Federated Identity Management

An Overview of Concepts and Standards

Eve **Maler**

Abstract

XML is all about labeling information so that it can be accessed and used more widely, accurately, and interoperably. One of the most valuable kinds of information is related to identity -- the attributes that make each of us unique. Is it possible to achieve a networked world in which individuals and businesses can engage in virtually any transaction, on any network device, using vital identity information to enrich and customize the experience without compromising privacy and security? It turns out that we're well on our way, and XML is playing a key role in getting us there.

This paper explores the use cases, concepts, and interoperability profiles defined by the most mature and widely deployed solutions in the federated identity space to date: the Security Assertion Markup Language (SAML) and Liberty Alliance standards. SAML has been providing XML protocols for single sign-on and identity management since 2001, and Liberty has developed a full suite of SAML-based federated identity specifications, along with business and legal guidelines to make deployment as successful and safe as possible.

Table of Contents

1. Identity, Digitized and Federated

Your identity is the collection of characteristics, traits, and preferences that makes you unique. To a first approximation, your *digital* identity is the collection of *electronic information* that defines you as unique -- digitized representations of those characteristics, traits, and preferences. You use this information frequently to gain access to security-protected online information and to customize your online experience. For example, typically you would log in to your bank's website using special knowledge held by you in order to prove that you have the right to move your money around, and once you've logged in, the display of your account reflects your own personal financial data and the display settings you've chosen.

The online world presents a whole host of opportunities to use identity to increase the quality and safety of many kinds of interactions, most especially commerce. In fact, for any business or data transaction of significant consequence, identity is a key component for ensuring security, access control, personalization, and even regulatory compliance. This is why most of us have a plethora of representations of our digital identity, one for each application or service (often, but not always, associated with a username-password combination). Unfortunately, these representations of identity information are typically entirely independent of each other -- silos of identity -- and their proliferation presents a problem for both convenience and security.

Developers of online services usually control all the services within their own organization's domain (for example, throughout a single company when it comes to business-to-employee services), and thus can dictate the use of one set of security infrastructure for authenticating and authorizing users and propagating their identity attributes. Security and identity management in this fashion is full of well-established technology and practice. Web-based applications and web services, however, present tantalizing possibilities for reaching across and connecting silos of information and activity. Crossing domain boundaries to share identity information in particular becomes desirable in many common scenarios.

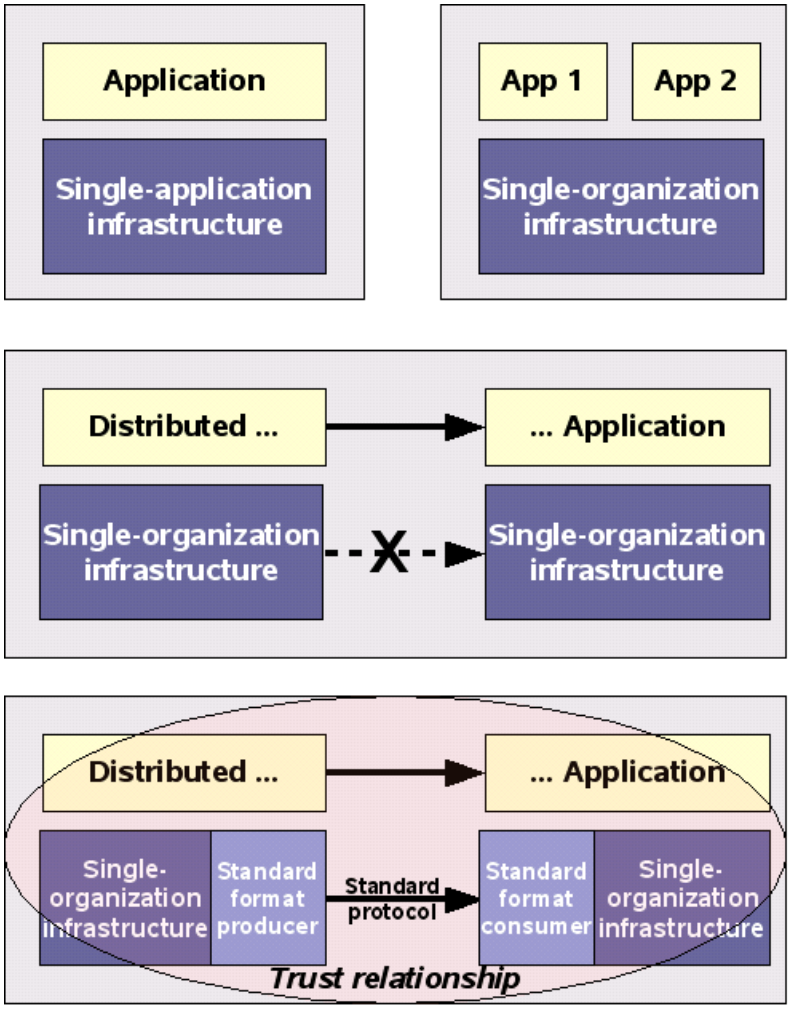
For example, if a company offers its employees a retirement savings and investment plan (such as 401(k) programs in the U.S.), typically the plan's management is outsourced to a financial services company. For each employee, there is a subset of their digital identity information that is interesting and appropriate for their employer and their 401(k) administrator to share in order to implement single sign-on (one login session that is reusable across multiple online applications) and personalized account access. Another outsourcing example might be a mobile telecommunications operator offering long-running pay-for-play games run by a third-party service provider. Another reason to share identity information is the pace of mergers and acquisitions in many industries: The merging of IT functions often leads to a multiplicity of application domains and vendor solutions within a single organization, with each employee needing to apply the same identity information to a variety of otherwise disconnected online tasks. Finally, the move to e-government systems, with government agencies providing online services to citizens, imposes a requirement for portable identity information that is not tied to any one vendor's solution.

As desirable as it is to share identity information, the proposition can be a difficult one. One reason has to do with abstraction: the differences between infrastructure technology choices that must be bridged when domains are crossed. Another important reason is privacy: the controls that must be put into place to protect people's (and organizations') confidential information from zinging around the net indiscriminately.

The following figure illustrates the disconnect among security infrastructures when components of an application become ever more loosely coupled. The first panel shows a classic monolithic application with a single security infrastructure. The second shows the beginnings of an application with two components, which can rely on the single security infrastructure choice that the organization has dictated. The third panel illustrates the difficulty when disparate security infrastructures can't make their data “jump across the chasm” to serve the highly distributed cross-organizational applications that rely on them. The final panel shows the basic requirements for a solution that bridges the gap:

- Standard formats for identity information and the security data that relates to it
- Standard, secure protocols for requesting and sending the identity information

- Trust relationships that are governed by technical, business, and legal frameworks and that take into account privacy considerations



Identity can be said to be *federated* when it is shareable across domain and platform boundaries in this fashion. This paper focuses on introducing two key efforts for XML-based federated identity that meet these requirements: SAML and Liberty.

2. The Standards Landscape for XML-Based Security and Identity

A number of technologies and standards have been widely used for many years to solve the problems of security and identity, among them directory standards like LDAP and security standards such as Kerberos and the various types of public key infrastructure PKI. With XML gaining popularity as a solution for more loosely coupled computer-to-computer communication, several standards efforts were launched that applied XML and web services technologies to these solutions -- codifying existing practice but not inventing, for example, wholly new authentication methods or cryptographic key distribution frameworks. In large part, these efforts have been complementary rather than overlapping. Following are notable examples:

- XML Signature [XML-Sig] and XML Encryption [XML-Enc]: These two standards from the World Wide Web Consortium (W3C) define methods of digital signing and encryption that are particularly suited to the opportunities and challenges presented by XML as a data format. They allow for selective signing and encryption of XML substructures and for the representation of signed and encrypted content in XML form. They are both widely used as the security underpinning for other XML-related standards.

- XML Key Management Specification [XKMS]: This W3C standard defines a set of XML web services for registering and looking up cryptographic keys. It provides a technology-neutral layer above particular PKI systems such as SPKI and PKIX and allows client devices to offload key management tasks to an external service.

- Service Provisioning Markup Language [SPML]: This OASIS standard allows for a platform-independent method of provisioning user accounts.

- Extensible Access Control Markup Language [XACML]: This OASIS standard defines a way to express access control policies in XML and provides a protocol for interacting with an XACML policy decision point to get authorization decisions.

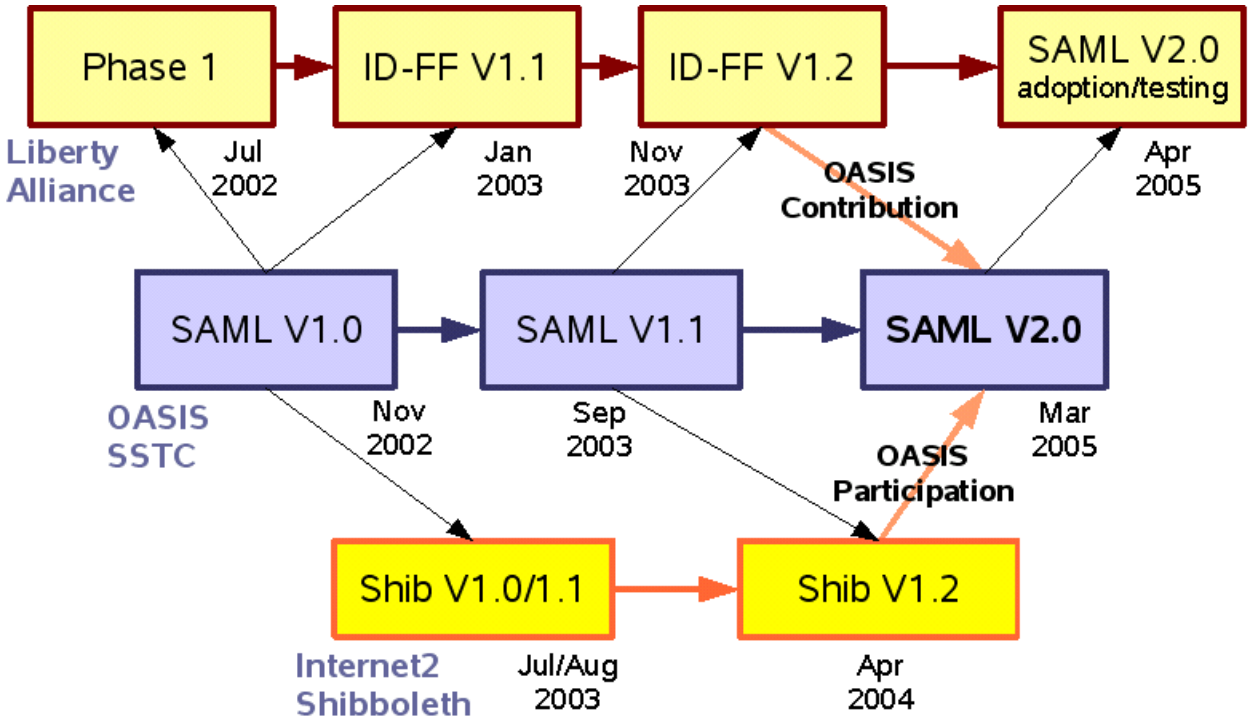
- Security Assertion Markup Language [SAML]: This standard provides XML formats for encoding identity information, XML protocols for exchanging that information, and profiles for achieving interoperability in performing common identity management tasks. Developed by the Security Services Technical Committee [SSTC] of OASIS, SAML has proven to be something of a “universal solvent” for identity and security information and it is being adopted as the underpinning for a number of other modern technologies in this space. SAML Version 2.0 was approved in March 2005.

- Liberty Alliance [Liberty] standards: The Liberty Alliance Project is an alliance of more than 150 companies, nonprofits, and government organizations from around the globe. It develops open standards for federated network identity, with an emphasis on supporting all existing and emerging network devices. It produces technology specifications such as the Identity Federation Framework [ID-FF] and Identity Web Services Framework [ID-WSF], along with technical, business, and legal guidelines for adoption and deployment. It also provides interoperability testing and certification services.

- Web Services Security: SOAP Message Security [WS-Security]: This OASIS standard and its series of companion “security token profiles” (including one that uses SAML [STP]) define how to apply digital signing and encryption to SOAP web service messages for end-to-end security protection.

SAML and Liberty have an intertwined history. Early versions of Liberty's Identity Federation Framework [ID-FF] were layered on early versions of the SAML standard, and ultimately Liberty's technical specification work was folded into SAML V2.0 in an explicit convergence project. Today, Liberty has adopted SAML's most recent version for its identity federation solution, but continues to provide interoperability testing and deployment guidelines, along with its own technical specifications for identity-based web services. Also included in the SAML V2.0 design effort were features of an identity federation system called Shibboleth [Shib], which as part of the Internet2 higher-education initiative provides a solution for sharing identity information across university domain boundaries for research purposes. Shibboleth hosts the OpenSAML.org open-source effort [OpenSAML] and, like Liberty, is moving to a SAML V2.0 basis.

The following diagram shows the relationship of the various SAML, Liberty ID-FF, and Shibboleth versions. The dates indicate the final publication, standardization, or announcement date.



In achieving convergence with Liberty ID-FF V1.2, SAML V2.0 was not made backwards compatible in terms of its syntax. SAML integrated ID-FF's functionality in an abstract manner, rather than retaining Liberty-namespaced structures whole. However, it did adopt Liberty terminology for various concepts. (It should be noted that identity management products offer conformance to various SAML and Liberty protocol versions, and deployments of pre-existing versions are found in the wild. In any federated identity deployment, some kind of agreement would be necessary among the communicating parties about versions to be used.) In keeping with the spirit of convergence, technical and terminological details in this paper are consistent with SAML V2.0 rather than one of the earlier versions that fed into it.

3. Essential Concepts

The following are informal descriptions of key SAML terms and concepts. For formal definitions, see the SAML Glossary [SAMLGloss] and Liberty Technical Glossary [LibGloss].

3.1. Entities and Subjects

An *entity* (or *system entity*) is any active element of a computer network or system. This can include, for example, individual software components and human actors.

A *principal* is an entity whose identity can be authenticated -- often but not exclusively a human being. A *subject* is a principal in the context of a security domain. In practice, SAML uses these two terms pretty much interchangeably. SAML defines a `Subject` element, for example, but some of the language in the standard refers to principals.

3.2. Identities and Identifiers

We have already been using *identity* to mean the essence of an entity, often described by its characteristics, traits, and preferences. Sometimes it's important for the sake of privacy to preserve *anonymity*, the state of keeping an identity unknown or concealed.

In the digital information world, identity information is manipulated by means of an *identifier*, a data object that uniquely refers to a particular entity. Sometimes a *pseudonym*, a special kind of identifier that preserves privacy, must be employed. SAML's `NameID` element is one of its important structures for the job of conveying identifiers.

3.3. Assertions and Statements

In general terms, an *assertion* is a declaration of fact, according to some entity. The SAML `Assertion` element is the central means of providing identity information about particular subjects. As part of its system of conveying assertions, SAML invents the notion of an *artifact*, a tiny string that represents a particular assertion and can be used to retrieve the real thing.

SAML's assertions are compound structures that can contain multiple *statements* about a subject's authentication, attribute, or authorization details.

3.4. Parties and Providers

An entity that produces a SAML assertion about a subject is an *asserting party* (sometimes also known as a *SAML authority*), and an entity that decides to take action based on receiving one of these assertions is a *relying party*. A trust relationship is assumed to exist between them.

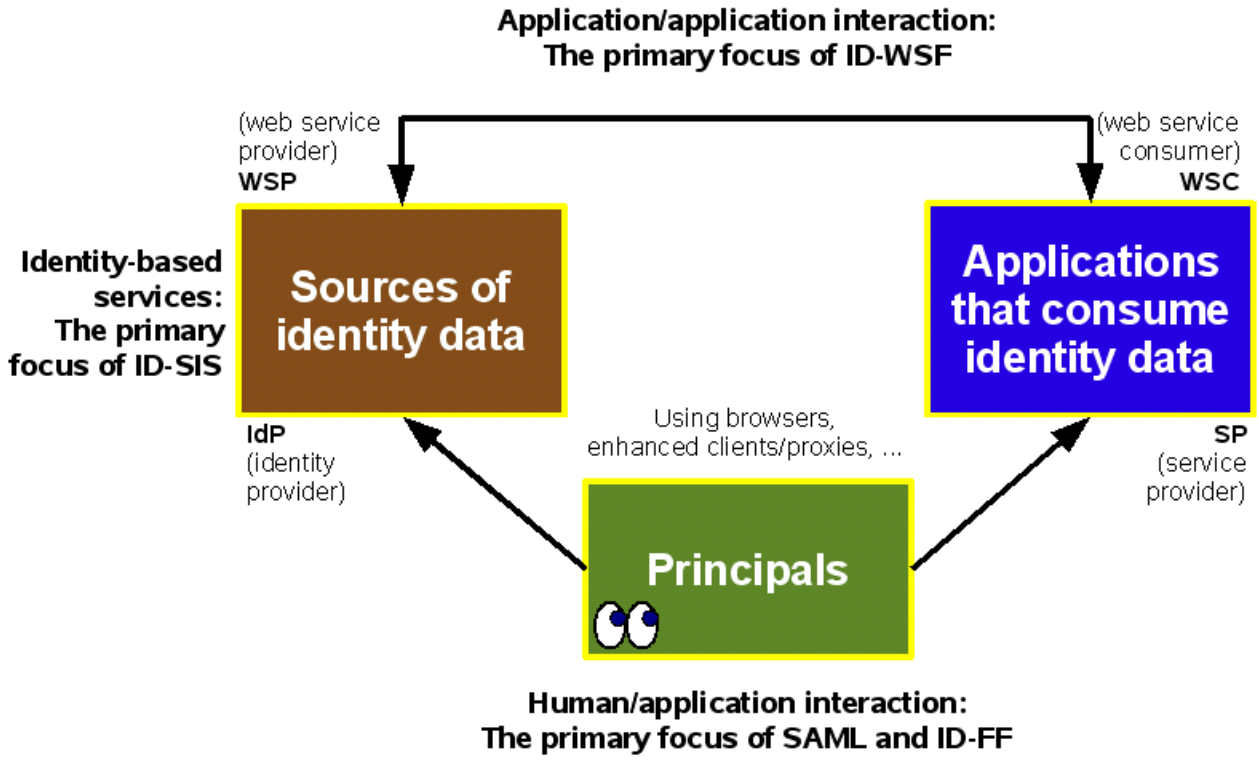
In slightly more concrete terms, typically there are one or more online *service providers* or *SPs* that can make use of assertions about a subject in order to control access and provide customized services, and accordingly they become the relying parties of an asserting party called an *identity provider* or *IdP*. You can think of an identity provider as a special class of service provider.

3.5. Identity Interactions Among Entities

For single sign-on and related scenarios that effect customization and access control, an IdP serves as a hub for one or more SP spokes, in an arrangement that Liberty calls a *circle of trust* -- an IdP/SP relationship that has both technical and business dimensions. The IdP is where the principal goes to get authenticated, and this act of authentication and other aspects of the principal's identity can be shared and reused by the SPs. For example, you might log in to your employee portal (IdP), whose back-end security system exchanges a SAML assertion with your 401(k) website (SP) in order to allow you access to your retirement plan without having to log in again. This basic template of interaction also paves the way for web services to communicate amongst themselves about your identity. For example, they might allow the 401(k) site (operating as a web service client) to look up your address as maintained by the employee portal (operating as a web service) in order to mail you paper copies of your account statements.

Liberty's Identity Web Services Framework [ID-WSF] specifications define mechanisms and special terminology around this kind of activity. The employee portal is serving as an *identity-based service*, serving up identity information to those who have permission, and the 401K service is an *identity-consuming service* that makes use of this information. Both are web services, but the first one is known in Liberty terms as a *web service provider* or *WSP* because it provides the identity information, and the second is known as a *web service consumer* or *WSC* because it accepts and acts on that information. Liberty's Identity Service Interface Specifications [ID-SIS] specifications define standard semantics for, and operations on, commonly useful identity-based services, such as a personal profile service and a contact book service.

The following diagram illustrates these interactions and the specifications that define them, and provides a quick reference to the terminology. Note that this is a very high-level view; below we will go into more detail about actual protocol flows. (Also note that this paper leaves aside ID-WSF and ID-SIS details, and concentrates on human/application interactions.)



4. Inventory of SAML Components

Following are the major piece-parts defined by the SAML specifications, working roughly from the inside out. Note that SAML defines a number of schemas (using W3C XML Schema) and namespaces for many of these components.

- **Assertions:** As mentioned above, SAML assertions are the means of expressing and exchanging several kinds of information about subjects, in order to help secure their interactions with applications and manage their identity. SAML predefines three kinds of assertion statements: authentication, attribute, and authorization decision. You can also define your own.

- **Authentication context:** SAML defines ways to represent authentication information for more than two dozen common technologies, including simple passwords, Kerberos, X.509 public key, SSL, and others. You can also define your own. Authentication context information allows for a determination on the part of the relying party that stronger authentication is needed and that the subject must therefore be re-authenticated. Strong authentication is an important way to help combat identity theft. Since sign-on sometimes isn't “single” in these scenarios, the common initialism “SSO” is sometimes expanded to “simplified sign-on.”

- **Protocols:** SAML defines a number of XML-based request/response message pairs for obtaining assertions and doing network identity management. You can also define your own. The predefined protocols allow you to query for or directly request an assertion, ask for authentication of a subject, resolve an artifact into a whole protocol message (typically containing an assertion at its heart), synchronize the mutual understanding of a subject's identifier between service providers, map identifiers in a privacy-preserving way, and perform a “single logout” from several services at once.

- **Metadata:** SAML's metadata specification offers a way for IdPs and SPs to encode and present their essential configuration data.

- **Bindings:** SAML's protocol messages need to flow along paths already provided by distributed computing infrastructure in some fashion. To aid in this, SAML defines bindings to underlying transport and communications protocols. You can also define your own. The predefined bindings allow you to convey the messages by means of SOAP, PAOS (“reverse SOAP” -- a multi-stage SOAP/HTTP exchange that allows an HTTP client to send an HTTP request containing a SOAP response), HTTP redirect, HTTP form POST, HTTP URL query or form control (meant exclusively for artifacts), and HTTP URI resolution.

- **Profiles:** SAML is a flexible framework, so in order to achieve interoperability for common tasks, it defines a number of profiles that narrow down the options in support of particular use case scenarios. You can also define your own. In fact, a number of organizations outside of the SSTC have defined SAML profiles, the most notable being WS-Security as discussed above.

  New in SAML V2.0 are **attribute profiles**, which define how to correctly interpret SAML attribute statements using common attribute/directory technologies.

- **Operational modes:** If profiles are SAML's “minimum unit of interoperability,” operational modes are its “minimum unit of conformance.” These are named groupings of profiles that are required to be supported in order for an implementation to count itself as conforming. (SAML does not currently offer formal conformance testing and certification; however, the Liberty Alliance does host interoperability testing and formally certifies implementations as “interoperable.”) The operational modes are IdP, IdP Lite, SP, SP Lite, ECP (enhanced client or proxy), Authentication Authority, Attribute Authority, Authorization Decision Authority, and Requester.

5. Inside SAML Assertions

Since assertions are key to understanding how SAML lets you pass around identity information, let's dig a little deeper into them. (Note that while authentication and attribute statements are central to usage of SAML, authorization decision statements have been left out here because of their lesser importance to SAML going forward; their use is deprecated in favor of XACML.)

5.1. Assertion and Authentication Statement Structure

It was noted above that SAML assertions are compound structures that contain statements, and that SAML predefines statements about authentication, attributes, and authorization decisions for a subject. All statements within an assertion share a single description of the subject to which they apply, which you can think of as answering the question “who are we talking about?” They also share some assertion metadata, such as when the assertion was created, by whom, and how long it's good for.

The authentication statement helps to answer “how” and “when” the subject proved its identity to the satisfaction of the asserting party. In other words, it describes a *previous act of authentication* that can then be reused if so desired (for example, in single sign-on) -- it does not initiate any kind of credential-checking. Here's an example of a SAML assertion containing an authentication statement.

```
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    Version="2.0" IssueInstant="2005-01-31T12:00:00Z">
  <saml:Issuer>www.example.com</saml:Issuer>
  <saml:Subject>
    <saml:NameID
      Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress">
      j.doe@example.com</saml:NameID>
  </saml:Subject>
  <saml:Conditions NotBefore="2005-01-31T12:00:00Z"
    NotOnOrAfter="2005-01-31T12:30:00Z"></saml:Conditions>
  <saml:AuthnStatement AuthnInstant="2005-01-31T12:00:00Z" SessionIndex="0">
```

```
    <saml:AuthnContext>
      <saml:AuthnContextClassRef>
        urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
      </saml:AuthnContextClassRef>
    </saml:AuthnContext>
  </saml:AuthnStatement>
</saml:Assertion>
```

Note the SAML assertion namespace. SAML defines many namespaces for its various vocabularies; the namespace names change only at publication of major-version updates, while minor-version updates reuse existing namespace names. The SSTC strives to make minor-version schema changes compatible with previous schemas published in the same major-version series. The name of the issuer and the time of assertion issue are provided.

The subject (the “who”) is identified with the `Subject` element, in which a `NameID` element appears. (Alternatively, an `EncryptedNameID` element could be used, which is an important tool for for identifier confidentiality.) The name identifier is provided using the format of an email address; this format is indicated by a canonical URI on an attribute of `NameID`. SAML predefines a number of useful formats, and you can define your own.

A number of conditions might adhere to the usage of this assertion; the two most common are the `NotBefore` and `NotOnOrAfter` fields, which together make up what is known as the assertion's *validity period*. Making this period as short as possible helps prevent security breaches.

Looking at the authentication statement (`AuthnStatement`), it provides the “when” of authentication in the `AuthnInstant` attribute, and the how in the `AuthnContext` element. In this case, a particular authentication method, password-protected transport, is referenced by URI (this turns out to be the namespace name for the PasswordProtectedTransport authentication context class vocabulary defined by SAML).

5.2. Attribute Statement Structure

Attributes provide all kinds of “what” information -- the characteristics, traits, and preferences of the subject. At heart, no matter what technology is used on the back end to store it (though LDAP repositories are common), an attribute is a name/value pair. SAML's structure reflects this. Within a SAML attribute statement you can provide multiple such pairs. Here's a very simple example of a SAML attribute statement, with the surrounding assertion information presumed to be the same as the example above.

```
<saml:AttributeStatement>
  <saml:Attribute NameFormat=“http://smithco.com” Name=“PaidStatus”>
    <saml:AttributeValue>PaidUp</saml:AttributeValue>
  </saml:Attribute>
  <saml:Attribute NameFormat=“http://smithco.com” Name=“CreditLimit”>
    <saml:AttributeValue xsi:type=“smithco:type”>
      <smithco:amount currency=“USD”>500.00</smithco:amount>
    </saml:AttributeValue>
  </saml:Attribute>
</saml:AttributeStatement>
```

This attribute statement is not going to be very interoperable unless the company SmithCo somehow documents the attributes that are in its proprietary `http://smithco.com` name format. SAML predefines some name formats such as URI Reference and Basic, but also defines attribute profiles, which are combinations of attribute name formats and additional rules for interpreting the names provided. This allows for greater interoperability in using SAML along with classic attribute-handling technologies.

For example, SAML defines an X.500/LDAP attribute profile, for ease and interoperability of SAML usage with repositories using LDAP directory schemas. Following is an example of how to use this profile in a SAML `Attribute` element.

```
<saml:Attribute
  xmlns:x500="urn:oasis:names:tc:SAML:2.0:profiles:attribute:X500"
  NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
  Name="urn:oid:2.5.4.42"
  FriendlyName="givenName">
    <saml:AttributeValue xsi:type="xs:string"
      x500:Encoding="LDAP">Steven
    </saml:AttributeValue>
</saml:Attribute>
```

In this example, the `Name` of `urn:oid:2.5.4.42` is understood to refer to a “givenName” LDAP directory attribute.

6. Examining The Single Sign-On Use Case

Assertions provide the standard format needed to convey identity information, but protocols and higher-order information flows determine just how to ask for and send that information. As explained above, SAML profiles describe common patterns of SAML protocol (and protocol binding) usage in order to accomplish specific tasks. Let's take a look at the most common task: single sign-on.

SAML predefines two profiles that relate directly to single sign-on (various other profiles deal with single logout, which reverses the login process, and other identity management tasks). The plain-vanilla one is called the Web Browser SSO Profile; it assumes that a human user is using nothing more than a standard commercial browser. There is a more sophisticated one called the Enhanced Client or Proxy (ECP) Profile; it assumes a “smart” client device like a phone or PDA (or a proxy able to simulate such a device) that knows innately how to locate the appropriate IdP for its user and also knows how to use the PAOS (reverse SOAP) binding trick, whereby it can pass along SOAP response information that it has received, embedding it in HTTP requests. We'll focus on the plain-vanilla profile here.

Let's flesh out the employer/401(k) scenario that we alluded to several times above. Sam Smith, an employee of Pitch Tree Corp., has a user account at work through which he uses the company's internal portal, and he also has a 401(k) retirement account through his workplace that is managed by Dollarz Investments. The Pitch Tree internal website is where Sam logs in (gets authenticated and authorized for various tasks). Pitch Tree and Dollarz have a trust relationship that allows the latter to rely on assertions by the former about Sam's authenticity and attributes; this allows Dollarz not to have to manage a heavyweight user account for Sam, but rather to allow him secure access to its website based largely on identity information coming from his employer. So, using our SAML terminology, Pitch Tree is the identity provider (IdP) and Dollarz is the service provider (SP).

The Web Browser SSO Profile offers several mechanisms for allowing a user's authentication and attribute information to be shared among providers. The essential ingredients are as follows:

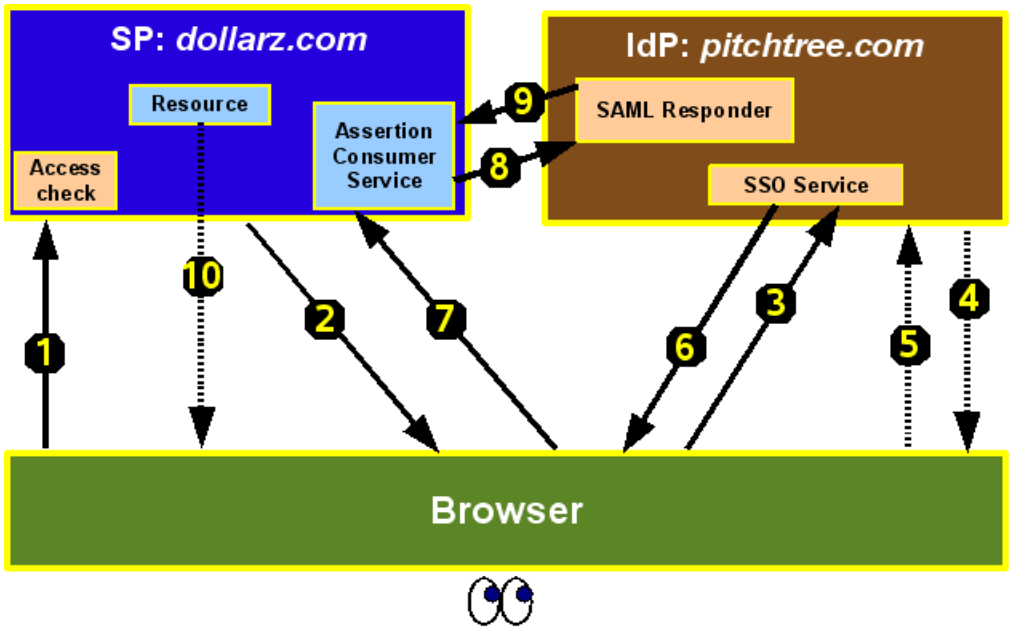
- The user has to authenticate at the IdP at some point, thus allowing the necessary SAML assertion to be created. (That's Sam, logging in to the Pitch Tree site.)
- The IdP has to convey the assertion to the SP. (That's Pitch Tree, sending an assertion about the facts of Sam's credential-checked login session -- possibly including some attributes about him -- to Dollarz.)

Beyond this very simplistic checklist, there are variations that depend on real-world circumstances. For example, sometimes it will be the case that Sam starts his online experience for the day by logging in to Pitch Tree's site, and only then travel over to the Dollarz site. But as often as not, he might go to the Dollarz site first, perhaps to look at stock market research over his morning coffee. So the flow will sometimes start with the IdP, and sometimes with the SP. The Web Browser SSO Profile accounts for both.

There are also a number of technical considerations having to do with the network architectures of the Pitch Tree and Dollarz sites and their technology preferences. For example, they might prefer to have identity information “pushed” through an HTTP POST or a redirect, or they might want to save initial bandwidth by using the SAML artifact construct, which is a very small string that you can use to “pull” the necessary identity information. All of these choices lead to eight different possible flows for the Web Browser SSO Profile:

- IdP-initiated (Sam starts out at Pitch Tree), where the needed assertion is conveyed as part of an unsolicited `Response` message:
  - The message is directly pushed using HTTP POST.
  - An artifact representing the message is sent, then the message itself is sent when “pulled” using the `ArtifactResolve` protocol.
- SP-initiated (Sam goes to Dollarz first), where the SP solicits the IdP for the needed assertion using the `AuthnRequest` protocol:
  - The SP uses one of three bindings -- HTTP POST, redirect, or artifact -- to send the authentication request.
  - The IdP uses one of two bindings -- HTTP POST or artifact -- to send the response message.

Let's focus in on just one of the eight options to give a taste of all these mechanisms: SP-initiated/POST/artifact. The following diagram and list explain the flow; dotted lines are steps that are outside SAML's control.



1. Sam attempts to access some protected area of the Dollarz website. We will assume he hasn't logged in here, so there is no current login session.
2. The Dollarz site sends an HTML form back to Sam's browser. It contains a SAML `AuthnRequest` characterizing Sam as the principal whose information is required.
3. The browser, either due to some action on the part of Sam or via an “auto-submit,” issues the POST to the Pitch Tree site's SSO Service.
4. If Pitch Tree doesn't already have a valid record of a recent-enough (and strong-enough) login by Sam, it challenges him for his credentials by means of the browser.

5. Sam logs in (we'll presume successfully for this example).

6. Pitch Tree's SSO service generates a SAML assertion describing Sam's authentication details and an artifact that can be used to pull a response message containing this assertion. The artifact is labeled with the “source identifier” of the Pitch Tree SAML responder. It could use redirection or an HTML form to send the artifact; we assume the latter here. The form control name is `SAMLart`.

7. The Dollarz Assertion Consumer Service gets the source identifier; because of its pre-existing trust relationship with Pitch Tree, is configured to understand this as a reference to the Pitch Tree SAML Responder.

8. The Dollarz Assertion Consumer Service goes ahead and contacts the Pitch Tree SAML Responder with an `ArtifactResolve` request containing the artifact it received.

9. Pitch Tree's SAML Responder now sends an `ArtifactResponse` message containing the needed assertion.

10. At this point, Dollarz has what it needs to decide whether to give Sam the protected Dollarz resource he wanted. Implementations vary, but typically it would set up a session for Sam as necessary, alert his browser to this fact by sending a redirection message with a cookie to it, and delivering the resource when asked to by the browser's HTTP GET.

This flow has quite a few moving parts, but Sam can be blissfully unaware of most of it. If you look past the binding choices and the question of which entity initiates the sequence, what adds complexity is the need to ensure confidentiality of the information so that only the intended audience has access to each message.

7. Achieving Privacy Through Identity Federation

Single sign-on is the simplest possible case of sharing identity information, allowing a one-way flow of information from its custodian to a service provider that can use it. But real life is a bit more complicated.

We've been talking about *federated identity* in the general sense of sharing or distributing identity information of various types across boundaries that have traditionally been impermeable. SAML, through its reliance on work first introduced by Liberty ID-FF, allows you to achieve this in several ways, perhaps most importantly by the means of *account linking* -- associating user's multiple accounts at multiple different providers so that the providers can communicate about the user in a way that ensures privacy. The phrase *identity federation* is usually used colloquially to refer specifically to account linking (though in technical terms it has a slightly broader meaning).

Let's map this to Sam Smith's scenario. Above it was mentioned that Dollarz need not have a heavyweight representation of Sam's identity, instead relying mostly on trusted “foreign sources” of this information such as Pitch Tree. In fact, using SSO, Dollarz need not strictly hold onto any representation of Sam's identity at all, but this is unrealistic in most scenarios -- particularly one involving something as important as Sam's retirement money and his relationship with Dollarz. In the likely case where Dollarz does keep a “native” account for Sam, it will be necessary to associate his IdP (Pitch Tree) account and his SP (Dollarz) account in some fashion.

If Pitch Tree and Dollarz exchange Sam's usernames and passwords specific to each account in order to link them, they will be violating his privacy in a most serious fashion. However, if SAML is used to mediate this account linking, it ensures his privacy by allowing for an agreement between Pitch Tree and Dollarz on a pseudonym that they will both internally map to their representation of Sam. Roughly, here is the sequence that will be used:

1. Sam logs in to Pitch Tree.

2. Sam logs in to Dollarz. (Actually, he could log in to Dollarz first and Pitch Tree second; the order doesn't matter.)

3. Sam requests or agrees to have both of his accounts linked. (This is important; Sam needs to consent to the linking in the first place in order to make this privacy picture complete.)

4. Pitch Tree and Dollarz engage in a SAML protocol to set up a NameID of the persistent-pseudonym type, which they can use to refer to Sam without revealing his details to each other.

The combined use of this pseudonym and adequate security mechanisms in its transport (for example, encryption) is what makes SAML Version 2.0 (as was its predecessor, Liberty ID-FF) the strongest and most complete identity standard for privacy preservation available.

8. Beyond Technical Specifications

Technical specifications are not often easy to understand on a quick read, but it should be remembered that they serve as a special kind of writing technology themselves. They are designed, first and foremost, to support interoperability of dissimilar implementations rather than a conceptual understanding of the "big picture." The SAML and Liberty specifications are no different. However, both are accompanied by a considerable body of supporting materials and programs. (They are also widely supported in identity management products, so that in many cases, you need not write any code of your own nor gain a deep understanding of the standards before deploying your solution.)

For example, the OASIS SSTC offers documents such as an Executive Overview [SAMLExec] and a Technical Overview [SAMLTech] to address the concerns and questions of different audiences. The latter is particularly interesting because it lays out the protocol flow options for all of the common identity management scenarios and many of the not-so-common ones, allowing you to match your needs against existing profiles and determine whether you need to invent a new one. The SSTC also makes available some presentation material and links to interesting third-party resources on its website [SSTC]. Finally, OASIS runs a mailing list called **saml-dev** where development issues can be discussed; see [SSTC] for information on this list.

One of the Liberty Alliance's specialties, as already mentioned, is to produce technical, legal, and business guidelines; it also publishes case studies and other helpful resources. Such documents produced so far include, for example, a Circle of Trust Legal Framework, Implementation Guidelines, Business Guidelines for Mobile Deployments, and Privacy and Security Best Practices. Liberty also regularly hosts deployment workshops at which you can learn from people who have deployed Liberty-enabled systems. Finally, Liberty's interoperability testing and conformance program produces results that can help you in your own purchasing and deployment decisions around identity management products. Information on all of these can be found at [Liberty].

Acknowledgements

The author would like to thank her colleagues at Sun, particularly Gerry Beuchelt, Tim Bray, Hellmuth Broda, Hubert Le Van Gong, Pat Patterson, Yvonne Wilson, Robin Wilton, and Lauren Wood, for their assistance and support in preparing this paper.

Bibliography

[ID-FF] *Liberty Identity Federation Framework (ID-FF) Version 1.2* set of specifications, Liberty Alliance standard. See http://www.projectliberty.org/resources/specifications.php .

[ID-SIS] *Liberty Identity Service Interface Specifications (ID-SIS) Version 1.0* set of specifications, Liberty Alliance draft standards. See http://www.projectliberty.org/resources/specifications.php .

[ID-WSF] *Liberty Identity Web Services Framework (ID-WSF) Version 2.0* set of specifications, Liberty Alliance draft standard, and *Liberty ID-WSF Data Services Template (DST) Version 2.0*, Liberty Alliance standard. See http://www.projectliberty.org/resources/specifications.php .

[Liberty] Liberty Alliance Project. See http://www.projectliberty.org/ . Guidelines can be found in the Resources area. Conformance testing information can be found in the Activities area.

[LibGloss] *Liberty Technical Glossary Version 1.4*, Liberty Alliance supporting document. See http://www.projectliberty.org/specs/liberty-glossary-v1.4.pdf .

[OpenSAML] OpenSAML Open Source SAML implementation. See http://www.opensaml.org/ .

[SAML] *Security Assertion Markup Language (SAML) Version 2.0* set of specifications, OASIS Standard, 15 March 2005. See http://www.oasis-open.org/specs/index.php#samlv2.0 .

[SAMLExec] *SAML V2.0 Executive Overview*, OASIS SSTC Commitee Draft, 12 April 2005. See http://www.oasis-open.org/committees/download.php/13525/sstc-saml-exec-overview-2.0-cd-01-2col.pdf .

[SAMLGloss] *Glossary for the Security Assertion Markup Language (SAML) V2.0*, OASIS Standard specification, 15 March 2005. See http://docs.oasis-open.org/security/saml/v2.0/saml-glossary-2.0-os.pdf .

[SAMLTech] *Security Assertion Markup Language (SAML) V2.0 Technical Overview*, OASIS SSTC Working Draft, currently at revision 07 on 13 July 2005. See the SSTC website for a link to the latest revision.

[SPML] *Service Provisioning Markup Language (SPML) Version 1.0* set of specifications, OASIS Standard, October 2003. See http://www.oasis-open.org/specs/index.php#spmlv1.0 .

[STP] *Web Services Security: SAML Token Profile*, OASIS Standard, 1 December 2004. See http://www.oasis-open.org/specs/index.php#wssprofilesv1.0 .

[Shib] Shibboleth project of Internet2/MACE. See http://shibboleth.internet2.edu/ .

[SSTC] Security Services Technical Committee of OASIS. See http://www.oasis-open.org/committees/security/ .

[WS-Security] *Web Services Security: SOAP Message Security 1.0 (WS-Security 2004)* and related specifications, OASIS Standard, March 2004. See http://www.oasis-open.org/specs/index.php#wssv1.0 .

[XACML] *eXtensible Access Control Markup Language (XACML) Version 2.0* set of specifications, OASIS Standard, 1 February 2005. See http://www.oasis-open.org/specs/index.php#xacmlv2.0 .

[XKMS] *XML Key Management Specification (XKMS 2.0)* and related specifications, W3C Recommendation, 28 June 2005. See http://www.w3.org/TR/xkms2/ [http://www.oasis-open.org/specs/index.php#xacmlv2.0] .

[XML-Enc] *XML Encryption Syntax and Processing* and related specifications, W3C Recommendation, 10 December 2002. See http://www.w3.org/TR/xmlenc-core/ .

[XML-Sig] *XML-Signature Syntax and Processing* and related specifications, W3C Recommendation, 12 February 2002. See http://www.w3.org/TR/xmldsig-core/ .

Biography

Eve **Maler**
Technology Director
Sun Microsystems, Inc. [http://www.sun.com]
Chief Technologist's Office
Bellevue
Washington
98007
United States of America

Eve Maler is a Technology Director at Sun Microsystems, developing interoperability strategies and leading partner engagements related to web services, security, and identity. Eve was one of the inventors of XML. She has also variously made major leadership, technical, and educational contributions to other successful standards, such as the Security Assertion Markup Language (SAML), the Liberty Alliance specifications, the Universal Business Language (UBL), and DocBook.