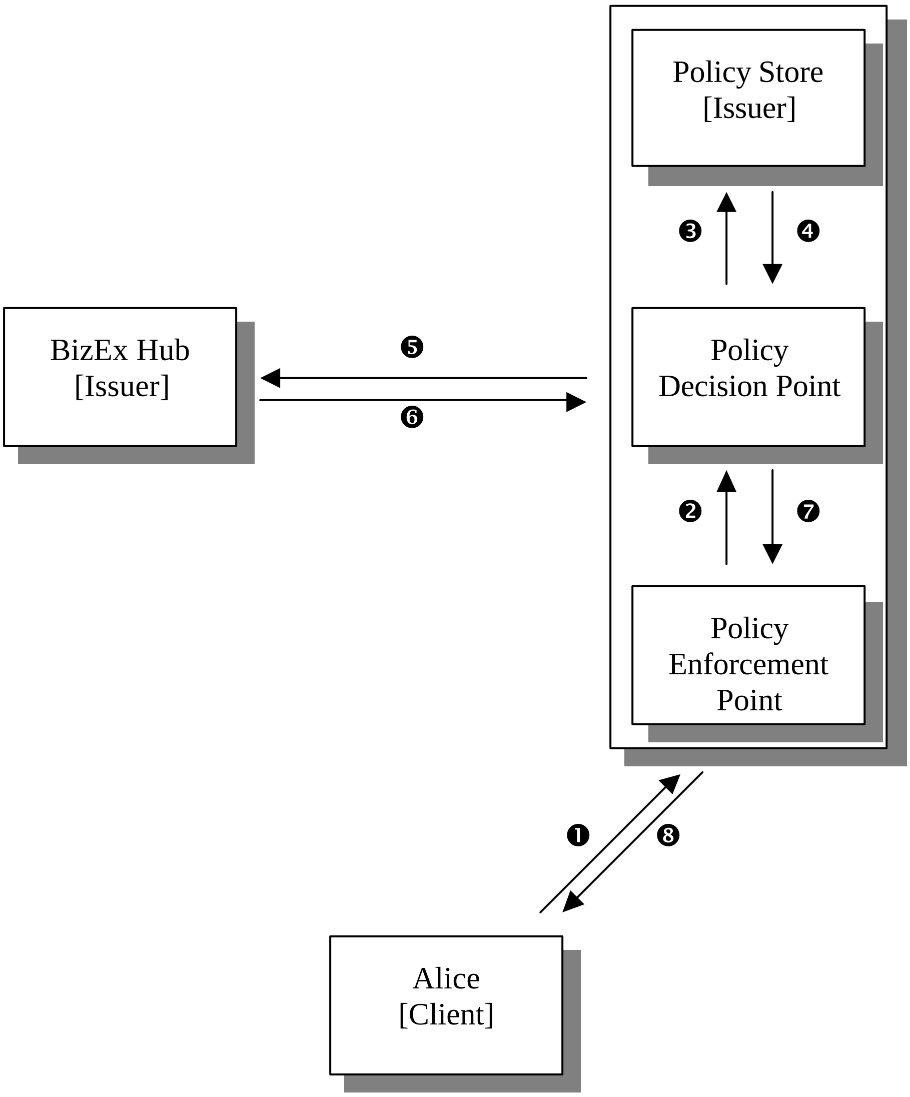


Figure 3: Delegated Decision Point

1.3.1 ❶ Request

Alice requests a resource from Carol’s store. Alice authenticates herself, either by means of public key or as in this case by a ticket issued by Bob’s Business Exchange.

```
https://store.carol.test/finance/bizex.asp?ticket=jubafOqNEpcwR3RdFsT7b
CqnXPBe5ELh5u4VEy19MzxkXRgrMvavzyBpVR==
```

1.3.2 ❷ Request Access Decision

The server receiving the access request delegates authorization decision processing to a Policy Decision Point.

```
<AssertionQuery>
   <Resources>
      <string>http://store.carol.test/finance
   <Subject>

<Ticket>jubafOqNEpcwR3RdFsT7bCqnXPBe5ELh5u4VEy19MzxkXRgrMvavzyBpVR
      ==
   <Respond>
      <string>Result
```

Note that responsibility for authenticating the authentication ticket MAY be placed on either the Policy Enforcement Point or the Policy Decision Point or both under local configuration control.

Depending on circumstances the Policy Enforcement point may require the PDP to return an assertion or just the result of the decision. In this instance only the result is required.

1.3.3 ❸ Request Access Policy

The Policy Decision Point makes a request for an access control policy for the specified resource from the Policy Issuing Server. The format in which the access policy is requested is outside the scope of SAML. A typical policy request might be:

```
<AssertionQuery>
   <Resources>
      <string>http://store.carol.test/finance
```

1.3.4 ❹ Access Policy

The format in which the access policy is specified is outside the scope of SAML. A typical policy request might be:

```
<TASS-ACL>
   <AssertionID>http://policy.carol.test/assertion/
   <Issuer>URN:dns-date:policy.carol.test:2001-03-03:1204
   <ValidityInterval>
      <NotBefore>
      <NotOnOrAfter>
   <Resources>
      <string>http://store.carol.test/finance
   <ACL>
      <ACE>
         <Subject>
            <Right>URN:dns-date:www.bizexchange.test:2001-01-
04:right:finance
         <Permit>RWED
      <ACE>
         <Deny>ED
         <Subject>
            <Right>URN:dns-date:www.bizexchange.test:2001-01-
04:right:ops
         <Permit>R
      <ACE>
         ...
```

1.3.5 ❺ Request Authorization Assertion

The Policy Decision Point does not wish to disclose the specific resource request to the business exchange. Instead the resource rights identifiers specified in the ACL are specified:

```
<AssertionQuery>
   <Resources>
      <string>URN:dns-date:www.bizexchange.test:2001-01-
04:right:finance
      <string>URN:dns-date:www.bizexchange.test:2001-01-04:right:ops
   <Subject>
      <Account>Alice
```

1.3.6 ❻ Authorization Assertion

The SAML authorization assertion is similar to that in the example of section 1.1 .In this case however the Business Exchange only returns the specific information requested:

```
<SAML>
   <AssertionID>http://www.bizexchange.test/assertion/AE0221
   <Issuer>URN:dns-date:www.bizexchange.test:2001-01-03:19283
   <ValidityInterval>
      <NotBefore>
      <NotOnOrAfter>
   <Conditions>
      <Audience>http://www.bizexchange.test/rule_book.html
   <Subject>
      <ds:KeyInfo>
         <ds:X509Data>...
   <Resources>
      <string>URN:dns-date:www.bizexchange.test:2001-01-
04:right:finance
```

1.3.7 ❼ Access Decision

The access decision alone is returned to the client. No validity interval, conditions or resource data was requested.

```
<SAML>
   <Status>Valid
```

1.3.8 ❽ Response

The data is returned to the client.

1.4 SAML Aware Client

An SAML aware client can optimize requests by using the information in an assertion to present the correct data in a request. In addition the need to exchange data between the Issuer and Relying servers directly is avoided.

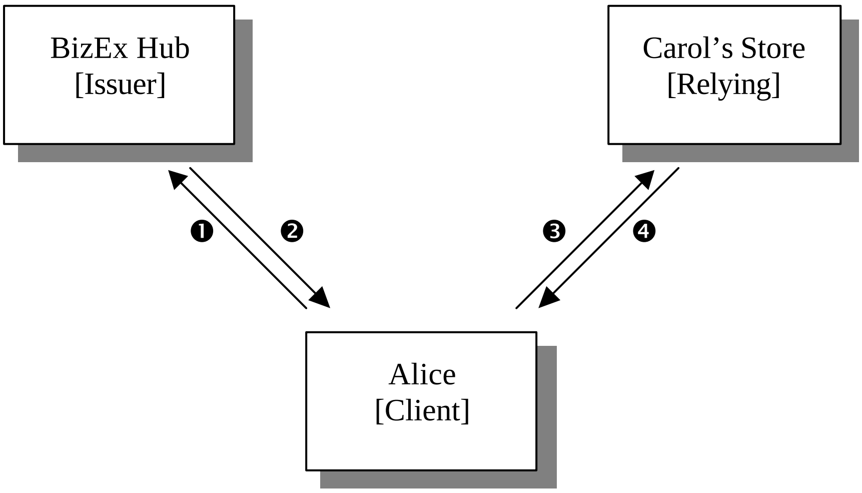


Figure 4: SAML Aware Client

Message	Format	Data
❶ Login	HTTP/SSL Request	Username, Password
❷ Response	HTTP/SSL Response	Assertion
❸ Access	HTTP/SSL Request	Resource_ID, Assertion
❹ Response	HTTP/SSL Response	Data

1.4.1 ❶ Login

Alice authenticates herself to the server using either a password or public key based authentication.

1.4.2 ❷ Response

Bob’s Business Exchange returns the assertion to Alice. In this particular configuration the assertion itself is an authentication instrument and presentation of the assertion alone will grant authorization to the “Alice” account:

```
<SAML>
   <AssertionID>http://www.bizexchange.test/assertion/AE0221
   <Issuer>URN:dns-date:www.bizexchange.test:2001-01-03:19283
   <ValidityInterval>
      <NotBefore>
      <NotOnOrAfter>
   <Conditions>
      <Audience>http://www.bizexchange.test/rule_book.html
   <Subject>
      <Account>Alice
   <Resources>
```

```
      <string>http://store.carol.test/finance
      <string>URN:dns-date:www.bizexchange.test:2001-01-
04:right:finance
   <ds:Signature>...
```

1.4.3 ❸ Access

Alice presents the assertion to Carol’s store:

```
<SAML>
   <AssertionID>http://www.bizexchange.test/assertion/AE0221
   <Issuer>URN:dns-date:www.bizexchange.test:2001-01-03:19283
   <ValidityInterval>
      <NotBefore>
      <NotOnOrAfter>
   <Conditions>
      <Audience>http://www.bizexchange.test/rule_book.html
   <Subject>
      <Account>Alice
   <Resources>
      <string>http://store.carol.test/finance
      <string>URN:dns-date:www.bizexchange.test:2001-01-
04:right:finance
   <ds:Signature>...
```

1.4.4 ❹ Response

The requested data is returned to Alice.

1.4.5 Using Public Key

One disadvantage of the SAML aware client approach is that the client may not have enough information to determine the applicability of a specific rights identifier. If the assertion itself is the authentication token the consequences of sending the assertion to the wrong location are a severe security failure.

A more robust approach is to use an assertion bound to a public key. The corresponding private key may be a long-term private key held by the assertion subject or may be generated ephemerally and established with the assertion issuer through a password based key exchange scheme.

2 Open Issues

2.1 Resource Encoding

Is a URI enough? At present the encoding of resources is simply an array of URIs with no indication of the permitted access modes etc.

```
<Resources>
   <string>http://store.carol.test/finance
```

This syntax can be extended to allow more specific statements:

```
<Resources>
```

```
   <Resource>
      <URI>http://store.carol.test/finance
      <Mode>RWED
```

If multiple resources are specified it may be convenient to allow the default access mode to be specified:

```
<Resources>
   <Mode>RW
   <Resource>
      <URI>http://store.carol.test/finance
   <Resource>
      <URI>http://store.carol.test/goods
```

2.2 Pairing of Requests & Responses

This is really a bindings issue rather than an assertion issue. It is however implicit that there is a means of unambiguously and securely binding requests and responses.