Tivoli. software

IBM

Redpaper

Axel Buecker
Paul Ashley
Neil Readshaw

Federated Identity and Trust Management

Introduction

The cost of managing the life cycle of user identities is very high. Most organizations have to manage employee, business partner, and customer identities. The relationship between the business and these individuals can change frequently, and each change requires an administrative action. For the individuals also the situation is unsatisfactory, because they need to create separate accounts at all the businesses that they access.

A *federation* is defined as a group of two or more business partners who work together. This federation can be formed to provide a better experience for their mutual customers, to reduce identity management costs, or both. For example, a financial institution might want to provide seamless access for their high-value clients to financial market information provided by a third-party research firm. Government departments might want to collaborate to provide a single citizen login for their government services. A small online store might not want to manage large numbers of customer records and instead prefer to partner with a financial institution to provide that service. In all of these cases, the businesses need to work together to create a *business federation*.

Business federations are built on *trust relationships*. These trust relationships are created using out-of-band business and legal agreements between the federation participants. These agreements must be in place before a federation can begin to operate.[1]

After the business and legal agreements are in place, these partners can begin to operate together using technology that supports the federation arrangements. That is, the technology provides the federation and trust management capabilities, cryptography support, and protocol implementations that allow a secure partnership to operate in the Internet environment.

To manage the identities across the federation, *federated identity management* provides a standardized system for simplifying identity management across company boundaries. This system allows organizations to off load identity and access management costs to business partners within the federation. This functionality enables a business to receive trusted

[1] With the new user-centric identity protocols discussed later in this IBM® Redpaper, more loosely coupled partnerships are also becoming possible.

information about a customer without registering that customer and without the customer logging in and being required to provide identity information again.

Another important consideration is the use of *Web services*. Web services have emerged to address cross-enterprise, cross-platform, and cross-vendor business integration issues. Web services are a family of technologies that enable easy interoperability of information technology (IT) services and integration of applications into a company's broader business processes. Web services technology enables companies to describe available services and provide access to those services over standard Internet protocols.

In this IBM Redpaper we discuss the IBM Tivoli® software solution for a consistent, unified, and evolutionary approach to securing cross-enterprise e-business environments. Built on open security standards, with tight integration to Web middleware (Java™ 2 Platform Enterprise Edition (J2EE™) and Microsoft® .NET), Tivoli security solutions allow you to increase the reach of your business. They build on existing Web security investments that can quickly evolve to take advantage of Web services and federation standards.

The Tivoli software technical strategy is multi-pronged:

- Support open standards to enable secure cross-enterprise, cross-platform, cross-vendor integration.
- Offer enhanced product support for Web services identity service mechanisms, with seamless integration with a variety of Web services development and deployment platforms (for example, IBM WebSphere®, Microsoft .NET, and so on).
- Use an evolutionary approach that builds on existing identity management, extranet access management, and trust management.
- Enable an entire new class of secure e-commerce business services building on federated identity management.

To put the technical strategy in context, this Redpaper highlights the required functional capabilities and benefits of federations and the security standards that apply. It also highlights how IBM Tivoli Federated Identity Manager is used to enable business federations by providing standards based security approaches for Web, Web services, and service-oriented architecture (SOA) environments.

Customer scenarios

In this section we describe customer examples where federated identity and trust management are the enablers for new business initiatives. Technical solutions for these scenarios are described later in this paper.

Public sector: Single sign-on for government agency services

In this scenario a government agency wanted its citizens to have a single login to all of the government services on the Internet and to be able to access services across the various departments seamlessly. This single login improves a convenient experience for users, motivates them to use online transactions, and reduces the operational costs to transact within department branches. The government agency wanted to achieve the cost savings from online transactions versus physical transactions and also wanted to reduce its escalating identity management costs due to duplicate identity management processes at all departments.

Telecommunications industry: Making services available to external partners

In this scenario a telecommunications provider wanted to make new services available to external partners in a secure manner. The telecommunication provider expanded its business by providing value-added services from external partners. These new services, accessible by way of standards-based Web services, are available only to trusted partners. Additionally, there is a requirement to accept requests from partners with differing IT capabilities and standards preferences. It is also important to minimize the changes that are required in the internal applications to accommodate these various partners.

Financial industry: Re-purposing of existing IT systems

In this scenario, an organization in the financial services industry wanted to reuse the business logic in its existing mainframe CICS® applications from a portal. This re-use allows new business partners to access the applications, increasing revenue and providing new growth opportunities. The organization chose an SOA because of the flexibility that it provided in integrating existing IT resources and potential service reuse opportunities.

Federated identity management

One of the key requirements when establishing a business federation is to manage identities throughout the federation. This process is commonly called *federated identity management* and is necessary for establishing secure and cost-effective business collaboration with an auditable trail of who is connecting to the target application and resources. This process allows organizations to off load identity management costs to business partners within the federation.

Federated identity management provides a direct benefit to the users of the business federation as well. User are required only to remember their credentials in order to identify and authenticate themselves to their own organization. Authentication to other organizations in the federation can now be a seamless operation without the need for direct user interaction. This simple authentication provides a more efficient user experience by reducing the number of credentials that users are required to remember, and the number of times that they are required to supply credentials to get access to services.

Roles in a federation

The important roles in any federation are the *identity provider* and *service provider*, as shown in Figure 1 on page 3.

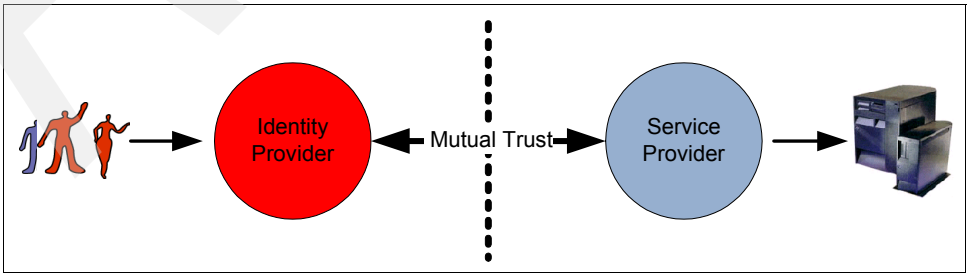

Figure 1 Business federation with identity and service providers

Let us take a closer look at those roles.

Identity provider

The identity provider (sometimes called the *account partner*) is the “vouch for” party in an identity federation. That is, it gives assurances of the identity of the user to the other party.

The identity provider is responsible for:

- Managing users and their identities
- Issuing credentials
- Handling user administration
- Authenticating the user
- Vouching for the user's identity with the service provider

Service provider

The service provider (sometimes called the *relying party* or resource partner or consumer) is the “validating party” in a transaction.

The service provider is responsible for:

- Controlling access to services
- Validating the asserted identity information from the identity provider (typically by way of verifying a digital signature)
- Providing access based on asserted identity
- Managing only locally relevant user attributes, not an entire user profile

For a particular set of business scenarios, an organization might act as both an identity provider and a service provider. For example, where government agencies access each other's applications, each agency plays the role of identity provider or service provider, depending on which organization is providing the services and applications in a given case.

Trust management

Trust is the expression between parties that one party to a relationship agrees to believe statements (also called *claims*) made by another party. Trust can be based on a combination of past history, experience, and risk tolerance.

Trust management addresses relationships between entities within organizations, security domains, and systems. These relationships can be system-to-system, business-to-business, and so on. Trust management deals with two aspects: business and technology. The business aspect deals with two entities agreeing upon a set of rules to conduct business. These rules include relationship management, liability management, and other legal and contractual aspects.

Business processes and policies are necessary for establishing trusted relationships. These processes might include choosing a legal procedure to follow and the process used for evaluating liability. This process might also include the policies specific to resource access. These policies often already exist as part of the business arrangement between the enterprise and its business partner.

The technology aspect of trust management deals with managing the infrastructure that supports the capability for establishing trust by cryptographic methods. These include key management (strength, key validation, and so on) protocols, attributes, and other technical considerations for establishing trust.

In the context of federated identity management, trust management capabilities should be considered as part of the IT infrastructure, not as part of application business logic. Trust infrastructure provides mechanisms such as an identity service for validating, transforming, and issuing security tokens that represent identities for users or Web services. Consistency, flexibility, and reduced time to market are facilitated by using a common trust infrastructure that is integrated with the various systems, applications, and platforms in an environment. In this paper we describe how IBM Tivoli Federated Identity Manager can provide a federation and trust infrastructure for seamless business to business and business to consumer collaborations.

Federation protocols

One of the issues that we address in this paper is the common requirement for users to have different authentication credentials (for example, user name and password) for each Web site they visit. Users are forced to enroll and create credentials at each Web site requiring authentication and are then required to remember all of these user names and passwords. The users also have to update these passwords and other identity data, as required by each of the Web sites. For sites that require stronger authentication, such as using a time or challenge response based token, the users will face the additional issue of key chains with growing sets of physical tokens.

Additionally, for each of the enterprises there are costly identity management issues around managing these user accounts. These costly issues are particularly burdensome for small and medium businesses that might have millions of occasionally used accounts. An example is the cost associated with password resets.

To overcome these issues, federation protocols have been designed to allow a user to sign-on once to a federation of cooperating Web sites. In this section we describe two types of Web single sign-on protocols. The first type focuses on enterprise-centric models and includes *Security Assertions Markup Language (SAML)*, *Liberty,* and *Web services (WS) Federation* specifications. These federation protocol specifications have been in existence for some years and are widely adopted. The second type is the user-centric identity schemes. These are newer specifications that focus on giving users more direct control over their digital identities and are beginning to be adopted.

The trust relationship is different with these various types of models. With enterprise-centric models, the trust relationship is always between two or more companies who are the business partners involved in the federation. In user-centric models that arrangement is still possible. However, with the user-centric model, self-issued credentials are also possible and can be used as a password replacement to alleviate some of the problems of user identity management for Web sites.

Federated single sign-on

Federated single sign-on is used in many browser-based scenarios. A user (the consumer) is given the ability to authenticate to a Web site (the identity provider) and then access other Web sites (the service provider) without the need for the user to authenticate again. The service providers trust the identity provider to authenticate the user and accept security tokens that are issued by the identity provider on the user's behalf. Figure 2 shows one example of this process, where the SAML browser artifact profile is used to allow the user single sign-on from Web site A to Web site B. In the figure, IBM Tivoli Federated Identity Manager is used at both the identity provider and service provider. However, because of the standards-based approach, other products or open source implementations can be used.

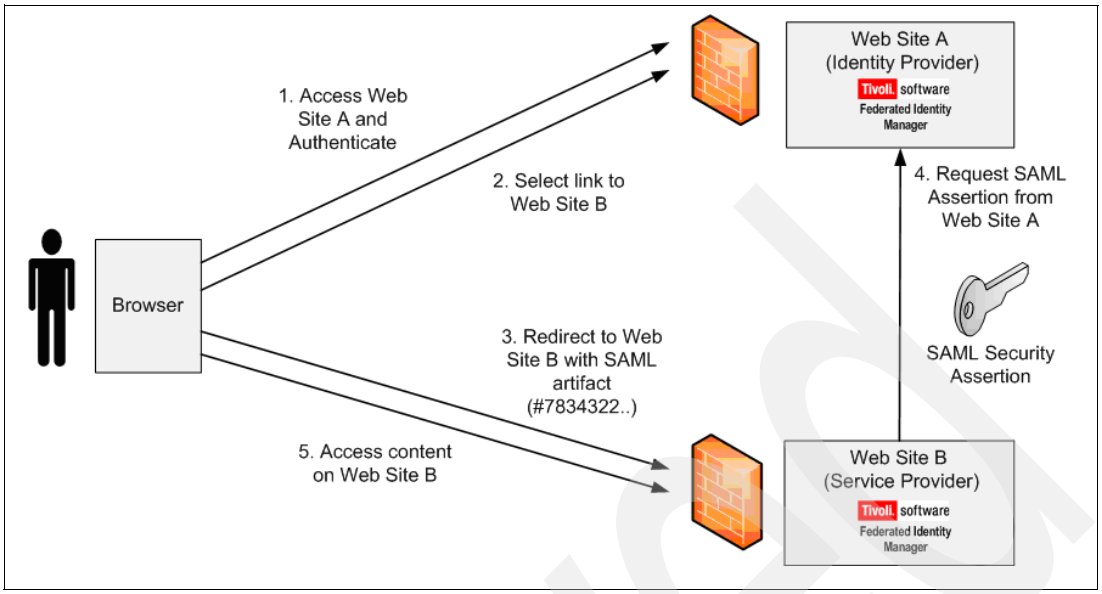

Figure 2 SAML Browser Artifact Profile for federated single sign-on

Federated single sign-on encompasses a number of identity functions including:

Single sign-on As shown in Figure 2, users authenticate to an identity provider and then access other service providers without needing to authenticate again.

Single sign-off For users performing a single log-off action, the system removes all active identity provider and service provider sessions for each user.

Account linking Linking of a user's identity provider and service provider accounts as a prerequisite for single sign-on.

Account de-linking De-coupling a user's identity provider and service provider accounts.

Alias management Enabling user access between identity providers and service providers to use aliases rather than the user's real identity.

Attribute request Service providers can request additional information about a user from an identity provider.

Federated single sign-on is, therefore, a particular use case of federated identity management focusing on managing identities across cooperating businesses accessed using the Web.

Industry standards are very important in the federated single sign-on scenario because the identity provider and service providers are commonly different companies with different IT environments. Hence a single vendor's proprietary solution is not suitable.

There are three popular industry standards supporting federated single sign-on. These are SAML, Liberty, and WS-Federation. IBM Tivoli Federated Identity Manager supports each of these specifications including different versions as shown in Table 1.

Table 1 IBM Tivoli Federated Identity Manager supported federated single sign-on standards

Protocol	Versions
SAML	1.0, 1.1, 2.0
Liberty	1.1, 1.2
WS_Federation	1.0

Security Assertions Markup Language

SAML is a specification that is designed to provide cross-vendor single sign-on interoperability. SAML was developed by a consortium of vendors (including IBM) under the auspices of OASIS, through the OASIS Security Services Technical Council (SSTC). SAML has two major components:

- It defines *SAML assertions* that describe security tokens representing users.
- It defines *SAML bindings and profiles* for a single sign-on protocol.

A SAML assertion is an XML-formatted token that is used to transfer user identity (and attribute) information from a user's identity provider to trusted service providers as part of the completion of a browser single sign-on request or Web services request. A SAML assertion thus provides a vendor-neutral means of transferring information among a federation of business partners. As such, SAML assertions have a lot of traction in the overall federation space.

As a protocol, SAML has three versions: SAML 1.0, 1.1, and 2.0. SAML 1.0 and SAML 1.1 (collectively, SAML 1.x) focus on single sign-on functionality. Using Liberty Identity Federation Framework (ID-FF) 1.2 as input, SAML 2.0 represents a major functional increase over SAML 1.x. SAML 2.0 now takes into account more of the identity life cycle functionality and addresses some of the privacy concerns associated with a federated environment.

More information about the SAML specification is available from:

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

Liberty

The Liberty Alliance Project is a group of vendors (including IBM) and user organizations aiming to deliver a federated network identity solution that enables sign-on for consumers and business users in an open, federated way.

The federated single sign-on protocols were developed as part of the Liberty Identity Federation Framework (ID-FF). There have been Liberty ID-FF 1.1 and 1.2 releases, with the 1.2 release being submitted to OASIS for input into SAML 2.0 (see above). Liberty 1.2 provided many identity life cycle functions not originally considered in SAML 1.0 and 1.1. Although Liberty ID-FF is still available to be used, because of the submission to OASIS, it can be considered to have been superseded by SAML 2.0.

More information about the Liberty Alliance Project is available from:

`http://www.projectliberty.org`

WS-Federation

WS-Federation was created by a group of vendors (including IBM) to provide extensions to the use of WS-Trust, mentioned in the next paragraph and in Table 2, to address the identity requirements of both Web applications and Web services. The intent is to provide a common method to support both browser-based applications and Web services-based applications. This commonality between browser and Web services is not a key consideration of either the

SAML or Liberty ID-FF single sign-on specifications. Additionally, WS-Federation is aligned closely with the rest of the WS-Security family of standards.

WS-Federation defines extensions to the WS-Trust Security Token Service to enable identity brokering, attribute request and retrieval, authentication and authorization claims between federation partners, and to privacy protect these claims.

There have been two releases of WS-Federation: 1.0 and 1.1.

User-centric identity

One criticism of the federated single sign-on systems just described is that they are *enterprise-centric* rather than *user-centric*. That is, the protocols are designed to meet the requirements of enterprises that need to federate with the control of information with the identity provider and service provider. Users have less control than is desired over what information is required by the identity and server providers, and often give over more information than is required for the type of transaction involved.

User-centric identity systems (sometimes referred to as *Identity 2.0*) are an attempt to put the control back into the user's hands. In this way a user gains consent as to what information about them is disclosed to which sites and for what purpose. One of the other benefits of user-centric identity systems is that they can be loosely coupled relationships. The relying party does not necessarily need to have a pre-existing trust relationship with the identity provider (be it a managed identity provider or self-issued). This arrangement allows for a non-password-based account bootstrapping process, which is easier for both users and relying parties.

Figure 3 on page 8 shows a typical flow in a user-centric system. In the figure, IBM Tivoli Federated Identity Manager is used at the identity provider and the relying party. However, because of the standards-based approach, other products and standards-based implementations can be used.

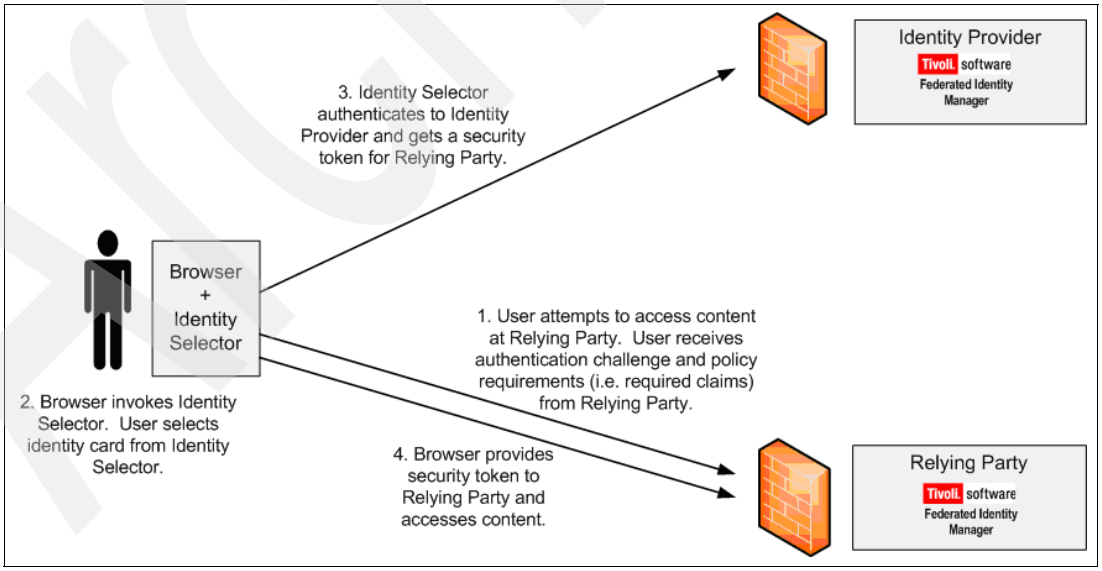

Figure 3 CardSpace for user-centric single sign-on

As shown in Table 2, IBM Tivoli Federated Identity Manager supports two families of user centric identity protocols OpenID[2] and information card profile using identity selectors, such as Microsoft Windows® CardSpace[3] and the Eclipse Higgins Project[4].

Table 2 IBM Tivoli Federated Identity Manager supported user centric single sign-on standards

Protocol	Versions
Identity selector (Microsoft Windows CardSpace)	1.0 (using WS-Trust 1.2)
Identity selector (Eclipse Higgins Project)	1.0
OpenID	Authentication protocol 1.1 Simple Registration Extension 1.0

Microsoft Windows CardSpace identity selector

Microsoft Windows CardSpace is an identity metasystem to enable users' management of their own digital identities. A new visual component on the user's desktop introduced by CardSpace is called an *identity selector*. An identity selector works in conjunction with a user's browser or other client-side application to broker authentication requests. The identity selector contains information cards that are visual representations of various user identities. These identities could be personal information cards (the card contains data asserted by the user without external validation) or managed information cards (where the user has an account at an identity provider).

In a typical CardSpace scenario, users access a site that provides a service (often called a *relying party*) with their browser. The relying party's authentication page contains an icon to indicate that it supports Information Cards. Upon clicking this icon, the relying party sends an authentication challenge to the user's browser. Included in this challenge is a determination of which attributes of a user's identity are required (these required attributes are sometimes called *claims*). The identity selector is displayed on the user's desktop, asking the user to choose which information card to use to authenticate to the relying party. The set of required and optional claims that are requested by the relying party is displayed so that the user is aware of what information is being requested. Additionally, the user has control over which information card to use and whether optional attributes are provided. After an information card is chosen (in the case of a managed information card), users then authenticate to their identity provider, and a security token suitable for the relying party is generated and returned to the identity selector. The browser then sends the security token to the relying party where it is validated to complete the authentication flow.

Higgins identity selector

Higgins is an open-source implementation of a user-centric identity metasystem similar to the one offered by Microsoft Windows CardSpace. Higgins provides three types of identity functionality: an identity selector (either in a browser or standalone), identity provider/consumer Web-based services, and an identity attribute service.

The identity selectors are compatible with the emerging *Information Card* (or *i-card*) based approach to authentication. The identity provider/consumer Web-based technologies allow Web sites and servers to be i-card and OpenID-compatible. The identity attribute service provides a framework for an interoperability and portability layer over existing identity data, a mash-up to allow contributions from current systems and directories.

[2] For more information refer to http://openid.net/
[3] For more information refer to http://msdn2.microsoft.com/en-us/library/aa480189.aspx
[4] For more information refer to http://www.eclipse.org/higgins/

Higgins itself is composed of a number of Eclipse plug-ins (or OSGI bundles), which allow composition of many of these services with minimal footprint and fewer dependencies. The following link may be helpful because it provides a list of the solutions that can be created using Higgins components (http://wiki.eclipse.org/index.php/Solutions).

OpenID

OpenID is a lightweight framework for a user-centric identity system that came from the open source community. With the OpenID protocols, a user's identity is represented by a familiar-looking URL, such as:

http://myid.myopenidprovider.com/

Like the other federation standards described here, it has the concept of identity providers and relying parties. OpenID is expected to grow in the future to include identity services beyond authentication.

OpenID is particularly suited to situations where a service provider does not require a tightly coupled relationship with a user or identity providers. This fact has fueled OpenID's adoption across a large number of non-commercial sites, and gained interest and attention from Internet service providers such as America Online (AOL).

Web services identity

The other common type of federation is implemented using Web services. Whereas the federated single sign-on scenario that we have described deals with a browser-based user interaction to access Web based applications, Web services federations are based on application-to-application communication. Web services have identities as well. The same identity federation and trust issues apply as in the federation single sign-on case. However, the protocols and security standards that are applicable are different, and in many cases direct user interaction might not be possible to provide authentication information.

Web services

Web services are self-contained, modular applications that can be described, published, located, and invoked over a network. Web services perform encapsulated business functions, ranging from a simple request-reply to full business process interactions. A typical Web services application consists of a service consumer, a service provider, and optionally a registry for storing the Web services definitions. Web services are accessible over standard Internet protocols that are independent from platforms and programming languages.

WSDL

Loose coupling of Web services is achieved through use of the Web services description language (WSDL). WSDL defines the data format and transports for Web services messages in a platform neutral fashion. WSDL provides a grammar for describing services as a set of endpoints that exchange messages. A WSDL document is an XML document that describes services, how to access those services and what type of response (if any) is expected.

SOAP

SOAP is an industry-accepted specification used to describe messages for transport across a network. SOAP is a mechanism that allows sending and receiving of XML documents regardless of the transmission protocol or the structure of the XML document. The SOAP

XML protocol sits above the transport used to send the message protocol and beneath the domain-specific XML documents in the protocol stack.

The SOAP specification itself does not address security, although the specification does indicate that security could be added through the SOAP extension mechanisms.

WS-Policy

WS-Policy provides a flexible and extensible grammar for expressing the capabilities, requirements, and general characteristics of entities in an XML Web services-based system. WS-Policy defines a framework and a model for the expression of these properties as policies. Policy expressions allow for both simple declarative assertions as well as more sophisticated conditional assertions.

WS-Policy defines a policy to be a collection of one or more policy assertions. Some assertions specify traditional requirements and capabilities that ultimately manifest on the wire (for example, authentication scheme and transport protocol selection). Some assertions specify requirements and capabilities that have no wire manifestation yet are critical to proper service selection and usage (for example, privacy policy and QoS characteristics). WS-Policy provides a single-policy grammar to allow various kinds of assertions to be expressed in a consistent manner. Subordinate standards such as WS-Security Policy (see below) provide more concrete profiles for interoperability in a particular class of policy.

The specification that makes up WS-Policy is available for download at:

`http://www.w3.org/Submission/WS-Policy/`

WS-Security specifications

The WS-Security family specifications shown in Figure 4 introduce a set of individual, interrelated standards to form a layered approach to identifying and securing Web services.

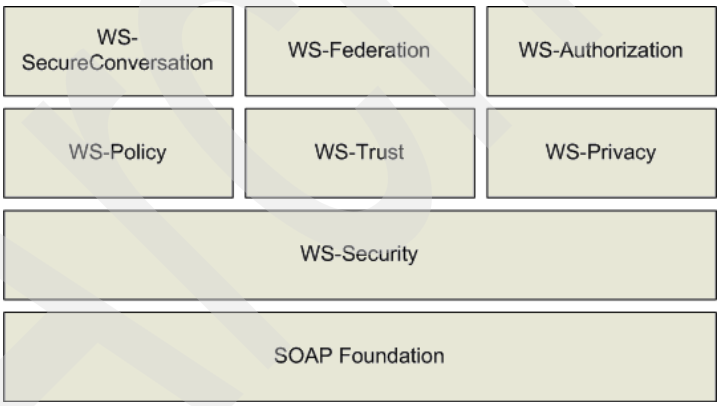

Figure 4 WS-Security family of specifications

A typical Web services identity and security implementation is shown in Figure 5 on page 12. In the figure the service consumer wants to send a Web service request to a service provider. The service consumer communicates with a *security token service* using a WS-Trust message to request a security token. This token is transmitted with the SOAP request. The security token carries identity information about the user, and a common example is a SAML assertion.

The request is then secured following the WS-Security specification so that it is encrypted and signed. The protected request and security token are sent to the service provider. At the service provider the signature on the message is checked and the message is decrypted.

Additionally the request is authorized and the token is validated by the receiver's own Security Token Service.

In Figure 5, the Security Token Service (STS) at both the Identity provider and service provider is implemented using IBM Tivoli Federated Identity Manager. However, because the design follows the Web services security standards, an alternative product or open source implementation can also be used.

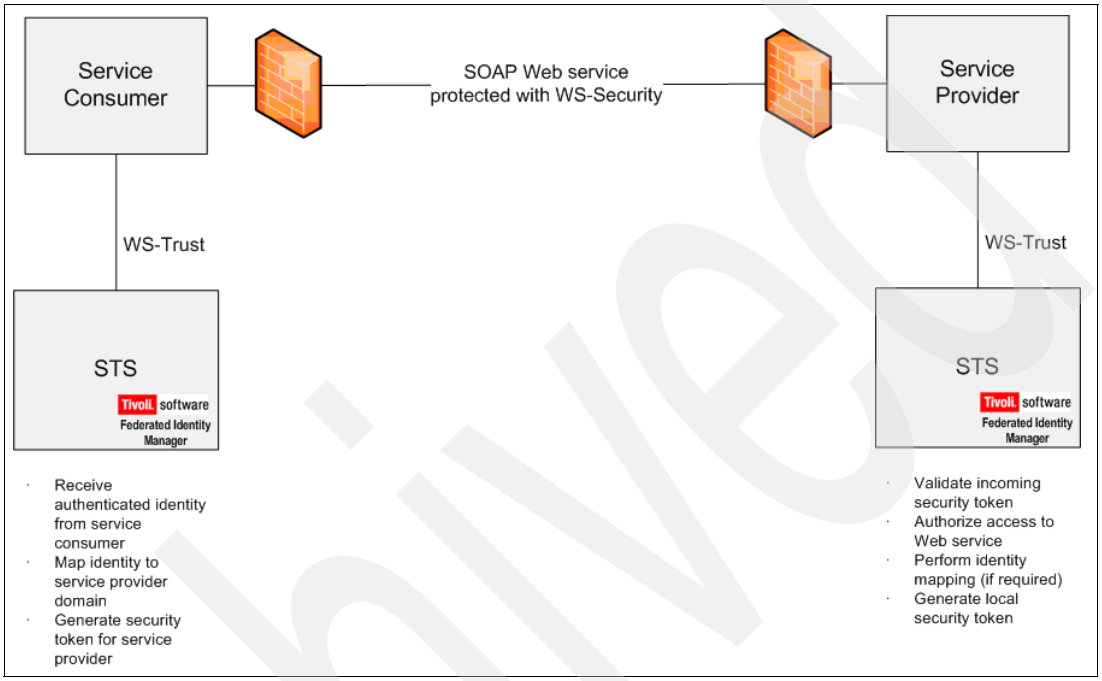

Figure 5 Securing Web services

WS-Security

The WS-Security specification provides standard method to capturing message-level security. The advantage of using WS-Security instead of SSL is that it can provide end-to-end, message-level security. This fact means that the messages are protected even if the message goes through multiple nodes, or intermediaries. Additionally, WS-Security is independent of the transport layer protocol. It can be used for any SOAP binding, not just for SOAP over HTTP.

As an overview, Figure 6 on page 13 shows the Web service message security elements that can be added to the SOAP header.

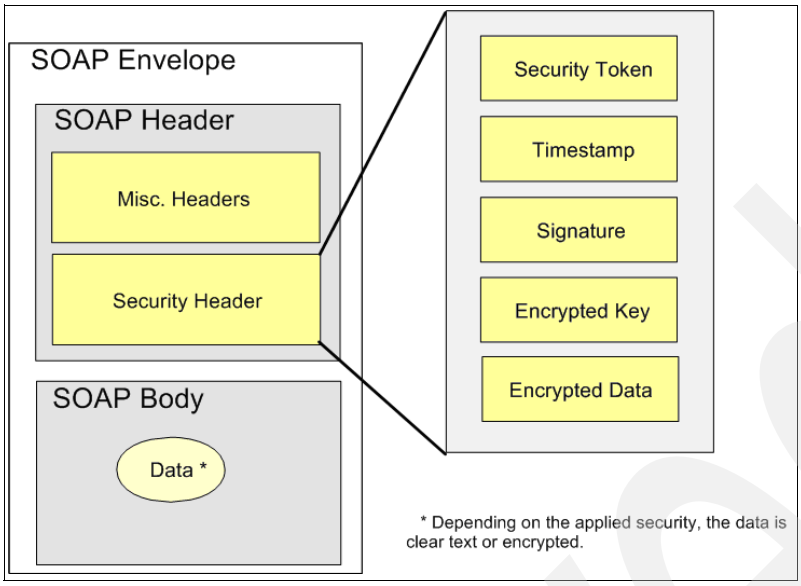

Figure 6 WS-Security

The WS-Security specification Version 1.1 was ratified by the OASIS WSS Technical Committee in February 2006. This specification proposes a standard set of SOAP extensions. This specification is flexible and is designed to be used as the basis for securing Web services within a wide variety of security models including PKI, Kerberos, and SSL. It provides support for multiple security token formats, multiple trust domains, multiple signature formats, and multiple encryption technologies to provide integrity or confidentiality.

The WS-Security specification defines the usage of *XML signature* and *XML encryption*. The specification includes security token propagation, message integrity, and message confidentiality. However, these mechanisms by themselves do not address all the aspects of a complete security solution. Therefore, WS-Security represents only one of the layers in a secure Web services solution design.

More information about the OASIS Web services security specification is available from:

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

WS-Trust

The Web services trust language (WS-Trust) uses the secure messaging mechanisms of WS-Security to define additional primitives and extensions for the issuance, exchange, and validation of security tokens. WS-Trust also enables the issuance and dissemination of credentials within various trust domains.

In order to secure a communication between two parties, the two parties must exchange security credentials (either directly or indirectly). However, each party needs to determine if they can trust the asserted credentials of the other party. This specification defines extensions to WS-Security for issuing and exchanging security tokens and ways to establish and access the presence of trust relationships.

IBM Tivoli Federated Identity Manager implements the *security token service* as defined in WS-Trust. This service can be used for creating, validating, and exchanging security tokens for WS-Security and WS-Federation. The STS also provides for authorization of Web service requests.

The specification that makes up WS-Trust is available from the OASIS Web site:

http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.pdf

WS-SecureConversation

The *Web services secure conversation language* (WS-SecureConversation) is built on top of WS-Security to provide secure communication between services. WS-Security focuses on individual message protection, but not establishing a security context. This specification defines mechanisms for establishing and sharing security contexts, and deriving keys from security contexts, to enable a secure conversation. A security context is important because it allows the establishment of a symmetric session key between the parties, providing more efficient cryptographic processing for each message. This is the same approach as used in SSL.

The WS-SecureConversation specification is available for download at:

http://docs.oasis-open.org/ws-sx/ws-secureconversation/v1.3/ws-secureconversation.pdf

WS-SecurityPolicy

The *Web services policy language* (WS-Policy) specification defines a set of policy assertions that apply to Web services. WS-SecurityPolicy defines ones profile of WS-Policy relating to security and provides policy assertions for WS-Security, WS-Trust, and WS-SecureConversation. WS-SecurityPolicy takes the approach of defining a base set of assertions that describe how messages are to be secured. Flexibility with respect to token types, cryptographic algorithms, and mechanisms used, including using transport-level security, is part of the design and allows for evolution over time. The intent is to provide enough information for compatibility and interoperability to be determined by Web services participants, along with all information necessary to actually enable a participant to engage in a secure exchange of messages.

The WS-SecurityPolicy specification is available for download at:

http://docs.oasis-open.org/ws-sx/ws-securitypolicy/v1.2/ws-securitypolicy.pdf

Identity service for service-oriented architectures

SOA connect loosely coupled services to construct new applications. The service implementations often have their own user registries that are administrated in isolation from those of other service implementations. Users and service entities in this environment are likely to differ in the various services that make up a composite application. Establishing the identity of the service requestor in each service request is a fundamental step in ensuring that business requirements such as authorization, audit and compliance can be implemented.

Identity services are, therefore, required in the SOA infrastructure so that services can be easily interconnected while the correct identities are propagated. IBM Tivoli Federated Identity Manager's Security Token Service (STS) provides a pluggable identity service solution for the challenge of SOA identity mediation that is:

- Capable of understanding and operating with a variety of formats for representing identity
- Capable of translating (mapping) between various identities
- Based on SOA principles that deliver a flexible, infrastructure-based solution de-coupled from application business logic

- Constructed using open standards to provide maximum interoperability with the platforms and systems on which SOA solutions are constructed

Figure 7 shows the interaction between an SOA component, such as an XML firewall, enterprise service bus, or even a back-end mainframe resource-based service, and the IBM Tivoli Federated Identity Manager STS using the WS-Trust standard. The SOA component calls on the security token service for generation, validation, or exchange of security tokens that helps identify the Web service with the environment.

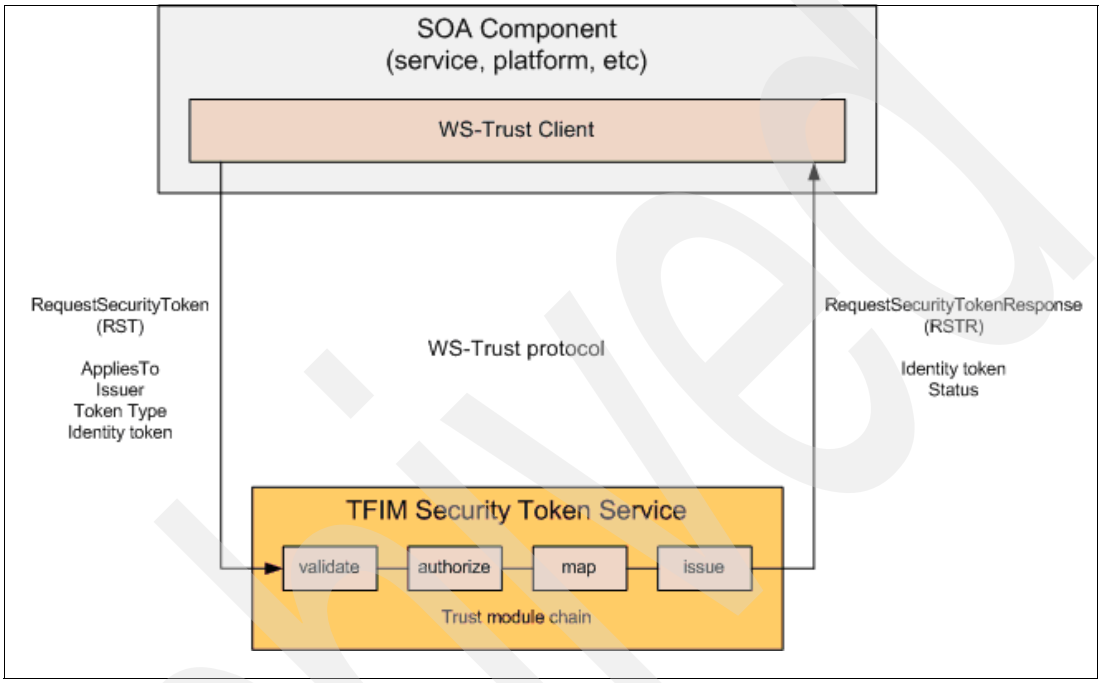

Figure 7 Tivoli Federated Identity Manager's implementation of WS-Trust

The benefits of using IBM Tivoli Federated Identity Manager's identity service in SOA identity mediation scenarios come from the number of integrated solutions which use the Tivoli Federated Identity Manager STS that are already available. The solutions that we describe here are just some examples of software components that include a WS-Trust client and use the IBM Tivoli Federated Identity Manager STS to enable identity mediation. The examples are described in the context of IBM SOA Foundation scenarios. You can find more information about securing SOA environments in the IBM Redbooks® publication *Understanding SOA Security Design and Implementation*, SG24-7310.

Scenario 1: Service creation

The service creation scenario (service provider and service consumer) is used to demonstrate exposing application functionality of an existing application or new business logic as a service. The services can then be consumed by other services or client applications within and between enterprises. The key driver for this scenario is the reuse of existing or new application functions as services.

The Web Services Security Management (WSSM) component of Tivoli Federated Identity Manager integrates with the Web services security capabilities of WebSphere Application Server. It provides:

- WSSM token consumer

  The token consumer authenticates the service requester to WebSphere Application Server after using the Tivoli Federated Identity Manager STS to validate the identity token received in the incoming Web service request. It allows applications running in WebSphere Application Server to support a range of identity token types that are not natively supported by the application server.

- WSSM token generator

  The token generator uses Tivoli Federated Identity Manager STS to convert the current identity in WebSphere Application Server to a different identity token type for transmission with outgoing Web services requests. It can be used to ensure that the identity included in a service request can be readily consumed by a service without the need for customization within the service.

The CICS integration pack for IBM Tivoli Federated Identity Manager offers a JAAS login module that sends the identity from the WebSphere Application Server subject to the Tivoli Federated Identity Manager STS in a UsernameToken and receives another UsernameToken representing the identity to be provided to the invoker of the module. Uses of this module include:

- Invocation from a CICS JCA module in WebSphere Application Server to obtain user name and password or RACF® Passticket to send to the CICS Transaction Gateway
- Invocation from a JDBC™ driver in WebSphere Application Server to obtain credentials for connecting to a relational database, such as IBM DB2®
- Programmatic invocation from a Java application

Scenario 2: Service connectivity

The service connectivity scenario is used to demonstrate the integration of service providers and consumers, allowing for the reuse of existing and new services across multiple channels. This scenario is appropriate for an enterprise that has a set of core services or systems that are to be made available as services to internal and external clients. Flexibility to make changes to service providers and service clients independent from each other is a requirement.

The focus of this scenario is on the underlying connectivity used to support business-centric SOA. An enterprise service bus (ESB) provides decoupling between clients and providers, providing the flexibility to implement applications more quickly. In circumstances where services are provided to or consumed from a third party, an ESB gateway can be used in conjunction with the ESB to add security measures. An ESB gateway alone can be sufficient if all of your service interactions are with third parties and if you have the basic requirements to mediate between service consumers and providers.

IBM Tivoli Federated Identity Manager's identity service offers standards-based identity-awareness for the three IBM ESB solutions.

- WebSphere ESB

  This solution provides integration with WebSphere Integration Developer from a tooling perspective to graphically add an identity mediation primitive to a mediation module. At runtime in WebSphere ESB and WebSphere Process Server, the identity mediation primitive extracts the identity token from an incoming request and uses the Tivoli

Federated Identity Manager STS to validate it and map to a different identity token, which is then included in service requests from the ESB.

- WebSphere Message Broker

  This integration provides an Identity Service user-defined node, which extracts the identity token from incoming SOAP messages and sends them to the Tivoli Federated Identity Manager STS for validation and mapping to an alternative identity. The identity received from the STS is included in outgoing requests from WebSphere Message Broker.

- WebSphere DataPower® SOA Appliances

  DataPower SOA appliances are often used at the edge of a network, playing the role of an XML firewall. DataPower SOA applications receive the requests from an un-trusted network and perform XML validation and WS-Security processing, such as checking signatures and decrypting messages.

  IBM Tivoli Federated Identity Manager integrates with DataPower by providing an exit in an authentication, authorization, and audit (AAA) policy definition to provide token validation and exchange, and authentication and authorization of the request. The communication between DataPower and IBM Tivoli Federated Identity Manager is by means of WS-Trust, with IBM Tivoli Federated Identity Manager providing the STS functionality for token validation and exchange.

Scenario 3: Interaction and collaboration services

The interaction and collaboration services scenario features single sign-on and a role-based portal used to consolidate access to information and application within the enterprise and between enterprises. The key drivers for this scenario are to improve people productivity and consumption of applications and content. The content can be personalized in the aggregated portal page based on the user role.

The single sign-on capabilities of IBM Tivoli Federated Identity Manager that we have described can be used to provide single sign-on between a portal and other Web applications. This functionality includes federated single sign-on to Web applications in different administrative domains.

The integrated capabilities described in Scenarios 1 and 2 also apply in this scenario, for Web services based integration from the portal to content provided by other portals using standards such as Web Services for Remote Portlets (WSRP).

Customer scenarios solutions

At the beginning of this paper we introduced several customer scenario challenges. Let us now take a look at possible solutions.

Public sector: Single login for government agency services

To solve the single login to government agency services, the agency chose to implement federated single sign-on using SAML 2.0 with IBM Tivoli Federated Identity Manager. As shown in Figure 8 on page 18, the government agency set up an identity provider, where the login credentials of users were managed. This centralized approach reduced the need to manage different credentials at each department. The identity provider authenticates users and produces SAML 2.0 security assertions for them. The users can then access the participating departments.

The government departments are SAML 2.0-compliant service providers that can validate the identity of the user based on the received assertions. In the initial implementations, the participating departments also used Tivoli Federated Identity Manager. However, it should be pointed out that because the SAML 2.0 standard is being implemented, the departments can choose other products or open source implementations.

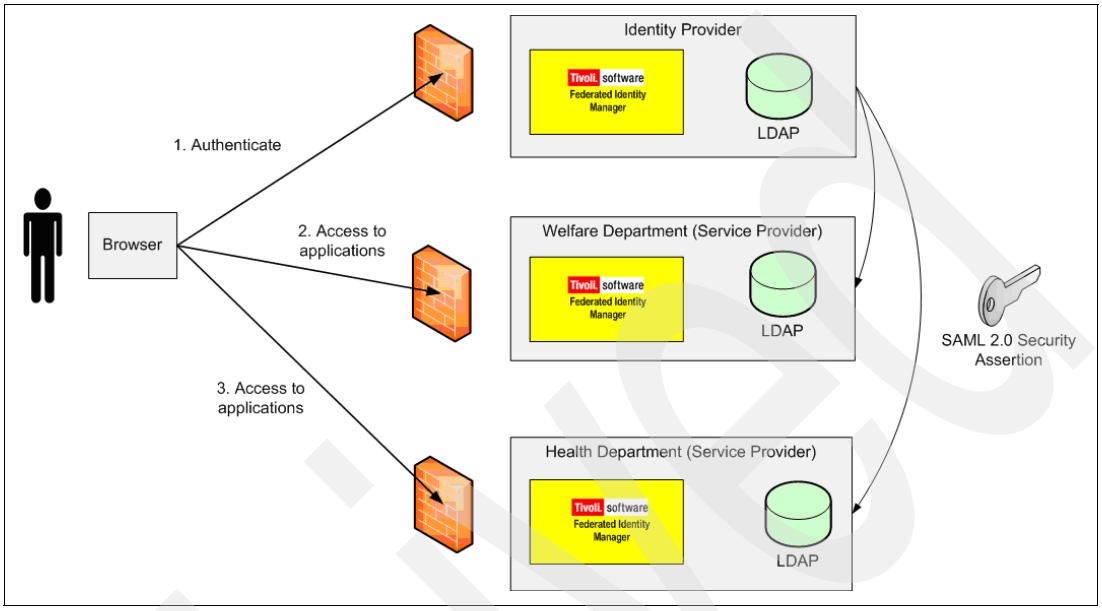

Figure 8 Single login for government agency services using SAML 2.0

In some cases a just-in-time provisioning model can be used to create the accounts at the service providers. The provisioning of the user accounts at the service provider can occur as part of the federated single sign-on processing on the Service Provider side rather than requiring that accounts be created at the Service Provider in advance. This functionality is enabled by the extensible nature of a SAML 2.0 assertion where it might contain arbitrary extended attributes. IBM Tivoli Federated Identity Manager provides flexible mechanisms for retrieving those attributes on the identity provider side before constructing the SAML 2.0 assertion, as well as ways to extract them from the assertion and integrate with other systems at the service provider.

Telecommunications industry: Making services available to external partners

To allow the telecommunications company to expose services to external partners, an edge gateway was set up in the company's DMZ to process incoming service requests. As shown in Figure 9 on page 19, the WebSphere DataPower XS40 provides the Web services proxy for incoming requests. The incoming messages are Web services requests, secured using WS-Security and carrying a SAML security assertion (or other tokens as are required by the business partners).

The XS40 communicates with IBM Tivoli Federated Identity Manager to validate the incoming security token, authorize the request, map the identity (with additional attributes added), and generate a new token that is suitable for inside the organization. In the figure, a new SAML 1.0 unsigned token is produced.

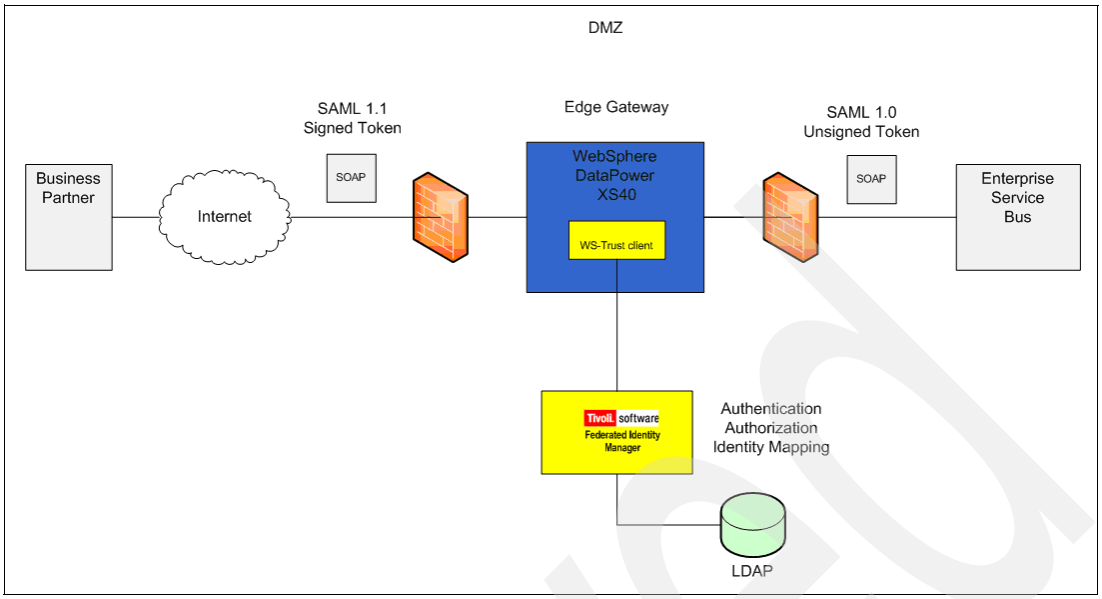

Figure 9 Telecommunications processing service requests from external partners

Financial industry: Re-purposing of existing IT systems

An organization in the financial services industry wanted to reuse business logic in the existing mainframe CICS applications from a portal. WebSphere Message Broker was used as the Enterprise Service Bus. The portal was secured using IBM Tivoli Access Manager and IBM Tivoli Directory Server. The CICS environment was running in an environment protected by RACF security server.

The identity propagation solution involved using the custom node in WebSphere Message Broker which integrates with IBM Tivoli Federated Identity Manager to validate and transform identities as shown in Figure 10.

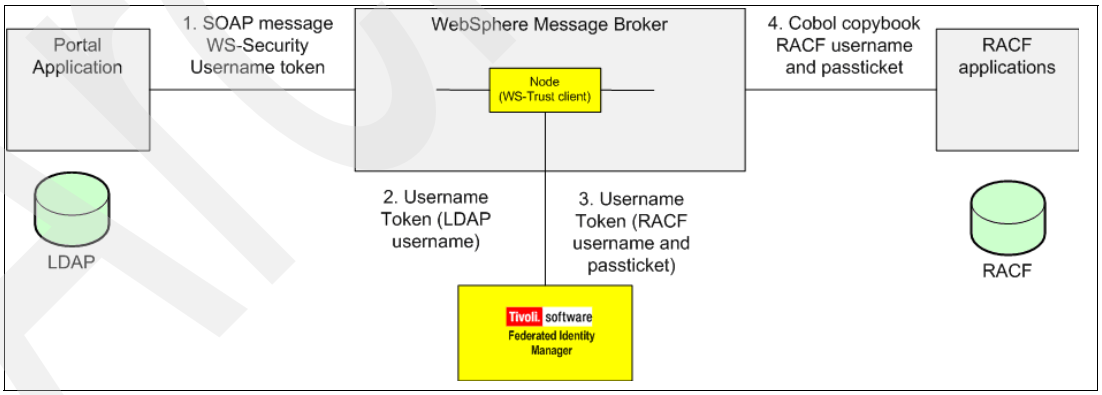

Figure 10 Using IBM Tivoli Federated Identity Manager for back end CICS integration

After authenticating the user, IBM Tivoli Access Manager for e-business would sign the user automatically onto the portal. When the portlet to access CICS content is invoked, the authenticated user identity is sent in a WS-Security UsernameToken to WebSphere Message Broker. The node in WebSphere Message Broker constructs a WS-Trust request and accesses IBM Tivoli Federated Identity Manager to locate the user in IBM Tivoli Directory Server and look up their RACF user name, which was stored in an attribute in the user's LDAP entry.

Next, a RACF passticket for the CICS application was generated in IBM Tivoli Federated Identity Manager, based on:

- The RACF user name obtained from the processing context in IBM Tivoli Federated Identity Manager
- A shared secret key configured in IBM Tivoli Federated Identity Manager to match the value in RACF
- An application identifier configured in IBM Tivoli Federated Identity Manager to match the value in RACF
- The current time

The RACF user name and passticket are returned to the node in WebSphere Message Broker, again in a WS-Security user name token. The Web service request from the WebSphere Message Broker to CICS or the CICS transaction gateway now carries the RACF user name and passticket, which are used to authenticate the service request.

Tivoli solution deployment and runtime

Let us now take a look at the solution deployment and runtime model.

Federated single sign-on

In Figure 11 we show the IBM Tivoli Federated Identity Manager components used when implementing federated single sign-on (SAML, Liberty or WS-Federation) or OpenID or Identity Selectors such as Microsoft Windows Cardspace™ or the Eclipse Higgins Project. This configuration is typical for a Web site acting either as an identity provider or service provider in a federation.

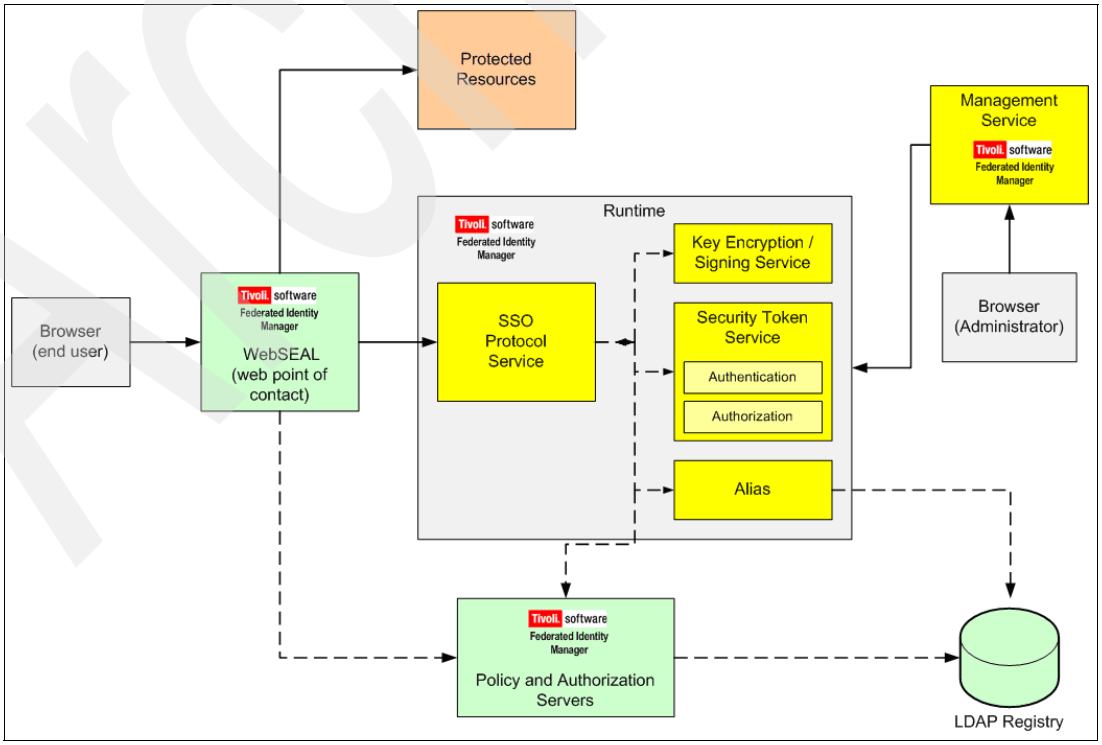

Figure 11 IBM Tivoli Federated Identity Manager deployment for federated single sign-on

In Figure 11, the important components are:

- IBM Tivoli Access Manager for e-business WebSEAL: Used for the Web point-of-contact server. All browser requests from the user pass through the point of contact.
- SSO Protocol Service: Implements the protocols for SAML, Liberty and WS-Federation.
- Security Token Service: An implementation of WS-Trust STS. Manages the tokens for the various federations, that is creation, validation, and exchange.
- IBM Tivoli Federated Identity Manager Management Service: A Web-based application for configuring the federation characteristics.
- IBM Tivoli Access Manager for e-business Policy Server: Used for management and distribution of the authentication/authorization policy.
- IBM Tivoli Access Manager for e-business Authorization Server: Provides the interaction with the LDAP directory from the STS authentication/authorization components.
- LDAP Registry: For storing the user identities and alias.

Flexible federated SSO deployment models

IBM Tivoli Federated Identity Manager also supports a deployment model where its SSO services can be integrated directly with Microsoft .NET and WebSphere applications without the need for IBM Tivoli Access Manager for e-business. This approach is particularly suitable in environments where an enterprise has an existing third-party access management solution or is a business partner that simply requires a deployment model in order to participate in a federation using the Web application server as the point of contact.

Conclusion

Trust management and federated identity management provide a standardized mechanism for simplifying identity management across company boundaries. This mechanism enables businesses to offload identity and access management costs to trusted business partners within the federation. Federated identities allow businesses to share identity information and entitlements in a trusted fashion between companies.

The flexibility and appeal of the Web services model is its building block foundation. Companies can build new services easily to deliver innovative business models or to link their value-chain network more efficiently with tight relationships with partners, suppliers, customers, and employees. Such a model can be successful only if it allows customers, partners, and users to navigate easily between Web sites that support these services.

This ease of use requires that users not be constantly authenticating or identifying themselves to the various sites within their business federation. The ability of Web services implementations to take advantage of Federated Identity Management facilitates real ROI for companies. This approach can result in improved integration, communication, and information exchange between suppliers, business partners, and customers. It uses IT productively to lower enterprise costs, improve productivity, and maximize efficiency in business operations.

This paper discusses specific federation scenarios that companies can use productively today with IBM Tivoli Federated Identity Manager. These scenarios represent commonly seen customer-use cases that securely link the value net of the company to achieve ROI through improved usability, decreased management costs (of users and infrastructure), and decreased development costs.

The team that wrote this paper

This paper was produced by a team of specialists from around the world working at the International Technical Support Organization, Austin Center.

Axel Buecker is a Certified Consulting Software IT Specialist at the International Technical Support Organization, Austin Center. He writes extensively and teaches IBM classes worldwide on areas of Software Security Architecture and Network Computing Technologies. He holds a degree in computer science from the University of Bremen, Germany. He has 21 years of experience in a variety of areas related to Workstation and Systems Management, Network Computing, and e-business Solutions. Before joining the ITSO in March 2000, Axel worked for IBM in Germany as a Senior IT Specialist in Software Security Architecture.

Paul Ashley is a Senior Certified IT Specialist and Lead Architect for the SOA Advanced Technologies Asia Pacific team, part of the IBM Software Group. The team specializes in new SOA engagements and technology. Paul has worked in the IT industry for 18 years and holds degrees in Electronics Engineering, Computer Science, and a Ph.D. in Information Security. Before joining the SOA Advanced Technologies team, Paul worked as a Security Specialist for Tivoli Security in both the U.S. and Australia. He is based at the Australian Development Labs on the Gold Coast.

Neil Readshaw is a Senior Certified IT Specialist in Tivoli's Worldwide Customer Solutions (SWAT) team. He is based in the Gold Coast, Australia. He has 15 years of experience in software development, network management, information security, and systems integration. He holds degrees in Computer Systems Engineering and Computer Science from the University of Queensland, as well as the Certified Information Systems Security Professional (CISSP) and IT Infrastructure Library® (ITIL®) certifications. He has written extensively for the Tivoli Developer Domain on the IBM developerWorks® site.

Thanks to the following people for their contributions to this project:

Anne Johnson, Editor
International Technical Support Organization, Research Triangle Park, North Carolina

Bruce Rich, Shane Weeden, Davin Holmes, Patrick Wardrop, Ravi Srinivasan, and Sridhar Muppidi
IBM

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

This document REDP-3678-01 was created or updated on May 15, 2008.

Send us your comments in one of the following ways:
- Use the online **Contact us** review Redbooks form found at:
  ibm.com/redbooks
- Send your comments in an email to:
  redbooks@us.ibm.com
- Mail your comments to:
  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400 U.S.A.

Trademarks