Design and implement just-in-time provisioning with SAML 2.0

Guruprasad Saralaya
Rajesh Baronia

August 14, 2012

Understand and get a demonstration of how to design and implement just-in-time (JIT) provisioning at a service provider with SAML 2.0 Federated Single Sign-On using Tivoli® Federated Identity Manager, Tivoli Directory Integrator, and Tivoli Access Manager.

Background

When businesses adopt cloud-based services, they provide services to end users through a federation of multiple independent business partners. Each business partner must be able to provision end users for their service. However, many times it is not practical to provision all end users to all services before the service is needed. Instead, businesses adopt a just-in-time approach by provisioning end users to the business partners' services the first time they sign on to the service.

Federated identity management concepts

In identity management, a federation is a group of two or more business partners who work together. Figure 1 illustrates the basic concepts in federated identity management. The identity provider (IDP) is the authoritative site responsible for authenticating an end user and asserting a trusted identity for that user to trusted business partners. Business partners who offer services but do not act as identity providers are known as service providers (SP). This division of responsibilities allows service providers to off-load authentication and access management costs to the identity providers and to receive trusted information about a customer without requiring customers to register separately at the service provider.
IBM Tivoli Federated Identity Manager (TFIM) is an IBM solution that provides web and federated single sign-on (SSO) to end users across multiple identity and service providers. TFIM supports end-user SSO using a variety of industry standard security tokens and protocols such as WS-Provisioning, and provides web service security for web service calls.

Federated identity provisioning

Provisioning is a key element in identity management for enterprises. Account provisioning is triggered within a company's internal trust domain when a change occurs in a user's account status. For example, a new user account must be created when a new employee is hired, or user account permissions might need to be modified when an employee changes jobs. These types of activities are known as identity provisioning.

Federated identity provisioning extends these provisioning management activities beyond the boundary of a single company and extends identity provisioning activities to the service providers in the federation. A service provider, when notified of the federated provisioning request, can perform the local provisioning necessary to supply its service to the specified end user.

WS-Provisioning is a specification authored by IBM to provide a web service interface to communicate provisioning requests and responses. However, using this mechanism requires that user identities be created at the service provider before a user can actually perform federation to access the services. This provisioning process can be made just-in-time using the approach described in this article.

Just-in-time provisioning

With just-in-time provisioning, the end user identity is provisioned (created or updated) at the service provider the first time the end user tries to access the service provider’s service—without the need for prior identity provisioning activity between the identity provider and the service provider.

Also with just-in-time provisioning, the identity provider can include attributes needed by the service provider for provisioning the end user. These attributes can include information such as the user name, first name, last name, and email address, and are packaged in a security token such as a Security Assertion Markup Language (SAML) assertion. The identity provider sends this SAML assertion to the service provider when the end user accesses the service provider as part
of a single sign-on. If no match exists for the presented user name, the service provider creates a new account with the end user attributes contained in the SAML assertion. The service provider also immediately grants access to the requested services to the end user.

If a match is found and an account does exist for the end user, the service provider updates the account according to the attribute information in the SAML assertion. A wide variety of attributes are supported (for example, a profile ID can be specified for new or existing accounts); therefore, the federated business partners have fine-grained control over the account creation and update process.

The following section illustrates a just-in-time provisioning scenario from the service provider's point of view.

**A just-in-time provisioning use case**

Company A is a large company that provides various employment benefits such as subsidized health care insurance, retirement savings plans, and other employment-related services. As part of reducing its employee costs, Company A has outsourced these employee benefits to a third-party benefit provider, Company B. Company B provides a hosted service that employees of company A can access for employment benefit services.

Let's say a new employee, John Doe, joins Company A and has been issued a set of credentials such as an ID and password to use when accessing Company A's internal applications. John needs to be able to access his benefits information hosted at Company B. This usually requires that company B create an account for John using a manual process or a back-end data exchange process between Company A and Company B. This step is typically outside the context of the federated single sign-on and needs to be completed before the user tries to access his benefits at Company B.

However, with a just-in-time provisioning solution in place at company B, when John Doe accesses Company B's website for the first time, the SAML-based federated single sign-on process enables Company B to automatically create John Doe's account and grant access to his employee benefits.

**Components of a just-in-time provisioning system**

Figure 2 shows the components of the solution and a high-level logical description of the interactions between them.
The major interactions between the components shown in Figure 2 are:

- The Identity Provider authenticates and authorizes the end user, then provides a SAML 2.0 assertion to the service provider using an IBM or third-party federation tool.
- The SAML 2.0 assertion token includes the attributes required to provision the user at the service provider. The attributes may include information such as the user ID, first name, last name, job title, department, role, and email address.
- The WebSEAL component acts as a point-of-contact server at the service provider and implements single sign-on to the requested services at the service provider.
- Tivoli Federated Identity Manager manages the federation configuration, validates digital signatures, processes security tokens, and triggers identity provisioning activities.
- A Tivoli Directory Integrator-based custom mapping rule is used to parse the SAML assertion token, extract attributes from the token and invoke the Tivoli Access Manager Connector to create or update the user's account, and map the account back to Tivoli Federated Identity Manager.
- Tivoli Access Manager grants access to resources at the service provider based on the user's account information and its authorization policy.

**How TFIM processes the SAML assertion**

Figure 3 illustrates the steps that Tivoli Federated Identity Manager follows to process a SAML assertion as part of the FSSO protocol.
Figure 3. Processing A SAML Assertion

Tivoli Access Manager, acting as the service provider point-of-contact server, receives the authenticated user identity in the form of a SAML assertion.

The point-of-contact server sends the SAML assertion token to Tivoli Federated Identity Manager's single sign-on protocol service, which is processed by the trust module chain as follows:

- The SAML assertion token is parsed, validated, and converted into an internal XML representation called STSUniversalUser token.
- A custom Tivoli Directory Integrator mapping module (see tdi_saml20_JITP_mappings.xml and JITP.Properties in the downloads.zip file in Downloadable resources) maps the attributes from the STSUniversalUser token to the token format used by Tivoli Access Manager. The mapping module calls Tivoli Access Manager to create or update the end user's account, as needed, using the attributes in the STSUniversalUser token.
- The STSUniversalUser token is converted to the token type required by Tivoli Access Manager.

Tivoli Access Manager receives the token from Tivoli Federated Identity Manager and grants access to the site resources requested by the user based on its authorization policy.

**Tivoli Directory Integrator mapping module**

Figure 4 shows the functions performed by the Tivoli Directory Integrator mapping module used in this project.
The assembly line retrieves the required user attributes from the STSUniversalUser token and transforms them into Tivoli Access Manager schema-specific attributes.

The assembly line connects to Tivoli Access Manager and triggers provisioning create or update actions using the attributes retrieved from the STSUniversalUser token.

The assembly line generates a Tivoli Access Manager token by setting the principal user type for Tivoli Access Manager.
Implementing a just-in-time provisioning system

The solution described in this article implements the service provider’s role for just-in-time provisioning using Tivoli Federated Identity Manager 6.2.2, Tivoli Directory Integrator 6.1.1 or 7.0, and Tivoli Access Manager 6.1.

The SAML 2.0 assertion token

The design for the system is independent of components or product deployment at the identity provider. The just-in-time provisioning system only relies on the identity provider sending a valid SAML assertion token that includes the defined mandatory attributes. Listing 1 shows an example of a common attribute definition.

Listing 1. Example attribute defined in SAML 2.0

```xml
<saml:Attribute Name="cn"
    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:unspecified">
    <saml:AttributeValue xsi:type="xs:anyType">testuser</saml:AttributeValue>
</saml:Attribute>
```

Table 1 shows a typical list of the minimum attributes needed to provision a Tivoli Access Manager account at the service provider.

Table 1. Typical list of minimum attribute required to create an account on Tivoli Access Manager

<table>
<thead>
<tr>
<th>SAML attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cn</td>
<td>Common name or full name</td>
</tr>
<tr>
<td>Sn</td>
<td>Last name</td>
</tr>
<tr>
<td>Givenname</td>
<td>First name</td>
</tr>
<tr>
<td>Mail</td>
<td>Email address of the user</td>
</tr>
<tr>
<td>Title</td>
<td>Title of the user in the organization. This can be used to determine the access.</td>
</tr>
<tr>
<td>Role</td>
<td>Role or position of the user. This can be used to determine the access.</td>
</tr>
<tr>
<td>UID / Principle Name</td>
<td>Unique identity that will be used to create the user. Please note that any attribute that uniquely identifies the user or the attribute identified according to the business requirement can be used.</td>
</tr>
<tr>
<td>Organization</td>
<td>Name of the company to which the user belongs</td>
</tr>
<tr>
<td>EmployeeNumber</td>
<td>Can be used for tracking purposes. This is the unique identifier representing the user within an organization.</td>
</tr>
</tbody>
</table>

ip_saml_20_JITP.xsl (see Downloadable resources) provides the sample identity mapping stylesheet that can be used to define the identity mapping rule at the IDP using Tivoli Federated Identity Manager for its configuration.

Setting up Tivoli Federated Identity Manager for single sign-on

At the service provider, use Tivoli Federated Identity Manager’s default configuration steps to configure the SAML 2.0-based federation to be used with the business partner’s identity provider.
Setting up the Tivoli Directory Integrator assembly line

Figure 4 shows the Tivoli Directory Integrator assembly line configuration screen and illustrates the three components of the assembly line needed for just-in-time provisioning.

Figure 5. Tivoli Directory Integrator assembly line

The first component of the assembly line is the Get Attribute (GetAttr) component, which retrieves and maps all the attributes from the incoming SAML assertion token to the STSUniversalUser token. Table 2 shows the mapping of the attributes from the SAML 2.0 assertion token to the STSUniversalUser token.

Table 2. Mapping SAML 2.0 attributes to the STSUniversalUser token

<table>
<thead>
<tr>
<th>Number</th>
<th>Mapped SAML attribute</th>
<th>Attribute name in the STSUniversalUser token</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Cn</td>
<td>STSUU.AttributeList.Attribute.cn.values</td>
</tr>
<tr>
<td>2</td>
<td>Sn</td>
<td>STSUU.AttributeList.Attribute.sn.values</td>
</tr>
<tr>
<td>3</td>
<td>Givenname</td>
<td>STSUU.AttributeList.Attribute.givenname.values</td>
</tr>
<tr>
<td>4</td>
<td>Mail</td>
<td>STSUU.AttributeList.Attribute.email.values</td>
</tr>
<tr>
<td>5</td>
<td>Title</td>
<td>STSUU.AttributeList.Attribute.title.values</td>
</tr>
<tr>
<td>6</td>
<td>Role</td>
<td>STSUU.AttributeList.Attribute.role.values</td>
</tr>
<tr>
<td>7</td>
<td>Uid</td>
<td>STSUU.AttributeList.Attribute.name.values</td>
</tr>
<tr>
<td>8</td>
<td>Organization</td>
<td>STSUU.AttributeList.Attribute.organization.values</td>
</tr>
<tr>
<td>9</td>
<td>Employeenumber</td>
<td>STSUU.AttributeList.Attribute.employeenumber.values</td>
</tr>
</tbody>
</table>

The second assembly line component is the CreateTAM component. Tivoli Directory Integrator provides a default Tivoli Access Manager connector that is used in update mode to create or update the Tivoli Access Manager account as needed. Table 3 shows how the mapped STSUniversalUser token attributes correspond to the Tivoli Access Manager account attributes.
Table 3. Mapping STSUniversalUser token attributes to Tivoli Access Manager account attributes.

<table>
<thead>
<tr>
<th>Number</th>
<th>TAM attribute</th>
<th>Attribute name in the STSUniversalUser token</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Firstname</td>
<td>Cn (mandatory)</td>
</tr>
<tr>
<td>2</td>
<td>Lastname</td>
<td>Sn (mandatory)</td>
</tr>
<tr>
<td>3</td>
<td>Username</td>
<td>Uid (mandatory)</td>
</tr>
<tr>
<td>4</td>
<td>RegistryUID</td>
<td>Return the complete LDAP DN as required and supported at your environment – (e.g. cn=John,dc=ibm,dc=com) (mandatory)</td>
</tr>
<tr>
<td>5</td>
<td>IsAccountValid</td>
<td>True (mandatory)</td>
</tr>
<tr>
<td>6</td>
<td>IsPasswordValid</td>
<td>True (mandatory)</td>
</tr>
<tr>
<td>7</td>
<td>Password</td>
<td>Dummy password (mandatory)</td>
</tr>
<tr>
<td>8</td>
<td>Mail</td>
<td>Mail</td>
</tr>
<tr>
<td>9</td>
<td>Groups</td>
<td>Role (Can also be derived based on title, organization or any other required attributes)</td>
</tr>
<tr>
<td>10</td>
<td>Title</td>
<td>Title</td>
</tr>
<tr>
<td>11</td>
<td>O</td>
<td>Organization</td>
</tr>
<tr>
<td>12</td>
<td>EmployeeNumber</td>
<td>EmployeeNumber</td>
</tr>
</tbody>
</table>

The third assembly line component is the setSAML20Attrs component, which sets the STSUU.Principal.Attribute.name.type attribute in the STSUniversalUser token to the Tivoli Access Manager name returned in the Urn:ibm:names:ITFIM:5.1:accessmanager field.

The tdi_saml20_JITP_mappings.xml file (see Downloadable resources) provides the required assembly line implementation that can be extended for the service provider configuration.

Integrating Tivoli Federated Identity Manager and Tivoli Directory Integrator

The Tivoli Directory Integrator Security Token Service module of Tivoli Federated Identity Manager must be configured to make client calls to Tivoli Directory Integrator's assembly line. As shown in Figure 5, use the partner configuration steps of the Federation Partner Wizard in Tivoli Federated Identity Manager to select the second radio button on the Identity Mapping Options screen, "Use Tivoli Directory Integrator for identity mapping."
Next, define the Tivoli Directory Integrator connection information to Tivoli Federated Identity Manager, as shown in Figure 6.

**Figure 6. Tivoli Federated Identity Manager identity mapping options**

Table 4 shows the fields on the Tivoli Directory Integrator connection screen that need to be configured.

![Identity mapping options](image)

![Tivoli Directory Integrator connection information](image)

Table 4 shows the fields on the Tivoli Directory Integrator connection screen that need to be configured.
Table 4. Configuring the Tivoli Directory Integrator connection.

<table>
<thead>
<tr>
<th>Field</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server hostname</td>
<td>TDI Hostname or IP address of the Tivoli Directory Integrator server running the assembly line</td>
</tr>
<tr>
<td>Server Port</td>
<td>TDI port number, usually 1090</td>
</tr>
<tr>
<td>Assembly Line Handler Pool Size</td>
<td>10 is the default</td>
</tr>
<tr>
<td>Amount of time for threads to wait for an assembly line handler to become available</td>
<td>The default is to wait indefinitely</td>
</tr>
<tr>
<td>Method for selecting the assembly</td>
<td>Select Manual then enter the name of the Configuration file and the name of the assembly line defined in the configuration file. Using the example files from Downloadable resources, the file name would be tdi_saml20_JITP_mappings.xml and the assembly line name would be JITP-SP_Saml20.</td>
</tr>
<tr>
<td>Identification format for the Work Entry attributes</td>
<td>Select “Attribute Name”</td>
</tr>
</tbody>
</table>

Configuring Tivoli Directory Integrator

Tivoli Directory Integrator must be configured to have a secure channel to make calls to Tivoli Access Manager. The detailed steps for configuring the secure channel for TAM communication is beyond the scope of this article. The code in Listing 2 defines the secure channel.

Listing 2. Command to establish a secure channel to Tivoli Access Manager

```
```

Next, the Tivoli Directory Integrator Assembly line that will be used by TFIM must be configured as described in Table 5.

Table 5. Tivoli Directory Integrator assembly line properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>api.remote.on</td>
<td>Set this to true. This allows Tivoli Federated Identity Manager to invoke the remote service API.</td>
</tr>
<tr>
<td>api.remote.ssl.on</td>
<td>Set this to true. Signifies that SSL should be used when invoking the remote API. (Configuration of SSL is beyond the scope of this article.)</td>
</tr>
<tr>
<td>api.remote.nonsssl.hosts</td>
<td>This property is not used because SSL is configured.</td>
</tr>
<tr>
<td>api.config.folder</td>
<td>Location of TDI configuration files that can be edited through the server API. The default is /opt/IBM/TDI/V7.1/configs. The TDI assembly line tdi_saml20_JITP_mappings.xml must be placed in this directory.</td>
</tr>
</tbody>
</table>

Next, configure the Tivoli Directory Integrator daemon properties as shown in Table 6. You can modify the JITP.Properties file (from Downloadable resources) to match your environment.

Table 6. Tivoli Directory Integrator daemon properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
TAM_ID | <SEC_MASTER>
-------|------------------
TAM_Password | Set this to the password of the <SEC_MASTER> ID
TAM_Domain | The TAM Domain, default is “Default”
TAM_ProgramName | Application name as specified when running the svrsslcfg command
TAM_ConfigurationFile | File specified when running the svrsslcfg command
TAM_EntryType | Default User
TAM_ImportUsers | Defaults to False

After the configuration changes have been made, restart the Tivoli Directory Integration Server daemon with the `-d` flag. For example, `ibmdisrv -d`

**Testing the just-in-time provisioning system**

Verify the just-in-time provisioning system with the following steps:

1. Choose a user identity and verify that it does not exist at the service provider.
2. Log in to the identity provider's site with the chosen identity.
3. Click the inter-site transfer link to go to the service provider's site.
4. Verify that the chosen user identity was created on the service provider site.
5. Verify that the chosen identity has been granted appropriate authority to resources on the site.

**Troubleshooting the just-in-time provisioning system**

For errors related to the JITP solution, the following steps can be followed to collect debugging information.

- Enable detailed logging for the TAM connector in TDI, to debug provisioning action. The log output would be directed to the `ibmdisr.log`.
- Enable TFIM trace for the following to debug the token messages being processed at SP. The log output would be redirected to `trace.log`.
  - `com.tivoli.am.fim.saml20.*=all`
  - `com.tivoli.am.fim.trustserver.*=all`
  - `com.tivoli.am.fim.audit.*=all`
  - `com.tivoli.am.fim.fedmgr2.*=all`

**Extending the system for complete life cycle management**

Deprovisioning user accounts is not covered as part of the use case presented above and is beyond the scope of this article, but there are two basic approaches that can be followed for deprovisioning.

- Deprovisioning through a custom attribute:
  - Use an attribute `account-expiry-time`, which is passed as part of the user identity assertion to the service provider.
  - The just-in-time user provisioning system on the service provider adds this attribute to the user account when provisioning the user.
• The service provider establishes a separate process that periodically runs to suspend/remove the user account based on the `account-expiry-time` attribute in the user account.

• Deprovisioning the traditional way:
  • The service provider receives a periodic file update from the identity provider with a list of inactive users.
  • The service provider processes this list and suspends or removes the user identity from its repository as appropriate.

Conclusion

The just-in-time provisioning system described in this article is a service provider-side solution that uses the attributes passed in a SAML assertion to create and update the user identity information at the service provider in real-time, as needed. The just-in-time provisioning system is useful in cases where the user identity information is not required to be created or known by the service provider before the end user attempts to access the service provider.
### Downloadable resources

<table>
<thead>
<tr>
<th>Description</th>
<th>Name</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code sample</td>
<td>downloads.zip</td>
<td>5KB</td>
</tr>
</tbody>
</table>
Related topic

- For details on the format of SAML assertions, see the SAML 2.0 Specification.

© Copyright IBM Corporation 2012
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)