Architecture

WSO2 Identity Server is a product built on top of WSO2 Carbon. Based on the OSGi specification, it enables easy customization and extension through its componentized architecture. This topic describes the architecture of the Identity Server. The users are given the choice of deployment to on-premise servers, private cloud or public cloud without configuration changes.

Each server in the WSO2 platform is built using the Carbon platform. "Carbon server" is a term used to depict any product, such as WSO2 Enterprise Service Bus, WSO2 Application Server, and WSO2 Identity Server, that is built on top of the Carbon platform.

The WSO2 Identity Server is used directly by multiple users, through its user-friendly Management Console. Apart from the default admin user (with the user name ‘admin’), other users can be created later by the admin users that have the privileges to create a new user, or by signing up. Each user can have roles, where each role can have privileges assigned to them. A user’s roles can be changed at any time by the admin user. See here for more information on user management.

Apart from such registered users, Identity Server is also used as an identity provider for third party applications, which also have their own sets of users.

Architecture and process flow

The following diagram depicts the architecture of the Identity Server and the various processes that take place within it.

The following table lists out the components pertaining to the architecture of the WSO2 Identity Server, which are depicted in the above figure.

<table>
<thead>
<tr>
<th>Name</th>
<th>Component and description</th>
<th>Process flow</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
A Service Provider (SP) is an entity that provides Web services. A service provider relies on a trusted Identity Provider (IdP) for authentication and authorization. In this case, the Identity Server acts as the IdP and does the task of authenticating and authorizing the user of the service provider.

Salesforce and Google Apps are examples of service providers and are used as such in this case.

### Related links

- See **Adding a Service Provider** for information on how to add a service provider to the Identity Server and do the necessary configurations to integrate this SP with the Identity Server.
The Inbound Authentication component of the Identity Server can handle any of the following requests.

- **SAML SSO**: Security Assertion Markup Language (SAML) is an OASIS open standard for representing and exchanging user identity and authentication data between parties. SAML provides the web-based Single-Sign-On capability.

- **OAuth/OpenID Connect**: OAuth 2.0 has three main phases. They are; requesting an Authorization Grant, exchanging the Authorization Grant for an Access Token and accessing the resources using this Access Token. OpenID Connect is another identity layer on top of OAuth 2.0. OAuth applications can get authentication event information over the IDToken and can get the extra claims of the authenticated user from the OpenID Connect UserInfo endpoint.

- **Passive STS**: A Security Token Service (STS) is a software based identity provider responsible for issuing security tokens, especially software tokens, as part of a claims-based identity system.

- **OpenID**: OpenID is a URL or an XRI issued by an OpenID Provider. This OpenID Provider also maintains a user profile and this OpenID is used as a means of authentication.

### Related links

- See Configuring Inbound Authentication for a Service Provider for information on how to configure inbound authentication.
Claim management is a key aspect of the Identity Server, which helps to map local claims to service provider claims and vice versa. It also enables you to map local claims to identity provider claims and vice versa.

Just-in-Time provisioning allows you to create users on the fly without having to create user accounts in advance. For example, if you recently added a user to your application, you don't need to manually create the user in the Identity Server. When they log in with single sign-on, their account is automatically created for them, eliminating the time and effort related to creating the account. Just-in-Time provisioning works with your identity provider to pass the correct user information to the Identity Server.

Related links

- See Configuring Claims for a Service Provider for information on mapping local claims and service provider claims.
- See Configuring Claims for an Identity Provider for mapping local claims and identity provider claims.
- See Configuring Just-In-Time Provisioning for an Identity Provider for information on configuring Just-in-Time provisioning.
- See Architecture Authentication Framework for more information on the various authentication processes possible in this framework.
### Local authenticators

**Local Authenticators**

- Username/Password
- IWA

Local authenticators are authentication processes available within the Identity Server itself. The Username/Password authentication happens by authenticating the credentials entered against the values in the user store connected to the Identity Server.

**Related links**

- See [Configuring Local and Outbound Authentication for a Service Provider](#) for more information on how to configure local authenticators in the Identity Server.

The IN channel of the authentication framework sends the authentication request to the local authenticators component.

The local authenticator does the authentication for the username and password or by using Windows-based authentication (IWA). Once this is authentication is done, it provides an authentication response to the OUT channel of the authentication framework.

![Authentication Framework Diagram](image_url)
Federated authenticators are authentication processes that are not available within the Identity Server. These need to be configured to reach out to external applications to do the authentication process and send the response back to the Identity Server.

**Related links**

- See [Configuring Federated Authenticators for an Identity Provider](#) for more information on how to configure federated authenticators with the various identity providers.

The IN channel of the authentication framework sends the authentication request to the federated authenticators component. The federated authenticators do the authentication request in the specified authenticator. For example, if Facebook is configured, the authentication process will reflect that. Once this authentication is done, it provides the authentication response to the OUT channel of the authentication framework.
Currently, the Identity Server only supports the following external applications.

- OpenID
- SAML2 Web SSO
- OAuth2/OpenID Connect
- Passive STS
- Facebook
- Yahoo
- Google
- Microsoft (Hotmail, MSN, Live)

In the case of OpenID, SAML2 Web SSO, OAuth2, OpenID Connect and Passive STS; these are not standalone applications but rather applications that use these features for their authentication purposes. Hence configurations need to be done at the external application’s side as well to receive authentication requests from the Identity Server. These applications are known as identity providers who perform the authentication.

The authentication request comes in from the federated authenticators component and is distributed to the application. The user is authenticated and logged in to the question.

A single authentication request can require authentication from multiple external applications.
## Provisioning framework

**Provisioning Framework**

The provisioning framework is responsible for all provisioning work done by the Identity Server. This framework integrates with the User Store Manager component and also receives provisioning requests from the authentication framework.

### Related links

- See [Architecture Provisioning Framework](#) for more information on the various provisioning processes possible in this framework.

---

### Diagram 1

The JIT provisioning component of the OUT channel in the authentication framework sends a request to the provisioning framework. This occurs if the user is not added into the user store and needs to be added on-the-fly.

The provisioning framework sends the user details to the user store manager and the user is added. Once the user is added, the user store manager contacts the provisioning framework with the response.

The SCIM and SOAP requests that arrive from the service provider are added to the user store manager. This information is sent on the provisioning framework. The provisioning framework sends this along to the outbound provisioning component.
Authorization does not play a direct role in the process flow but as a component, it integrates with various other components in the Identity Server. This is primarily a functionality that can be managed using APIs that are written to perform authorization tasks.

Authorization Manager

WSO2 Identity Server contains an advanced entitlement auditing and management. It provides entitlement management for any REST or SOAP calls. WSO2 Identity Server provides attribute and claim-based access control via XACML, WS-Trust, OpenID, OpenID Connect and claim management. WSO2 Identity Server also provides role-based access control (RBAC) and fine-grained policy-based access control via XACML.

WSO2 Identity Server provides a friendly user interface for policy editing. It also supports multiple Policy Information Point (PIP) and policy distribution to various Policy Decision Points (PDPs). It provides a high performance network protocol (over Thrift) for PEP/PDP interaction, and policy decision and attribute caching. Notifications are provided for policy updates. Moreover, the WSO2 Carbon TryIt tool that comes bundled with the Identity Server lets the user explore the policy impact.

Related links

- See Working with Entitlement for more information on how to use and manage entitlement within the Identity Server.
The identity provider and service provider configurations provide the basis for all actions that happen within the authentication framework and provisioning framework.

### Related links

- See [Adding a Service Provider](#) for more information on how to configure the service provider.
- See [Adding an Identity Provider](#) for more information on how to configure the identity provider.
Inbound provisioning requests can come in the form of SCIM or SOAP.

The System for Cross-domain Identity Management (SCIM) specification is designed to make managing user identities in the WSO2 Identity Server easier. Identity provisioning is a key aspect of any Identity Management Solution. In simple terms, it is to create, maintain and delete user accounts and related identities in one or more systems or applications in response to business processes which are initiated either by humans directly or by automated tasks.

Simple Object Access Protocol (SOAP) is a protocol for exchanging XML-based messages over a network, normally using HTTP. SOAP forms the foundation layer of the Web services stack, providing a basic messaging framework that more abstract layers can build on. SOAP services are defined using Web Services Definition Language (WSDL) and are accessible using a URL that is known as a SOAP endpoint. Here, a SOAP API is used to provision users to the Identity Server.

### Related links

- See [Configuring Inbound Provisioning for a Service Provider](#) for more information on configuring inbound provisioning.
- See [Architecture Provisioning Framework](#) for more information on the various provisioning processes possible in this framework. This includes inbound provisioning.
WSO2 Identity Server implements flexible user store via built-in LDAP (powered by ApacheDS), external LDAP, Microsoft Active Directory or any JDBC database. It provides an API for integrating identity management to any application. WSO2 Identity Server allows tenants/organizations to configure their user stores through the admin console. WSO2 Identity Server supports multiple profiles per user using its flexible profile management feature.

**Related links**

- See [Working with User Stores](#) for more information on how to configure user stores.
- See [Configuring Users, Roles and Permissions](#) for more information on how to work with users and roles.

The user store manager receives provisioning requests from the provisioning framework. These provisioning requests are handled and the relevant user store is updated. The request can affect multiple user stores if the configuration is such. Once the request has been handled, an update is sent back to the framework.

The inbound provisioning component sends SCIM and SOAP provisioning requests on to the user store manager.

The user store manager receives the provisioning request and sends it on to the provisioning framework for outbound provisioning.
### Claim Manager

A claim is a piece of information about a particular subject. It can be anything that the subject is owned by or associated with, such as name, group, preferences, etc. A claim provides a single and general notion to define the identity information related to the subject. Claim-based identity is a common way for any application to acquire the identity information. It provides a consistent approach for all applications by hiding the lower level implementation. Claims are also used in identity propagation, by packaging the claims into one or more tokens (such as SAML). These tokens are then issued by an issuer; commonly known as a security token service (STS).

### Related links

- See [Claim Management](#) for more information on managing claims.
- See [Configuring Claims for a Service Provider](#) for information on how to configure claims on the service provider side.
- See [Configuring Claims for an Identity Provider](#) for information on how to configure claims on the identity provider side.

Claim management does not play a direct role but as a component, it integrates with various other components of the Identity Server. Primarily, it integrates with the following components:

- Authentication framework
- Authorization manager
- Provisioning framework
- User store manager
| XACML | XACML does not play a direct role in the process component, it integrates with various other components in the Identity Server. |

XACML is ideally a part of the authorization manager component but it is depicted separately due to its unique role in the Identity Server architecture. XACML (eXtensible Access Control Markup Language) is an XML-based language for access control that has been standardized by the Technical Committee of the OASIS consortium. XACML is popular as a fine grain authorization method among the community. However, there are aspects of XACML that enable it to surpass being just a fine grain authorization mechanism. XACML describes access control policy language, request/response language and reference architecture. The policy language is used to express access control policies (who can do what when). The request/response language expresses queries about whether a particular access should be allowed (requests) and describes answers to those queries (responses).

| Auditing | Auditing does not play a direct role in the process component, it integrates with various other components in the Identity Server. IS can be configured to produce its components but the following components are used for logging details. |

WSO2 Identity Server provides a comprehensive management console with enterprise-level security. It also comes with a built-in collection and monitoring of standard access and performance statistics. Operational audit and KPI monitoring and management is achieved by integrating with the WSO2 Business Activity Monitor. Further key metrics monitoring and management is achieved with JMX MBeans. WSO2 Identity Server offers a flexible logging support with integration to enterprise logging systems. WSO2 provides a centralized configuration management across different deployment environments with lifecycles and versioning, with integration to WSO2 Governance Registry.

- Authentication framework
- Provisioning framework
- User store manager
<table>
<thead>
<tr>
<th>Identity manager</th>
<th>Identity Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise IT Systems are constantly changing; their perimeters are expanding and their policies keep changing. Therefore, in such a rapidly evolving world, security solutions need to be forward thinking and innovative. They need to be configurable in order to keep pace and adapt to rapid changes. This can be achieved by the identity manager component because it caters to security requirements at hand as well as looking toward the future. It has a very customizable user interface and can be easily implemented in order to ensure maximum security for your system.</td>
<td></td>
</tr>
</tbody>
</table>

**Related links**

- See [User Authentication and Account Management](#) for more information on the features of the identity manager.
The Outbound Provisioning component of the Identity Server can send provisioning requests to applications that support the following connectors.

- SCIM
- SPML
- Google
- Salesforce

These connectors reach out to identity providers who perform the provisioning.

### Related links

- See [Configuring Outbound Provisioning Connectors for an Identity Provider](#) for more information on how to configure outbound provisioning connectors.