Technical Overview of the OASIS Security Assertion Markup Language (SAML) V1.1

Working Draft 01, 16 February 2004

Document identifier:
sstc-saml-tech-overview-1.1-draft-01

Location:
http://www.oasis-open.org/committees/documents.php?wg_abbrev=security

Editors:
John Hughes, Entegrity Solutions (john.hughes@entegrity.com)
Eve Maler, Sun Microsystems (eve.maler@sun.com)

Abstract:
The Security Assertion Markup Language (SAML) standard defines a framework for exchanging security information between online business partners. It was developed by the Security Services Technical Committee (SSTC) of the standards organization OASIS (the Organization for the Advancement of Structured Information Standards). This document provides a technical description of SAML V1.1.

Status:
This is a non-normative document; readers should refer to the normative specification suite for precise information concerning SAML V1.1. This document is not currently on an OASIS Standard track. It has been produced by the Security Services Technical Committee. Publication of this draft does not imply TC endorsement. This working draft may be updated, replaced, or obsoleted at any time.

Committee members should submit comments to the security-services@lists.oasis-open.org list. Others should submit comments by filling out the form at http://www.oasis-open.org/committees/comments/form.php?wg_abbrev=security. The committee will publish vetted errata on the Security Services TC web page (http://www.oasis-open.org/committees/security/).

For information on whether any patents have been disclosed that may be essential to implementing the SAML specification suite, and any offers of patent licensing terms, please refer to the Intellectual Property Rights web page for the Security Services TC (http://www.oasis-open.org/committees/security/ipr.php).

Table of Contents

1 Introduction....3
2 SAML Overview....4
3 SAML Architecture....6
3.1 SAML Concepts....6
3.2 SAML Structure and Examples....7
3.3 Security of SAML....9
4 Use Cases and Profiles....10
4.1 Browser/Artifact Profile....10
4.1.1 Detailed Processing for the Local-Site-First Scenario....11
4.1.2 Detailed Processing for the Destination-Site-First Scenario....12
4.2 Browser/POST Profile....13
4.2.1 Detailed Processing....14
5 Documentation Roadmap....16

1 Introduction

The Security Assertion Markup Language (SAML) standard defines a framework for exchanging security information between online business partners.

More precisely, SAML defines a common XML framework for exchanging security assertions between entities. As stated in the SSTC charter, the purpose of the Technical Committee is:

> *…to define, enhance, and maintain a standard XML-based framework for creating and exchanging authentication and authorization information.*

SAML is different from other security systems due to its approach of expressing assertions about a subject that other applications within a network can trust. What does this mean? To understand the answer, you need to know the following two concepts used within SAML:

Asserting party

The system, or administrative domain, that asserts information about a subject. For instance, the asserting party asserts that this user has been authenticated and has given associated attributes. For example: This user is ***John Doe***, he has an email address of ***john.doe@acompany.com***, and he was authenticated into this system using a ***password*** mechanism. In SAML, asserting parties are also known as SAML authorities.

Relying party

The system, or administrative domain, that relies on information supplied to it by the asserting party. It is up to the relying party as to whether it trusts the assertions provided to it. SAML defines a number of mechanisms that enable the relying party to trust the assertions provided to it. It should be noted that although a relying party can trust the assertions provided to it, local access policy defines whether the subject may access local resources. Therefore, although the relying party trusts that I'm ***John Doe –*** it doesn't mean I'm given carte blanche access to all resources.

2 SAML Overview

Why is SAML needed? The SSTC developed a number of use cases to drive SAML's requirements. For SAML 1.x, the most important of these use cases described a SAML-based solution to the problem of Web Single Sign-On (SSO). Web SSO allows users to gain access to website resources in multiple domains without having to re-authenticate after initially logging in to the first domain. To achieve SSO, the domains need to form a trust relationship before they can share an understanding of the user's identity that allows the necessary access. Figure 1 illustrates the high-level Web SSO use case; more details about how this is achieved are provided later in the document.

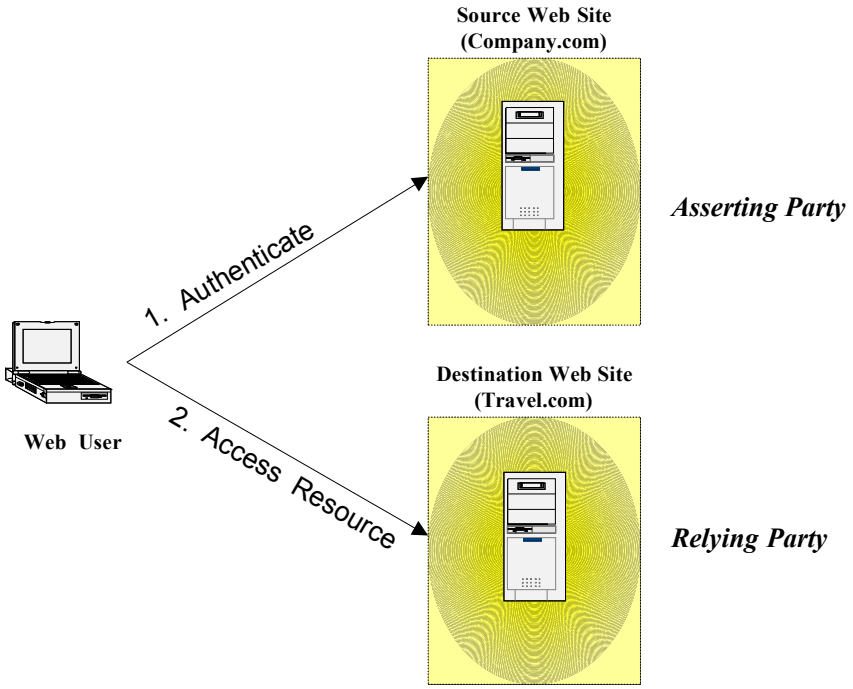

Figure 1: Web SSO High-Level Use Case

Following are some specific scenarios to which SAML's SSO capabilities are relevant:

- **Government Portal**

  A Government department has implemented a centralized portal system. Linked to the portal system are a number of satellite systems. The central portal system maintains the authentication information for all users; however, the satellite systems use a wide range of access management products from a variety of vendors. Users should only be required to be authenticated once, and they can either go initially to the satellite system or the central portal. In this scenario the portal is the asserting party for the whole system and the satellite systems are the relying parties.

- **Travel Bookings**

  Authenticated users of Company.com need to gain access to protected resources at Travel.com in order to make travel arrangements. The Company.com users should not need to have to re-authenticate to Travel.com. In addition, only certain privileged users (for example, above a certain job grade) may book international travel.

- **Goods Purchasing**

  Authenticated users of Company.com use an internal purchasing system to place orders for office supplies from Supplier.com. Supplier.com needs to know the user and their shipping address. Supplier.com also needs to know whether the user is authorized to purchase goods of that value.

The following technical factors drove an urgent need for SAML when it was first created:

- **Limitations of browser cookies:** Before SAML, most SSO products used browser cookies to maintain state so that re-authentication is not required. Browser cookies are not transferred between DNS domains. So, if you obtain a cookie from www.abc.com, then that cookie will not be sent in any HTTP messages to www.xyz.com. This could even apply within an organization that has separate DNS domains. Therefore, to solve the cross-domain SSO problem requires the application of a different approach.
- **SSO interoperability:** Products had implemented cross-domain SSO in completely proprietary ways, meaning that organizations that want to perform cross-domain SSO had to use the same SSO product in all the domains, whether within one organization or across trading partners.
- **Web services:** There is an increasing trend towards inter-organizational distributed computing. Many standards have emerged that facilitate this trend, in particular web services based applications. However, there has been no standard way to convey security attributes associated with inter-organizational communications.

When SAML V2.0 is released in 2004, additional use cases will be supported. To find out more about the scope and design of SAML V2.0, visit the SSTC home page at http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security and review the SAML V2.0 Scope/Work Items document.

3 SAML Architecture

The SAML technology is rooted in XML. The information passed around between asserting parties (SAML authorities) and relying parties is mostly in the form of XML, and the format of these XML messages and assertions is defined in a pair of SAML XML schemas.

3.1 SAML Concepts

SAML has the following key concepts:

- **Assertions:** An assertion is a package of information that supplies one or more statements made by a SAML authority. SAML defines three kinds of statements that can be carried within an assertion. *Authentication statements* say "This subject was authenticated by this means at this time." *Attribute statements* provide specific details about the subject (for example, that a user holds "Gold" status). A*uthorization decision statements* identify what the subject is entitled to do (for example, whether a user is permitted to buy a specified item). The XML format for assertions and their allowable extensions is defined in an XML schema.
- **Protocol:** SAML defines a request/response protocol for obtaining assertions. A SAML request can either ask for a specific known assertion or make authentication, attribute, and authorization decision queries, with the SAML response providing back the requested assertions. The XML format for protocol messages and their allowable extensions is defined in an XML schema.
- **Bindings:** A binding details exactly how the SAML protocol maps onto transport and messaging protocols. For instance, the SAML specification provides a binding of how SAML request/responses are carried within SOAP exchange messages over HTTP.
- **Profiles:** Profiles are technical descriptions of particular flows of assertions and protocol messages that define how SAML can be used for a particular purpose. They are derived from use cases. Use cases and profiles are discussed later on in the document.

Figure 2 shows the relationship between these components.

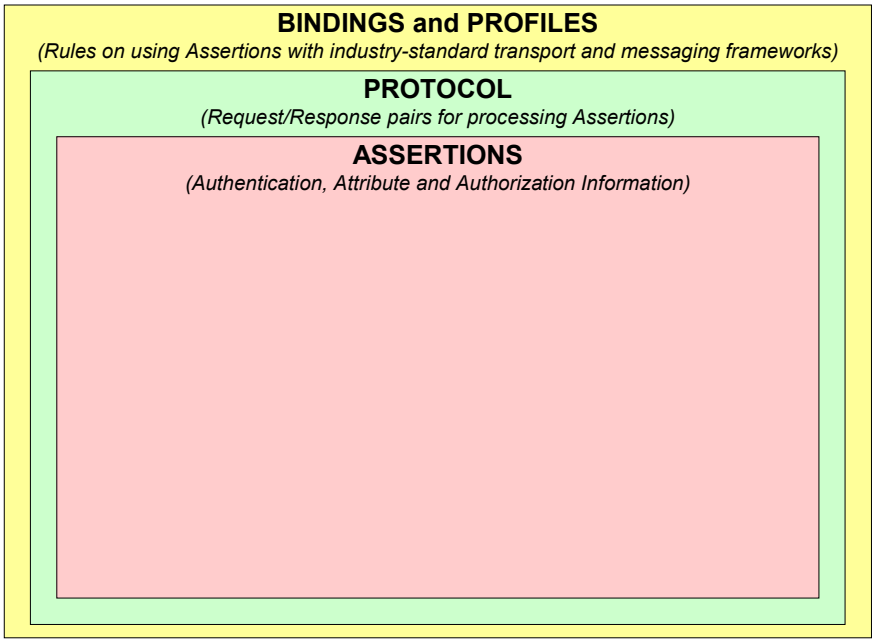

Figure 2: Relationship between SAML Components

3.2 SAML Structure and Examples

The sole binding specified in SAML V1.1 is the “SOAP-over HTTP” binding. Figure 3 illustrates the relationship between SOAP and the SAML protocol messages being transported within the SOAP body.

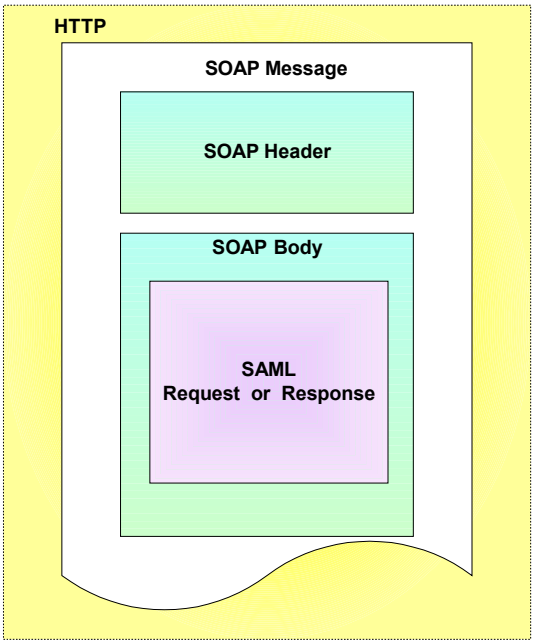

Figure 3: SOAP over HTTP Binding

SAML responses carry assertions that satisfy the parameters of the SAML request. Figure 4 illustrates a SAML response being transported within a SOAP body. Note the following characteristics:

- The SAML response contains header information.
- One more assertions can be transported, although typically only a single assertion is provided in a SAML response.
- An assertion consists of one or more statements. For SSO, typically a SAML assertion will contain a single authentication statement and possibly a single attribute statement.

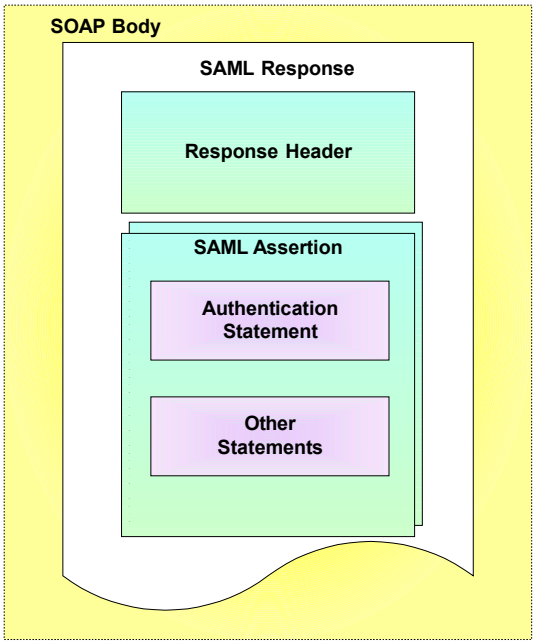

Figure 4: SAML Response Structure

So what does the XML look like? Figure 5 shows an example of a SAML request being transported within a SOAP message. In this example, a SAML assertion is being requested pertaining to a supplied artifact. The use of the artifact is explained later in the Use Case and Profiles section. The SAML request has been highlighted.

```
<env:Envelope
  xmlns:env=”http://www.w3.org/2003/05/soap/envelope/”>
       <env:Body>
             <samlp:Request
               xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
               xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
               MajorVersion="1"
               MinorVersion="1"
               RequestID=" 192.168.16.51.1024506224022"
               IssueInstant="2002-06-19T17:03:44.022Z">
                    <samlp:AssertionArtifact>
                    AAGZE1RNQJEFzYNCGAGPjWvtDIRSZ4
                    lWDqBphqAEYkgG/RBdHoeMsu1f
                    </samlp:AssertionArtifact>
            </samlp:Request>
       </env:Body>
</env:Envelope>
```

Figure 5: SAML Artifact Request

Figure 6 shows how a SAML response is embedded within a SOAP message. The SAML response provides details as to the version of SAML being used and what request it is responding to. The ResponseID, InResponseTo, version numbers, IssueInstant and the status code represent the SAML response header. Within the response is the SAML assertion and typically one or more statements. The SAML response has been highlighted.

```
<env:Envelope
  xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
       <env:Body>
             <samlp:Response
               xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol"
               ResponseID="P1YaA+Q/wSM/t/8E3R8rNhcpPTM="
               InResponseTo=" 192.168.16.51.1024506224022"
               MajorVersion="1"
               MinorVersion="1"
               IssueInstant="2002-06-19T17:05:37.795Z">
                    <samlp:Status>
                          <samlp:StatusCode Value="samlp:Success" />
                    </samlp:Status>
```

```
                        …… SAML ASSERTION AND STATEMENTS

                </samlp:Response>
        </env:Body>
</env:Envelope>
```

Figure 6: SAML Response

Figure 7 shows an example assertion with a single authentication statement. The authentication statement has been highlighted. Note the following:

- The subject (e.g. user) that the authentication pertains to is “joe”. The format of the subject has been defined. In this case its a custom format; however, a number of predefined formats have been provided in the SAML specification, including email addresses and X.509 subject names.
- Joe was originally authenticated using a password mechanism at “2002-06-19T17:05:17.706Z".

```
<saml:Assertion
  xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion"
  MajorVersion="1"
  MinorVersion="1"
  AssertionID="P1YaAz/tP6U/fsw/xA+jax5TPxQ="
  Issuer="www.acompany.com"
  IssueInstant="2002-06-19T17:05:37.795Z">
        <saml:Conditions NotBefore="2002-06-19T17:00:37.795Z"
          NotOnOrAfter="2002-06-19T17:10:37.795Z"/>
        <saml:AuthenticationStatement
          AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password"
          AuthenticationInstant="2002-06-19T17:05:17.706Z">
                <saml:Subject>
                        <saml:NameIdentifier
                          NameQualifier=http://www.acompany.com

Format="http://www.customformat.com/">uid=joe</saml:NameIdentifier>
                        <saml:SubjectConfirmation>
                                <saml:ConfirmationMethod>
                                urn:oasis:names:tc:SAML:1.0:cm:artifact-01
                                </saml:ConfirmationMethod>
                        </saml:SubjectConfirmation>
                </saml:Subject>
        </saml:AuthenticationStatement>
</saml:Assertion>
```

Figure 7: SAML Assertion

3.3 Security of SAML

Just providing assertions from a relying party to an asserting party doesn't make a secure system! How does the relying party trust what is being asserted to it? In addition, what prevents a “man-in-the-middle” attack that grabs assertions to be illicitly “replayed” at a later date? SAML defines a number of security mechanisms that prevent or detect such attacks. The primary mechanism is for the relying party and asserting party to have a pre-existing trust relationship, typically involving a Public Key Infrastructure (PKI). Whilst use of a PKI is not mandated, it is recommended. Use of particular mechanisms is described for each profile; however, an overview of what is recommended is provided below:

- Where ***message integrity*** and ***message confidentiality*** are required, then HTTP over SSL 3.0 or TLS 1.0 is recommended.
- When a relying party requests an assertion from an asserting party then ***bi-lateral authentication*** is required and the use of SSL 3.0 or TLS 1.0 using server *and* client authentication are recommended.
- When an assertion is “pushed” to a relying party (as with the Browser/POST profile), then it is mandated that the response message be ***digitally signed*** using the XML digital signature standard.

4 Use Cases and Profiles

Early in its business requirements analysis, the SSTC defined a number of use cases for SAML. To date, only the Web SSO use case has been profiled. With the emergence of SAML V2.0 in 2004, a number of other use cases will also be profiled.

SAML V1.1 has defined Web SSO two profiles. Theses profiles assume:

- Use of a standard commercial web browser using either HTTP or HTTPS
- The user has authenticated to the local source site
- The assertion's subject refers implicitly to the user that has been authenticated

The profiles are:

- ***Browser/Artifact Profile***: This represents a “pull model”. A special form of reference to the authentication assertion (called an artifact) is sent to the relying party, which can using this reference to obtain (or pull) the assertion from the Asserting Party.
- ***Browser/POST Profile:*** This represents a “push model”. An assertion is POSTed (using the HTTP POST command) directly to the relying party.

We shall now go on to describe in detail each of these profiles.

4.1 Browser/Artifact Profile

This Browser/Artifact profile is based on a pull model. Figure 8 illustrates the overall processing.

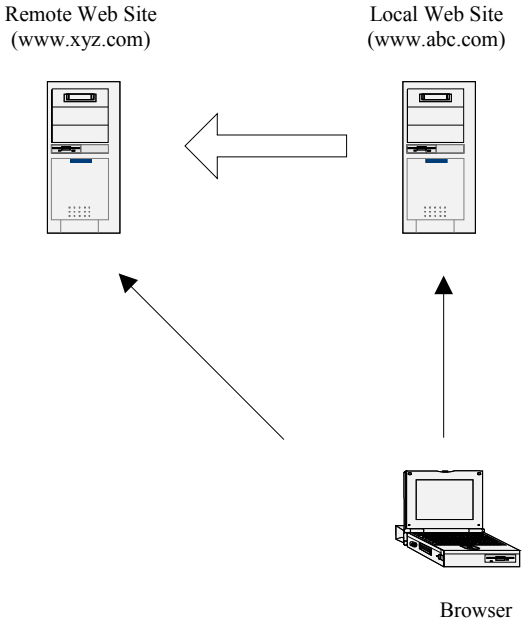

Figure 8: Browser/Artifact Profile Overview

In summary, the processing is as follows:

1. A user has an authenticated session on the local source site.
2. The user wants to access a resource on the remote web site and is directed there. In the HTTP message, an HTTP query variable is passed called an *artifact*. The artifact is a base-64 encoded string. It consists of a unique identity of the source site (called the Source ID) and a unique reference to the assertion (called the AssertionHandle). The artifact therefore enables the remote web site to

reference an assertion on a given web site.

3. The remote site needs to determine the identity and entitlements of the user and sends a SAML request, containing the artifact, to the local site (the asserting party) asking it what it can assert about the user. The assertions are transferred back in a SAML response.
4. The remote site then can make whatever authentication and authorization decisions it needs to, based on the received assertion(s).

Two scenarios are possible in this use case:

- ***Local-site-first:*** The user visits their local site first and is authenticated at the local site before using a click-through link to gain access to the destination site.
- ***Destination-site-first:*** The user visits the destination site first; however, they need to be authenticated at the local (source) site prior to being granted access to resources on the destination site. This scenario typically represents a centralized portal architecture.

4.1.1 Detailed Processing for the Local-Site-First Scenario

The following figure shows the processing and message flows for the Browser/Artifact profile in the Local-Site-First scenario.

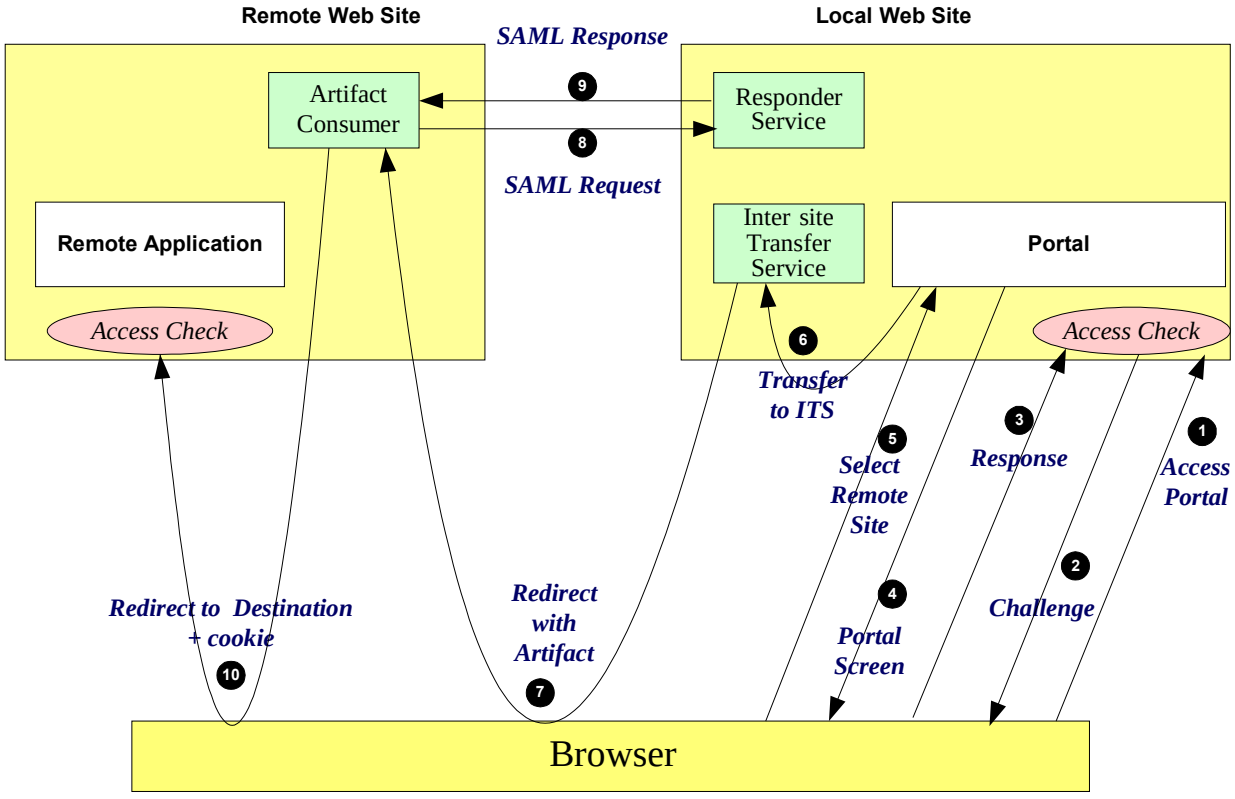

Figure 9: Browser/Artifact Profile - Detailed Processing

The processing is as follows:

1. The user accesses the local web site (www.abc.com).
2. The local web site performs an access check and determines that the user does not have a current session and requires the user to be authenticated. As a result, the user is challenged to authenticate.
3. The user supplies back credentials, for instance username and password.
4. If the authentication is successful, then a session is created for the user and the appropriate welcome screen of the Portal application is displayed to the user.

5. The user selects a menu option (or function) on the displayed screen that means the user wants to access a resource or application on a remote web site www.xyz.com (although, of course, the user may not be made aware of this).
6. The portal application then directs the request to the local Inter-site Transfer Service (in this example, hosted on the same web site). The request contains the URL of the resource on the remote site. This is known as the TARGET URL. For instance, the portal application will issue an HTTP GET to the Inter-site Transfer Service on the www.abc.com site which is listening on port 8002. The URL would look something like the following (without the URL encoding):
7. https://www.abc.com:8002/InterSiteTransfer?TARGET=http://www.xyz.com/index.asp
8. The Inter-site Transfer Service generates an assertion for the user while also creating an artifact (The Asserting Party). The artifact contains the source ID of the www.abc.com SAML responder together with a reference to the assertion (the AssertionHandle). The Inter-site Transfer Service then sends back an HTTP redirection response to the browser, with the HTTP location header containing the URL of the Artifact Receiver service, the TARGET URL, and the artifact. On processing the redirect, the Browser will issue an HTTP GET of the form provided below, where the <artifact> is a base 64 encoded number. This will be sent to the server hosting the TARGET URL.

   https://www.xyz.com:7001/ArtifactConsumer?TARGET=http://www.xyz.com/index.asp&SAMLart=<artifact>

9. On receiving the HTTP message, the Artifact Receiver, on the remote web site, extracts the source-ID. A mapping between source IDs and remote Responders will already have been established administratively. The Artifact Receiver will therefore know that it has to contact the www.abc.com SAML responder at the prescribed URL. The www.xyz.com Artifact Receiver will send a SAML request to the www.abc.com SAML responder containing the artifact supplied by the Inter-site Transfer Service of www.abc.com.
10. The www.abc.com SAML responder supplies back a SAML response message containing the assertion generated during step 7. In most implementations, if a valid assertion is received back, then a session on www.xyz.com is established for the user (the relying party) at this point.
11. The Artifact Receiver, on the remote web site, sends a redirection message containing a cookie back to the browser. The cookie identifies the session. The browser then processes the redirect message and issues a HTTP GET to the TARGET resource on www.xyz.com. The GET message contains the cookie supplied back by the Artifact Receiver. An access check is then back to establish whether the user has the correct authorization to access the www.xyz.com web site and the index.asp resource.

4.1.2 Detailed Processing for the Destination-Site-First Scenario

In a number of use case scenarios the user may not initially access the asserting party. For instance, in the case of a centralized portal system, a user may first access a satellite system but is required to be authenticated centrally. This is known as “Destination-Site-First”. The processing is a variant of the previous use case and is as follows:

1. The user accesses the remote web site (www.xyz.com).
2. The local web site performs an access check and determines that the user must be authenticated by the central site. A redirection is issued to the central site. Typically, this redirection is to the central site's Inter-site Transfer Service.
3. The central site (the asserting party) challenges the user.
4. The user supplies back credentials, for instance username and password.
5. The portal application then directs the request to the local Inter-site Transfer Service (in this example, hosted on the same web site). The request contains the URL of the resource on the remote site originally requested.
6. The Inter-site Transfer Service generates an assertion for the user while also creating an artifact. The artifact contains the source ID of the www.abc.com SAML responder together with a reference to the assertion (the AssertionHandle). The Inter-site Transfer Service then sends back an HTTP redirection response to the browser, with the HTTP location header containing the URL of the Artifact Receiver service, the TARGET URL, and the artifact.

7. On receiving the HTTP message, the Artifact Receiver sends a SAML request to the www.abc.com SAML responder containing the artifact supplied by the Inter-site Transfer service of www.abc.com.
8. The www.abc.com SAML responder supplies back a SAML response message containing the assertion generated during step 7.
9. The Artifact Receiver, on the remote web site, sends a redirection message containing a cookie back to the browser. The cookie identifies the session. The Browser then processes the redirect message and issues a HTTP GET to the TARGET resource on www.xyz.com that was originally requested in step 1.

Figure 10 illustrates the processing steps.

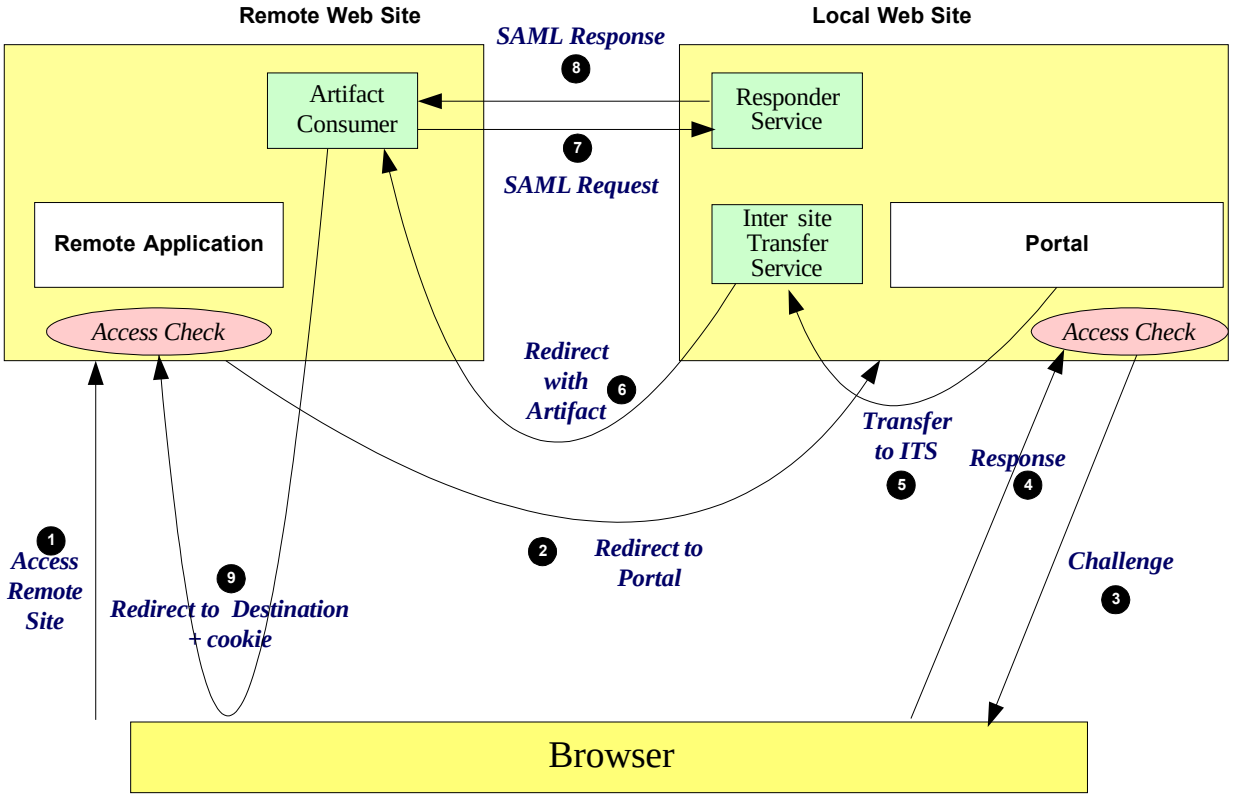

Figure 10: Browser/Artifact Profile - Destination-Site-First – Detailed Processing

4.2 Browser/POST Profile

This profile uses the push model and does not rely on an artifact. The processing, in summary, is as follows:

- A user has an authenticated session on the local source site (the asserting party).
- The user wants to access a resource on the remote web site. An HTML form is provided back to the browser from the local site. The form contains the assertion about the user. The form will also contain a button (or other type of trigger) that causes a POST of the assertion to the remote site to occur. This could also be in the form on JavaScript "auto-submit" action so that the user doesn't have to press a button.
- The remote site then can make whatever authentication and authorization decisions it needs to, based on the received assertion contained within the POST message.

The following detailed description describes a “local-site-first” use case; however, this profile can also work in a “destination-site-first” situation.

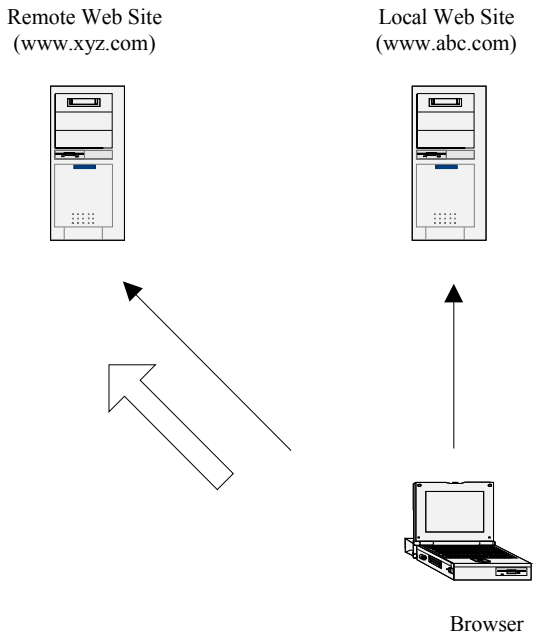

Figure 11 – Browser/POST Profile Overview

4.2.1 Detailed Processing

The processing is as follows:

1. The user accesses the local web site (www.abc.com)
2. The local web site performs an access check and determines that the user does not have a current session and requires the user to be authenticated. As a result, the user is challenged to authenticate.
3. The user supplies back credentials, for instance username and password.
4. If the authentication is successful, then a session is created for the user and the appropriate welcome screen of the Portal application is displayed to the user.
5. The user selects a menu option (or function) on the displayed screen that means the user wants to access a resource or application on a remote web site www.xyz.com.
6. The portal application then directs the request to the local Inter-site Transfer Service (in this example, hosted on the same web site). The request contains the URL of the resource on the remote site (the TARGET URL).
7. The Inter-site Transfer Service sends a HTML form back to the browser. The HTLM FORM contains a SAML response, within which is a SAML assertion. The SAML specifications mandate that the response must be digitally signed. Typically the HTML FORM will contain an input or submit action that will result in a HTTP POST.
8. The browser user will action the HTTP POST causing the SAML response to be sent to the destination's (relying party) Assertion Consumer service.
9. The replying party's Assertion Consumer validates the digital signature on the SAML Response, if this validates it the sends a redirect to the browser causing it to access the TARGET resource. An access check is then made to establish whether the user has the correct authorization to access the www.xyz.com web site and the TARGET resource.
10. The TARGET resource is returned to the browser.

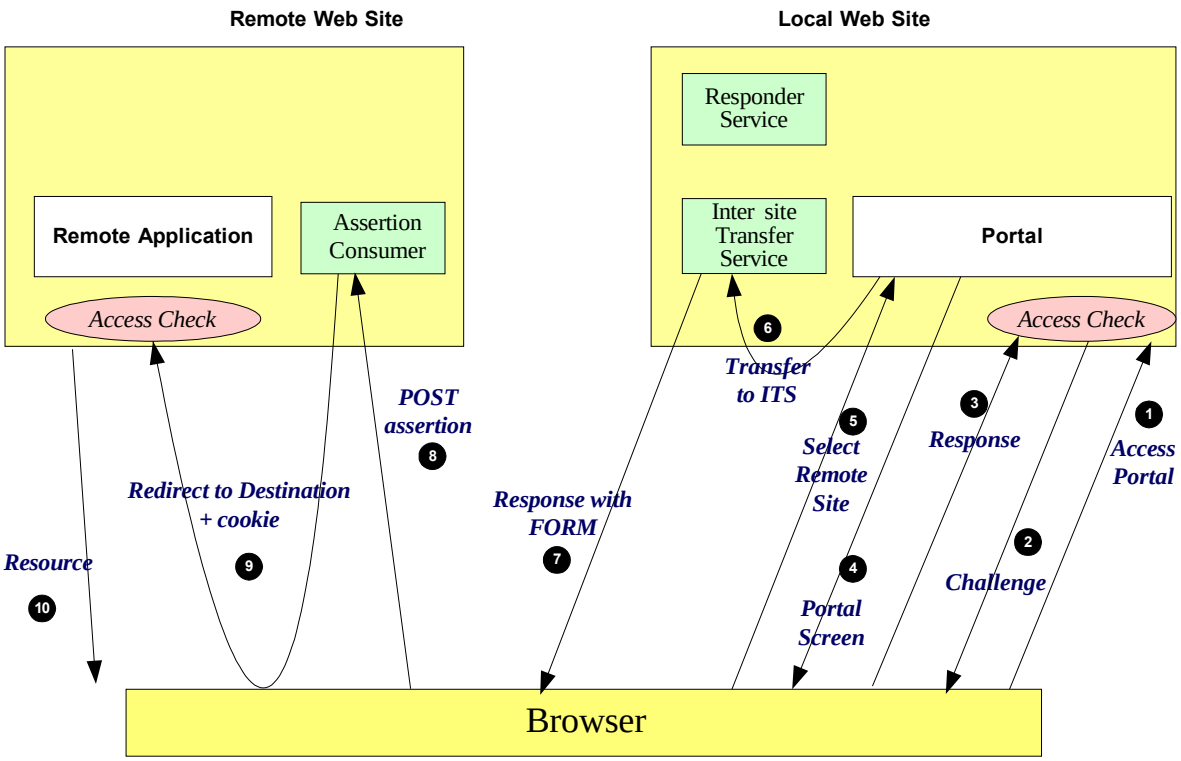

Figure 12: Browser/POST Profile – Detailed Processing

5 Documentation Roadmap

Following is the SAML V1.1 suite of specifications, approved and published on 2 September 2003.

Short Name	Document Identifier	Description
Assertions and Protocol (also known as the “core” spec)	oasis-sstc-saml-core-1.1	Defines the syntax and semantics for XML-encoded assertions about authentication, attributes and authorization, and for the protocol that conveys this information.
Assertion schema	oasis-sstc-saml-schema-assertion-1.1	The schema document governing the formal definition of SAML's XML-form assertions.
Protocol schema	oasis-sstc-saml-schema-protocol-1.1	The schema document governing the formal definition of SAML's XML-form request and response protocol messages.
Bindings and Profiles	oasis-sstc-saml-bindings-1.1	Defines protocol bindings and profiles for the use of SAML assertions and request-response messages in communications protocols and frameworks.
Security and Privacy Considerations	oasis-sstc-saml-sec-consider-1.1	Describes and analyzes the security and privacy properties of SAML. (Note that the Bindings and Profiles specification also contains some security information pertaining to each profile.)
Conformance Program Specification	oasis-sstc-saml-conform-1.1	Describes the program and technical requirements for SAML conformance.
Glossary	oasis-sstc-saml-glossary-1.1	Defines terms used throughout the SAML specifications and related documents.

The following are other documents related to SAML V1.1.

Short Name	Document Identifier	Description
Technical Overview	sstc-saml-tech-overview-1.1	This document. It provides an overview of basic SAML goals and concepts and the flows specified in the SAML profiles.
Differences from V1.0	sstc-saml-diff-1.1-draft-01	A description of the changes made to the SAML specifications from V1.0 to V1.1.
V1.1 Errata	sstc-saml-errata-1.1-draft-16	A list of problems and resolutions kept during the public review of the SAML V1.1 Committee Specifications. Note that this is **not** a list of errata on the final SAML V1.1 specifications. **This is a historical document only.**
V1.1 Issues	sstc-saml-1.1-issues-draft-02	The list of issues from which the SSTC worked during the creation of SAML V1.1. **This is a historical document only.**

These documents can all be found at the public SAML home page:

http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security

A. Notices

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification, can be obtained from the OASIS Executive Director.

OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to implement this specification. Please address the information to the OASIS Executive Director.