JSAML Toolkit

Netegrity's Java implementation of the Security Assertions Markup Language (SAML) specification

Netegrity White Paper

PUBLISHED: SEPTEMBER 28, 2001

Copyright © 2001 Netegrity, Inc. All Rights Reserved.

Netegrity and SiteMinder are U.S. registered trademarks of Netegrity Inc.
All other brand or product names are trademarks or registered trademarks of their respective companies.

Executive Summary

JSAML is Netegrity's Java implementation of SAML, the Security Assertions Markup Language.

SAML defines an eXtensible Markup Language (XML) framework for exchanging security information between business partners over the Internet.

SAML is being standardized at OASIS, the Organization for the Advancement of Structured Information Standards, an international consortium that creates interoperable industry specifications based on XML.

JSAML is a standards-based toolkit designed for developers to build secure solutions for:

- distinct business partners who exchange profile and entitlement information over the Internet,
- single-sign on between vertical applications (such as SAP, Oracle, and Peoplesoft) and enterprise infrastructures.

JSAML delivers the following benefits:

- **Self-contained security package** – JSAML allows developers to build browser-based single sign-on and XML messaging solutions without the use of any other proprietary products.
- **Standards-based toolkit** – JSAML is based on Java and uses standard, widely available cryptographic libraries and transport-level security packages.
- **Flexible developer solution** – JSAML provides the source code for example solutions that developers can easily modify to adapt to their specific environment requirements.
- **Available at no cost** – The JSAML toolkit is freely downloadable from Netegrity's Web site (www.netegrity.com).

The JSAML toolkit includes executables packaged in a Java Archive (JAR) file together with example source code that can be modified by the developer to accommodate a particular security application.

The JSAML toolkit is not designed to perform tasks usually done by full-fledged security engines, such as:

- Out-of-the-box integration with industry-standard user directories
- User administration
- Distributed access-policy management
- Session management
- High performance and scalability (load balancing and failover)

Those tasks will be supported by the JSAML-based products recently announced by Netegrity, viz.,

- AffiliateMinder (browser-based interaction between partners in an e-business network),
- TransactionMinder (securing the documents used in Web services and other business-to-business XML message exchange).

Netegrity provides JSAML users with a moderated forum on Netegrity's Web site where developers can exchange information and benefit from Netegrity's expert advice.

SAML Backgrounder

Introduction

In December 2000, Netegrity created an OASIS industry-wide Technical Committee (TC) called Security Services (www.oasis-open.org/committees/security), which is responsible for submitting a draft specification of SAML to the OASIS Board members in the second half of 2001.

SAML is an open, standard framework for sharing security information on the Internet through XML documents. SAML's scope is based on three use cases (implemented as examples in Netegrity's JSAML toolkit):

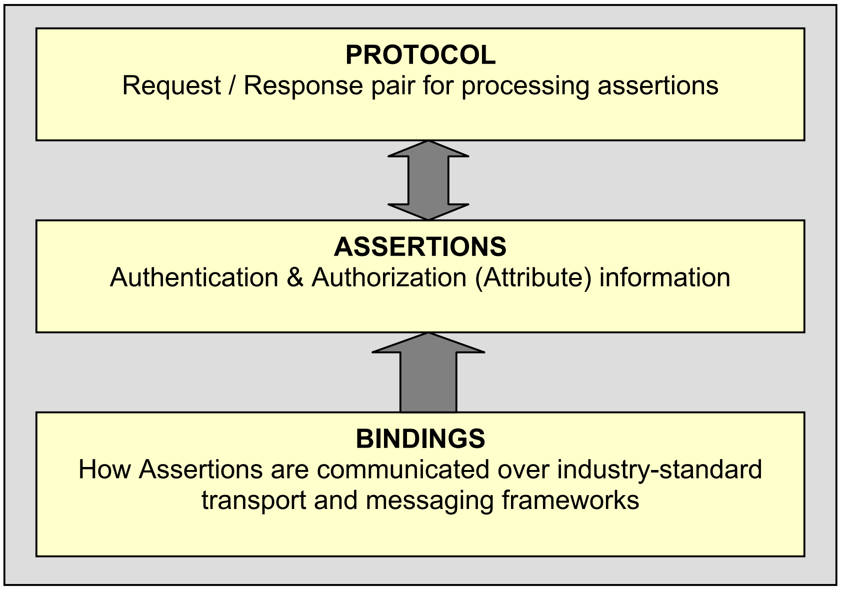
- Browser-driven interaction
- XML message transfer
- Remote authorization

SAML presents several advantages over proprietary solutions.

- In a browser-based single sign-on environment, no user directory duplication or synchronization is necessary. Security information (in the form of SAML assertions) "travels" with users. As a result, users coming from a source Web site do not have to be registered at the destination Web site. User information does not have to be duplicated at each site involved in an e-business network.
- In an XML message transfer environment, SAML provides attribute-based authorization that goes significantly beyond authentication based upon XML digital signatures.
- In a remote authorization environment, SAML supports a scalable "hub-and-spoke" security model which eliminates the requirements for a point-to-point solution. The same language is used by many services to many enterprises.

SAML Architecture

The SAML specification includes three distinct parts: Assertions, Protocol, Bindings.



PROTOCOL
Request / Response pair for processing assertions

ASSERTIONS
Authentication & Authorization (Attribute) information

BINDINGS
How Assertions are communicated over industry-standard transport and messaging frameworks

SAML Assertion Types

SAML assertions are encoded in an XML schema. Assertions can be digitally signed (XML-DSIG).

- *Authentication Assertion*
  An authentication assertion is issued by an authentication authority upon successful authentication of a subject. It defines the issuer, authenticated subject, time of issuance, validity interval, and other authentication-related attributes.

- *Attribute Assertion*
  An attribute assertion is issued by an attribute authority, based on policies. It may be used to describe the "entitlements" of a subject.

- *Authorization Decision Assertion*
  An authorization decision assertion is issued by an authorization authority in response to a request for access to a protected resource by an authenticated subject.

As is the case with Netegrity, authentication, attribute, and authorization decision authorities may be hosted by the same product (e.g., SiteMinder).

SAML Protocol

The SAML Protocol is encoded in an XML schema as a set of request-response pairs. It defines the interactions between:

- a policy-enforcement point (PEP) and a policy-decision point (PDP),
- an authentication authority and a client program,
- an attribute authority and a client program.

A SAML request may include authentication, attribute, and authorization queries. All types of SAML requests are met with a common SAML response.

SAML Bindings

The SAML Protocol Bindings specify how SAML request-response message exchanges are mapped to standard messaging protocols. The SAML specification currently provides for a SAML HyperText Transport Protocol (HTTP) Binding and a SAML Simple Object Access Protocol (SOAP) Binding.

SAML Profiles specify how SAML assertions are inserted in, and extracted from, a message framework or protocol. Currently, the SAML specification includes a Web Browser Profile and a SOAP Profile.

Introducing Netegrity's JSAML Toolkit

The purpose of the JSAML Toolkit is to

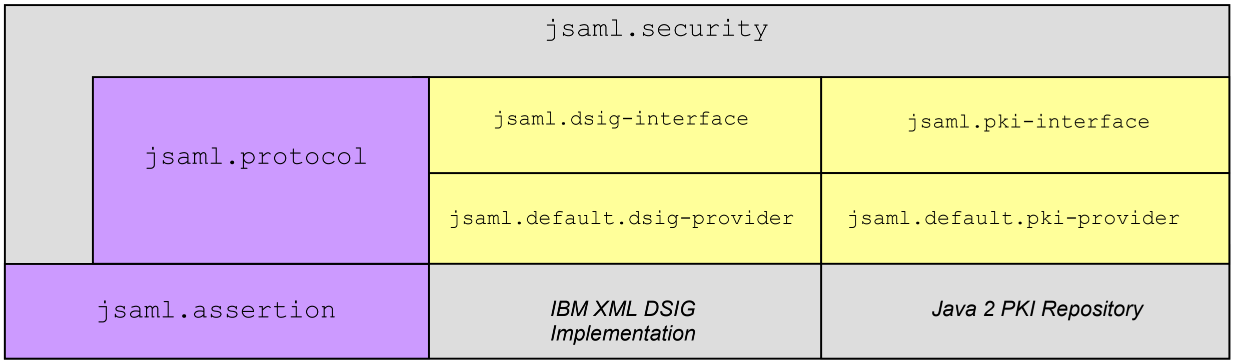
- implement every feature defined in the SAML specification,
- allow developers to modify the source code provided for usage examples to meet their own requirements.

With JSAML, developers can build actual security solutions based on the SAML standard.

The JSAML Toolkit is downloadable from Netegrity's Web Site and includes the following components:

1. Java library of objects implementing the SAML specification based on the following OASIS reference XML Schemas:
   - **`draft-sstc-schema-assertions-16`**
   - **`draft-sstc-schema-protocol-16`**
2. Source code for the implementation of the examples covering all three SAML uses cases
   - Browser Single Sign-On between e-business network partners
   - XML message transfer with SAML credentials
   - Remote authorization using the SAML Protocol
3. Documentation
   - JSAML Installation Guide
   - SAML Primer
   - JSAML Documentation (JavaDoc)
   - JSAML Examples Guide
4. Open Software
   - Apache XML Java Parser (Xerces-J)
   - Apache SOAP 2.2 (necessary for the examples implementation)

JSAML Libraries



`jsaml.assertion`: Produce and consume SAML assertions
`jsaml.protocol`: Produce and consume SAML requests and SAML responses
`jsaml.pki-interface`: Interface for key management library
`jsaml.default-pki-provider`: Standard wrapper for Java key management tools
`jsaml.dsig-interface`: Interface for digital signing library
`jsaml.default-dsig-provider`: Standard wrapper for IBM DSIG library
`jsaml.security`: Use configuration information, key management, and signing to verify and validate SAML assertions, generate SAML artifacts, and any other required cryptographic operation

Required Third-Party Software

The following components must be downloaded from the Internet by the JSAMLToolkit users.

- Sun's Java Cryptography Extension (http://java.sun.com/products/jce/)
  JCE provides a framework and implementations for encryption, key generation and key agreement, and Message Authentication Code (MAC) algorithms.

- Sun's Java Secure Socket Extension (http://java.sun.com/products/jsse/)
  JSSE provides for the secure transfer of data between a client and a server using any application protocol (HTTP, FTP, etc.) over TCP/IP. JSSE is a Java implementation of SSL (Secure Sockets Layer) and TLS (Transport Layer Security) protocols and includes functionality for data encryption, server authentication, message integrity, and optional client authentication.

- IBM XML Security Suite (http://www.alphaworks.ibm.com/)
  The IBM XML Security Suite (xss4j-20010420.zip) provides security features that go beyond transport-level security protocol (e.g., SSL), including digital signature, element-wise encryption, and access control.

- Apache Tomcat (http://jakarta.apache.org/): Java Servlet/JSP container installable in Microsoft's IIS or Netscape's iPlanet, for implementing the use case examples.

Note: JSAML requires Sun's Java Development Kit (JDK) V1.3.

Implementation Examples

The source code for implementation examples is included a Web Archive (WAR) file. It allows developers to make modifications based on the requirements of their specific environment.

Browser-Based Interaction (SSO)

M&M Consulting's portal site allows its subscribers to view contents from several content providers' sites. JSAML allows subscribers to navigate between M&M Consulting and content providers in a single sign-on environment to access protected resources.

In this scenario, M&M Consulting (source site) generates and sends SAML assertions to the Content Provider (destination site).

The M&M Consulting site provides

- a Login page,
- Contents, a servlet that displays content to the authenticated user,
- Forward, a servlet that transmits assertions to the Content Provider site.

Two SAML assertions are generated:

- Authentication assertion, which describes subject and authentication at M&M Consulting,
- Attribute assertion, which describes profile information associated with the subject.

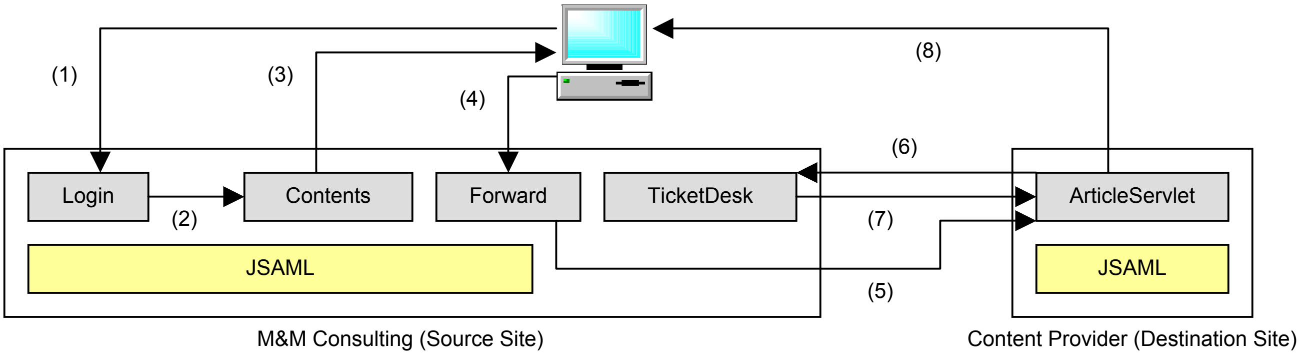
These assertions are transmitted by adding SAML artifacts to a URL and providing a stateful service (implemented by the TicketDesk servlet) for serving up the SAML assertions to the Content Provider site.

Note: SAML artifacts are defined in the Bindings section of the SAML Specification (Web Browser Profile for SAML). A SAML artifact is a small-size, random number designed to point to full SAML assertions. SAML artifacts are passed between sites by the browser on URL query strings. Assertions are "pulled" by the destination site from the source site using the SAML artifact information.

M&M Consulting uses Apache Tomcat container-managed security. Information about users together with their password is held in an XML file which plays the same role as a bona fide user directory (e.g., LDAP, RDBMS) in a production environment.

The Content Provider entries (i.e., the contents that are offered to M&M Consulting subscribers) are displayed at the source site as links. When the user clicks on a link, the content's URL is accessed together with the SAML artifact parameters, as part of the URL query string.

At the Content Provider site, the ArticleServlet servlet removes the SAML artifacts from the URL line, and based on the partner configuration file, pulls the SAML assertions from M&M Consulting, and then checks the SAML assertions for validity.



(1) The user (an M&M Consulting subscriber) submits a log-in page with a customer ID (username) and password to the M&M Consulting site.
(2) Login provides a session ID cookie and redirects to the Contents servlet.
(3) The Contents servlet checks for the session cookie, and creates the Contents page from a template and a list of URLs, and then returns the Contents page to the user.
(4) The user clicks on the Contents page link, and submits the request to the Forward servlet.
(5) The Forward servlet creates the SAML assertion, generates the SAML artifact for that assertion and forwards the SAML artifact to the Content Provider Site.
(6) The ArticleServlet servlet calls back with the SAML artifact.
(7) The TicketDesk servlet provides the full SAML assertion to ArticleServlet.
(8) ArticleServlet processes the SAML assertion and then returns the requested resource to the user.

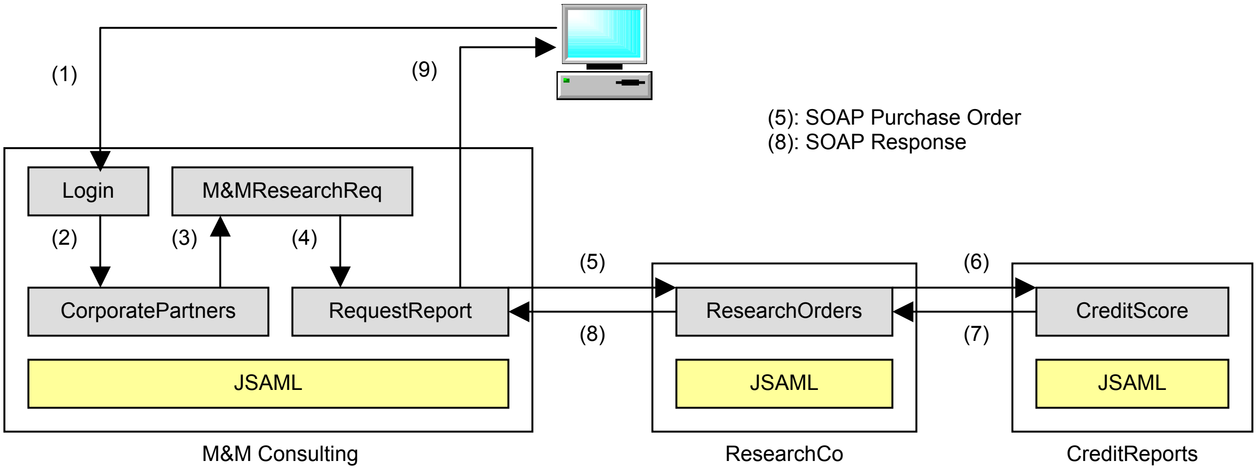
XML (SOAP) Messaging

LeDepot is a wholesaler that logs in to M&M Consulting's portal to get information on competing vendors. M&M Consulting uses SOAP messaging to invoke a Web service at ResearchCo.

This scenario begins with the M&M Consulting Login page. A form for LeDepot user's request is created and posted into the RequestReport servlet at M&M Consulting, which creates a purchase order placed as a payload in a SOAP message. The SAML attribute assertion is inserted into the SOAP envelope (the SAML attribute assertion must be digitally signed by M&M consulting and it must use the public key technique for assertion attachment integrity).

The RequestReport servlet POSTs into an HTTPS (server-side certificate) URL bound to the ResearchOrders servlet at ResearchCo.

ResearchOrders receives the SOAP message including the purchase order from M&M Consulting, and validates the attribute assertion.



(1) The user (a LeDepot employee) submits a log-in page to M&M Consulting.
(2) After logging in, the user selects the CorporatePartners page.
(3) The user selects ResearchCo from the CorporatePartners pick list (M&M Consulting syndicates Web services from several providers).
(4) The user fills out the request form provided by M&MResearchReq and submits the form to RequestReport.
(5) RequestReport uses the submitted information to create an XML document (a purchase order for a research report), inserts SAML assertions (the information gathered at log-in time) in the SOAP message envelope, and sends the SOAP message to ResearchCo.

Remote Authorization

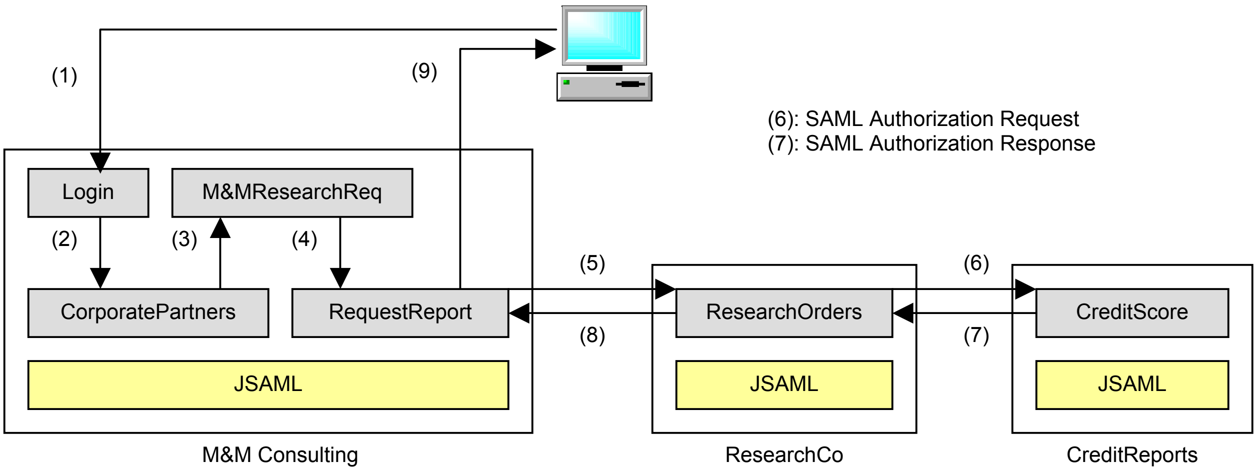
Before meeting M&M Consulting's request, ResearchCo calls out CreditReports, a credit-rating company, to make sure LeDepot can pay for the research report (remote authorization).

In this scenario, the ResearchCo site creates a SAML authorization request message using as evidence the attribute assertion contained in the SOAP message.

The authorization request is sent to the CreditScore servlet at the CreditReports site over server-side HTTPS.

The CreditReports site checks for validity the signature on the request message as well as the assertion sent as evidence.

Using the attributes found in the assertion, the CreditScore servlet calculates credit rating information, which it returns to ResearchCo in a SAML response.



(6) ResearchOrders reads the SOAP message and processes the SAML assertions. Based on program logic, ResearchCo decides to send a SAML request for information on LeDepot to CreditReports.
(7) CreditScore reads the SAML authorization query, processes it, and sends a (positive) SAML authorization response back to ResearchOrders at ResearchCo.
(8) ResearchOrders returns to RequestReport the information originally requested by LeDepot (using a SOAP message).
(9) The report is made available to the LeDepot user.

Conclusion

Netegrity is the first vendor to provide an implementation of SAML based on the current specification of the standard.

Thanks to JSAML, developers are able to design secure single sign-on and message exchange solutions that work with other SAML-compliant environments outside their enterprise. Likewise, JSAML allows software vendors to SAML-enable their applications and provide single sign-on to end-users in heterogeneous environments.

Netegrity will use JSAML in forthcoming products to provide full-fledged solutions for browser-based e-networks and secure Web services.