Demo-Suite tool for WSO2 Identity Server

WSO2 Identity Server solve lot of enterprise-level security and identity management related problems. This demo-suite tool makes the life easier to who are trying to see the solutions in WSO2 Identity Server.

Getting Start

In order to try out each solution patterns using this demo-suite, follow the below steps.

1. Download WSO2 Identity Server 5.4.0
2. Download the demo-resources.
3. There is a demo.sh file in the demo-resource folder. Run demo.sh script for any solution patterns as follows,

   Ex:

   ./demo.sh solution-01 solution-02

   Or

   ./demo.sh

   And follow the commands in terminal.

   It will install the required artifacts in Identity Server and start a tomcat with required client web applications.

Please refer the last section of this document to further detail for each solution patterns.

Structure of the tool

Here are the main resource folders in demo-suite tool.

demo-resources

- Solutions
- common-resources
- Servers

Solutions

Under this folder we have created all the solutions. As in the screenshot, there is a specific folder and file structure that this tool expect to run a given solution against the Identity Server.

We have created all the installation artifacts for the standard solution patterns under this folder and any one can create a new solution under this folder by following the expected structure as explained below.

Here is the meaning of each files and folders.

solution-05 -> name of the solution pattern which is used as a prefix for all the artifacts to uniquely identify the solution in identity server.
Ex: Service provider will create as “solution-05-lebens.com” , here lebens is the name of client application.

identity-server -> This name represent the identity server.

default -> This name refer the identity-server instance and it should be defined all the hostname, port and other configs under instance name in server-config.yaml file. If we want to start a new Identity Server instance, the we can create different folder in same level of ‘default’ and add that configs in server-config.yaml file.

carbon.super -> This represent the tenant domain name. If we want to create a separate tenant, then we can create tenant domain folder name. Ex: Folder name > “abc.com”

identity-providers -> All the identity provider artifacts should be under this folder. File name is not related to the artifact and it is just a meaningful name only.

service-providers -> All the service provider artifacts should be under this folder. Each service provider has its own inbound type and it is represented by a separate file by following this format [*service provider file name + “_” + protocol name*] => Ex: testserviceprovider_

user-stores -> All the user store artifacts should be under this folder. File name is not related to the artifact and it is just a meaningful name only. It will generate *.h2.db file per user store which is refer by the user stores.

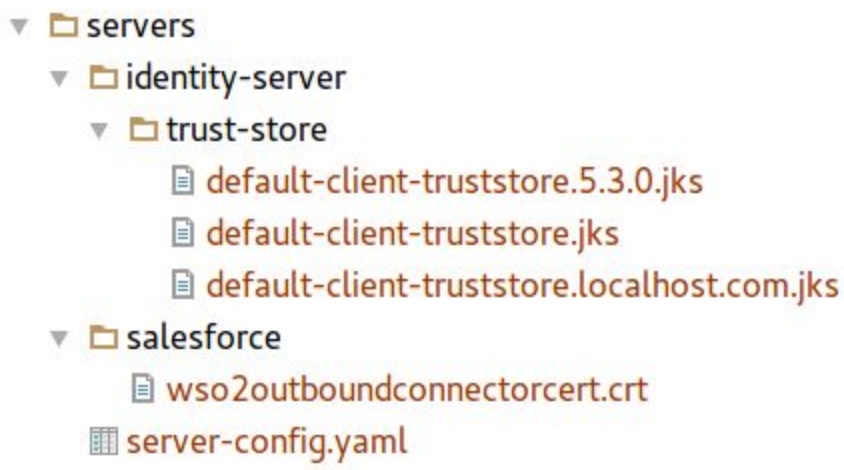
xacml-policy -> All the XACML policies will be installed from this folder. File names are not related to the XACML policies.

user-role -> Here we have simple property files to create user and roles in very simple manner. user.properties and role.properties are the 2 files and we can have any user and roles under these 2 files for each solution pattern.

There is a **solution-config.yaml** file which is provide the instruction to the client tool to know the sequence of the artifact to be run and inactive/active when it is required. Each solution contain this file.

Servers

Under this folder, there is a server-config.yaml to provide the custom configs in each Identity Server instances and the Tomcat. Demo resources provide a tomcat container out of the box and anyone can point to the remote Tomcat server as well by this config file.



Common-Resources

This folder contains all the common resources for solution deployment. All the client web application will be here and DB backup also under this folder.

We have implemented 2 simple client applications which is based on OAuth2 and SAML protocols.

1. **francesca.com**
   This web application authenticate using **SAML** protocol and once we deployed this for a solution pattern, it can be accessed as in below.
   ex:
   Solution : solution-03
   Client application URL :
   http://localhost:8080/solution-03-francesca.comhttp://localhost:8080/solution-03-francesca.com

2. **lebens.com**
   This web application authenticate using **OAuth2** protocol and once we deployed this for a solution pattern, it can be accessed as in below.
   ex:
   Solution : solution-05
   Client application URL : http://localhost:8080/solution-05-lebens.com

3. **lebens-legazy.com**
   This web application authenticate using **OAuth2** protocol and once we deployed this for a solution pattern, it can be accessed as in below.
   ex:
   Solution : solution-23
   Client application URL : http://localhost:8080/solution-23-lebens-legazy.com

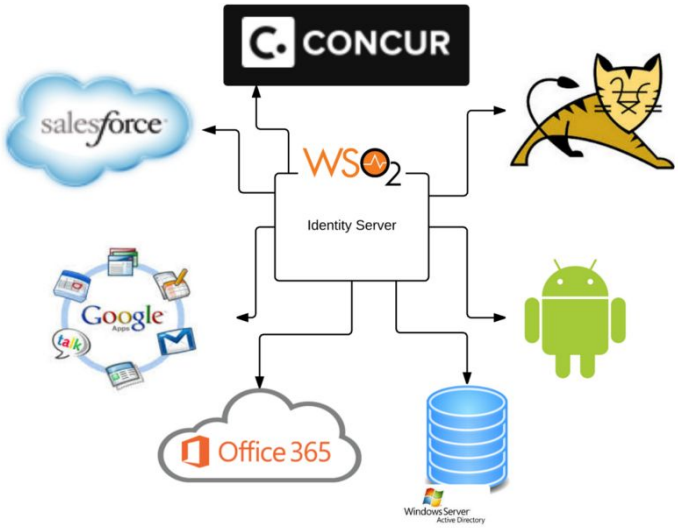
4. **lebens-spa.com**
   This web application authenticate using **OAuth2** protocol and once we deployed this for a solution pattern, it can be accessed as in below.
   ex:
   Solution : solution-17
   Client application URL : http://localhost:8080/solution-17-lebens-spa.com

Solution Pattern Deployment and Results

[solution-01] Single Sign On between multiple heterogeneous identity federation protocols.

Problem :

1. The business users need to access multiple service providers supporting multiple heterogeneous identity federation protocols.
2. Some service providers are on-premise while others are in the cloud. For example Google Apps (SAML 2.0), Salesforce (SAML 2.0), Office 365 (WS-Federation) are cloud based while JIRA, Drupal, Redmine are on-premise service providers.
3. A user logs into any of the service providers should be automatically logged into the rest.



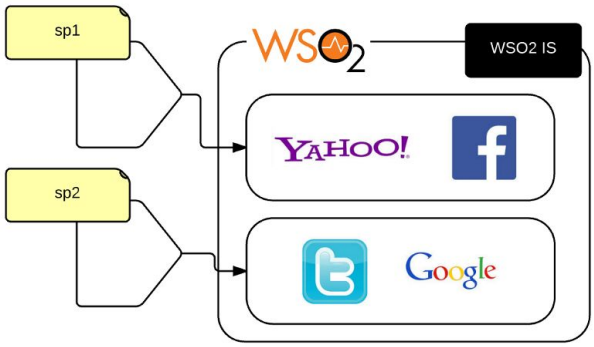
Solution :

1. Installing solution-01-lebens.com and solution-01-francesca.com client applications to the tomcat and register as service providers in Identity Server. Then you can access the following URLs.
2. If someone login to the one service provider, the other one also will get logged in automatically.
   http://localhost:8080/solution-01-francesca.com
   http://localhost:8080/solution-01-lebens.com

[solution-02] Multiple login options by service provider.

Problem :

1. The business users need to access multiple service providers, where each service provider requires different login options. For example login to Google Apps with username/password, while login to Drupal either with username/password or Facebook.



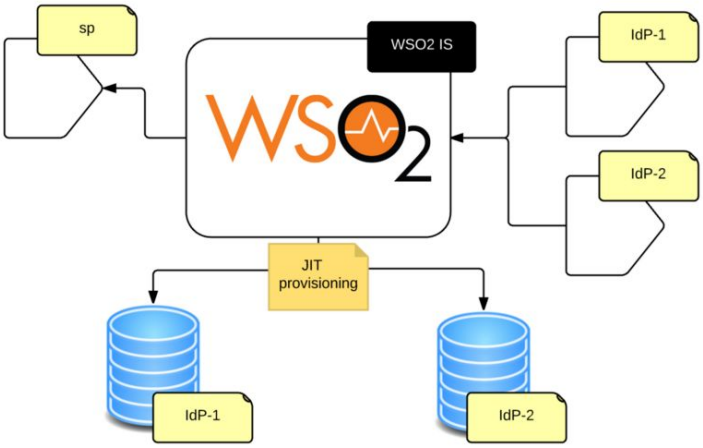
Solution :

1. Represent each service providers in Identity Server and under each service provider, configure local and outbound authentication options appropriately. To configure multiple login options, define an authentication step with the required authenticators.
2. Once you deploy the solution-02, it will install the 2 service providers (francesca.com - SAML and lebens.com - OAuth2) in Identity Server and deploy the 2 client web applications in tomcat as well.
3. Once you try to login, you can have multiple login options for each.
   http://localhost:8080/solution-02-lebens.com
   http://localhost:8080/solution-02-francesca.com

[solution-03] Provision federated users by the identity provider.

Problem :

1. The business users need to login to multiple service providers via multiple identity providers. For example login to Drupal via Facebook or Yahoo! Credentials.
2. Irrespective of the service provider, need to group federated users by the identity provider and store all the user attributes locally. For example, the identity admin should be able to find all the Facebook user or the Yahoo users who have accessed the system (i.e. login to any service provider)

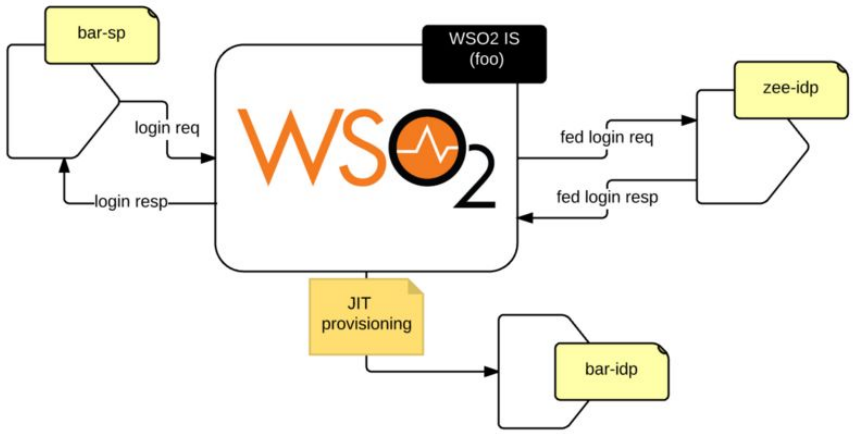


Solution :

1. Deploy WSO2 Identity Server over multiple user stores and name each user store after the name of the corresponding identity provider.
2. Represent each federated identity provider in Identity Server. For example, represent Facebook as an identity provider in Identity Server.
3. Enable JIT provisioning for each identity provider, and pick the user store domain to provision users.
4. Once you deploy the solution-03, it will install the 1 service provider (lebens.com - OAuth2) in Identity Server and deploy the 1 client web applications in tomcat as well. http://localhost:8080/solution-03-lebens.com
5. As in the scenario, we can see these user provisioning y login to the identity console. https://localhost:9443/carbon

---

[solution-04] JIT provision users to cloud service providers.



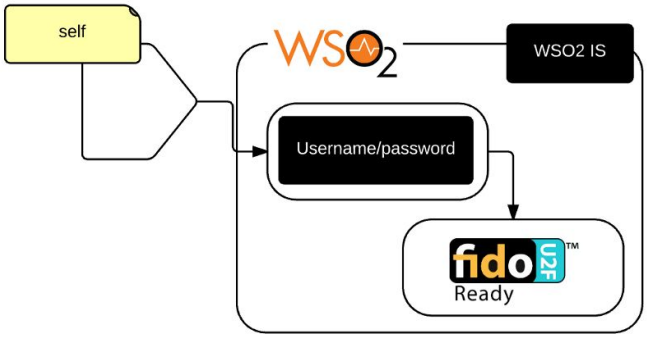
Problem :

1. The company foo has an account with the bar cloud service provider (it can be Google Apps, Salesforce, Workday).
2. The company foo trusts employees from the company zee to login into the bar cloud service provider, under the foo account.
3. For example, foo company wants the users from company zee to login into its Google Apps domain.

Solution :

1. Install lebens.com -> OAuth2 as the service providers.
2. Configured Google as the federated identity provider and salesforce as the outbound connector. Once lebens.com authenticate from the Google, the user will provision to the Salesforce.
3. We can try the scenario from the lebens.com client using below URL. http://localhost:8080/solution-04-lebens.com

[solution-05] Multi-factor authentication for WSO2 Identity Server management console.



Problem :

1. Enable MFA for the WSO2 Identity Server Management Console.
2. In other words, the Identity Server's Management Console itself must be protected with MFA.

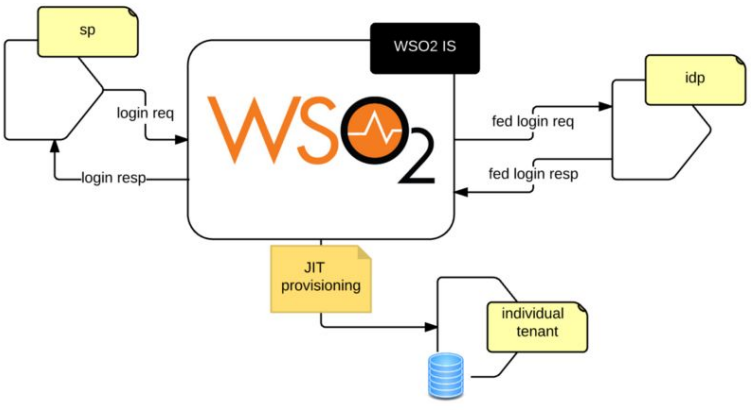
Prerequisites :

1. We have to change the below configuration in Identity Server manually before run the solution-05.
   http://blog.facilelogin.com/2016/03/enabling-mult-factor-authentication-for.html

Solution :

1. Under the service provider configuration, configure multi-step authentication having authenticators, which support MFA in each step.
2. Once we try to login to the Identity Server admin console, it will redirect to the common login endpoint and will allow to have multi factor authentication.
   https://localhost:9443/carbon

[solution-06] Provision federated users to a tenant.



Problem :

1. The business users need to login to multiple service providers via multiple identity providers. For example login to Drupal via Facebook or Yahoo! Credentials.
2. Irrespective of the service provider, need to provision federated users to a single tenant (let's say, individual tenant).

Solution :

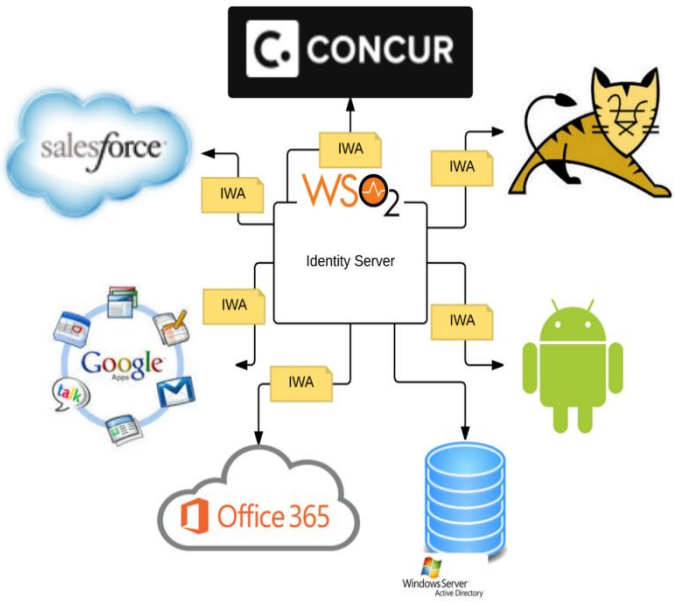
1. Define a user store with CarbonRemoteUserStoreManager in the WSO2 Identity Server pointing to the individual tenant.

2. Represent each federated identity provider in Identity Server. For example, represent Facebook as an identity provider in Identity Server.
3. Enable JIT provisioning for each identity provider, and pick the user store domain(CarbonRemoteUserStoreManager) to provision users.
4. We can try the scenario from the lebens.com client using below URL.
   http://localhost:8080/solution-06-lebens.com

[solution-07] Login to multiple service providers with the current Windows login session.

Problem :

1. The business users need to login to multiple service providers supporting multiple heterogeneous identity federation protocols.
2. Some service providers are on-premise while others are in the cloud.
3. A user logs into his Windows machine and should be able to access any service provider.



Prerequisites :

1. Configure Active Directory in a Windows machine that we need to point from the Identity Server.
2. Follow the steps in

   https://docs.wso2.com/display/IS530/Configuring+IWA+on+Linux

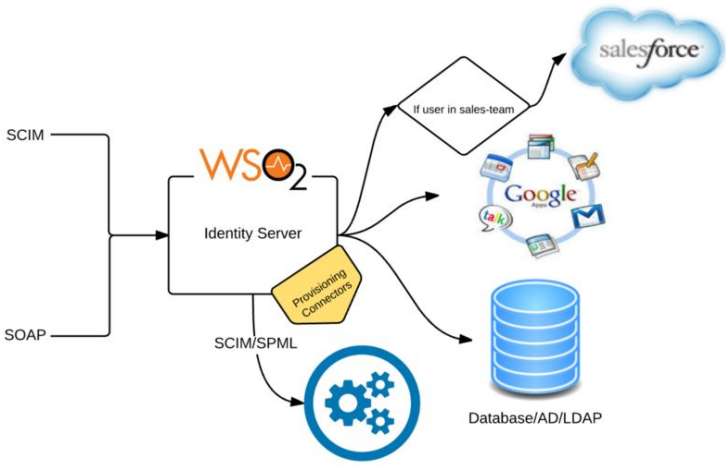
Solution :

1. Install lebens.com -> OAuth2 as the service providers.
2. Once your Windows machine is logged in from given Active Directory, it will allow to login to the service provider which is authenticated from IWA without having the credential.
   http://localhost:8080/solution-07-lebens.com

[solution-08] Rule-based user provisioning.

Problem :

1. The identity admin needs to provision all the employees to Google Apps at the time they join the company.
2. Provision only the employees belong to the sales-team to Salesforce.



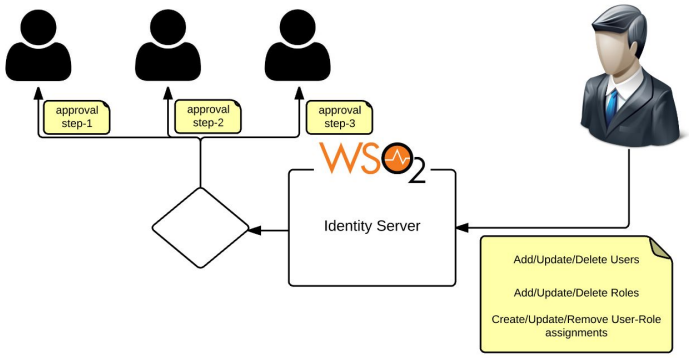
Solution :

1. Install lebens.com -> OAuth2 as the service providers.
2. Represent Salesforce and Google Apps as provisioning identity providers in the WSO2 Identity Server.
3. Under Salesforce Provisioning Identity Provider Configuration, under the Role Configuration, set sales-team-s8 as the role for outbound provisioning.
4. Under the Resident Service Provider configuration, set both Salesforce and Google Apps as provisioning identity providers for outbound provisioning.
5. In this scenario, it will only provision the users to the salesforce that which is belong to the sales-team-s8 only.
   http://localhost:8080/solution-08-lebens.com

[solution-09] User management upon multi-layer approval.

Problem :

1. All the user management operations must be approved by multiple administrators in the enterprise in a hierarchical manner.
2. When an employee joins the company, it has to be approved by a set of administrators while, when the same employee is assigned to the sales team, must be approved by another set of administrators.



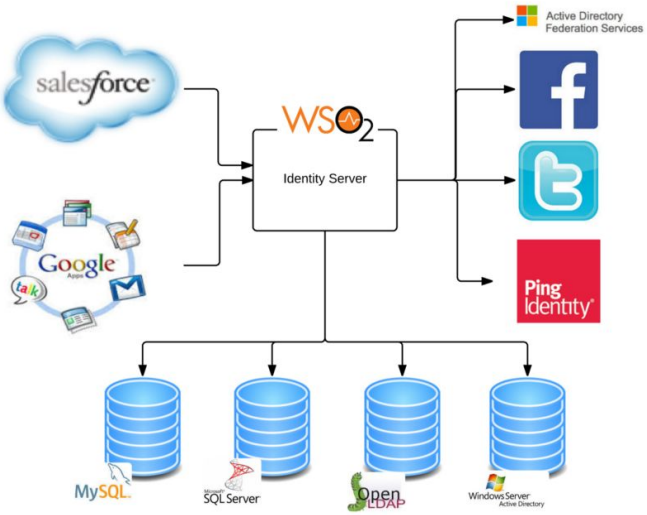
Solution :

1. Create a workflow with multiple steps. In each step specify who should provide the approval.
2. Define a workflow engagement for user management operations and associate the above workflow with it.
3. When defining the workflow, define the criteria for its execution.
4. Login the Identity Server console and try to create an user, the it will add the user as a pending user to the system.
   https://localhost:9443/carbon
5. Now the admin user can login to the dashboard and approve the pending request. Then you can see the user is added to the system.
   https://localhost:9443/dashboard

[solution-11] Identity federation between service providers and identity providers with incompatible identity federation protocols.

Problem :

1. The business users need to login into a SAML service provider with an assertion coming from an OpenID Connect identity provider.
2. In other words, the user is authenticated against an identity provider, which only supports OpenID Connect, but the user needs to login into a service provider, which only supports SAML 2.0.



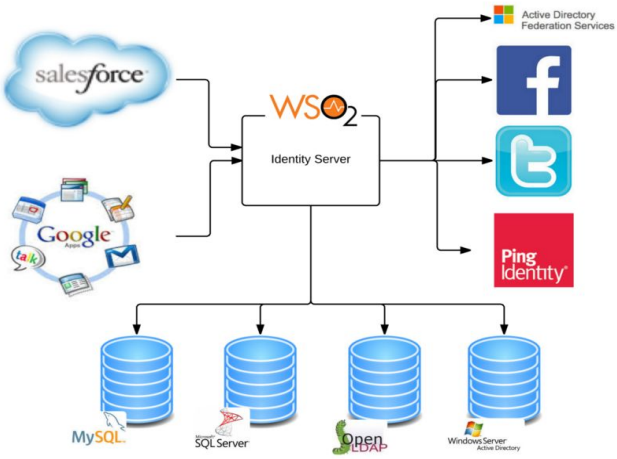
Solution :

1. Represent all the service providers in the WSO2 Identity Server and configure the corresponding inbound authenticators (SAML, OpenID, OIDC, WS-Federation).
2. Represent all the identity providers in the WSO2 Identity Server and configure corresponding federated authenticators (SAML, OpenID, OIDC, WS-Federation).
3. Associate identity providers with service providers, under the Service Provider configuration, under the Local and Outbound Authentication configuration, irrespective of the protocols they support.
4. You can login to the francesca.com which is based on SAML and then it will redirect to the facebook which is based on OIDC.
   http://localhost:8080/solution-11-francesca.com

[solution-12] Claim mapper

Problem :

1. The claim dialect used by the service provider is not compatible with the default claim dialect used by the WSO2 Identity Server.
2. The claim dialect used by the federated (external) identity provider is not compatible with the default claim dialect used by the WSO2 Identity Server.



Solution :

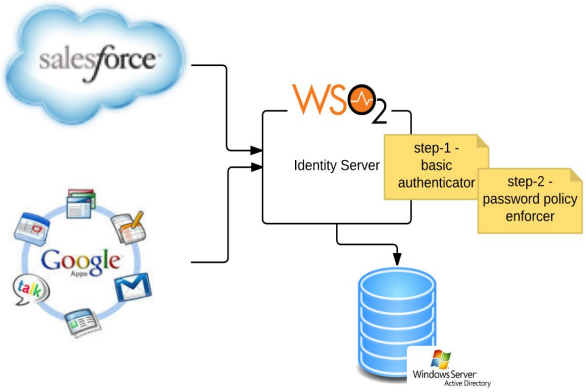
1. Install francesca.com -> SAML as the service providers.
2. For each service provider define custom claims and map them to the WSO2 default claim dialect.
3. Represent all the identity providers in the WSO2 Identity Server and configure corresponding federated authenticators (SAML, OpenID, OIDC, WS-Federation).
4. For each identity provider define custom claims and map them to the WSO2 default claim dialect.
   http://localhost:8080/solution-12-francesca.com

[solution-14]

Enforce password reset for expired passwords during the authentication flow.

Problem :

1. During the authentication flow, enforce to check whether the end-user password is expired and if so, prompt the user to change the password.



Prerequisites :

1. Follow this to setup the Identity Server
   https://docs.wso2.com/display/ISCONNECTORS/Configuring+Password+Policy+Authenticator

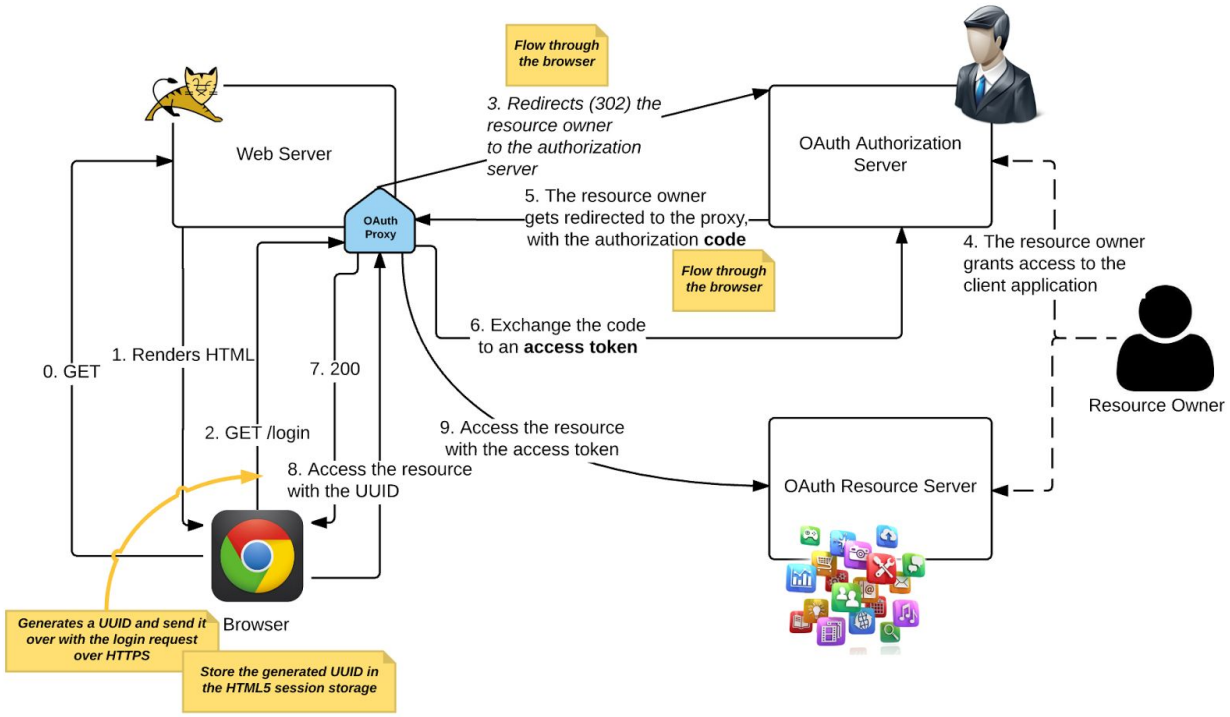
Solution :

1. Configure multi-step authentication for the corresponding service provider.
2. Engage basic authenticator for the first step, which accepts username/password from the end-user.
3. Write a handler (a local authenticator) and engage it in the second step, which will check the validity of the user's password and if it is expired then prompt the user to reset the password.
4. We can try to login to the service provider using following URL http://localhost:8080/solution-14-lebens.com

---

[solution-17]

Single Page Application (SPA) proxy



Problem :

1. Authenticate users to a single page application in a secure manner, via OAuth 2.0.
2. The SPA accessing an OAuth-secured API, the access token must be made invisible to the end-user.
3. The SPA accessing an OAuth-secured API, the client (or the SPA) must be authenticated in a legitimate manner.

Prerequisites :

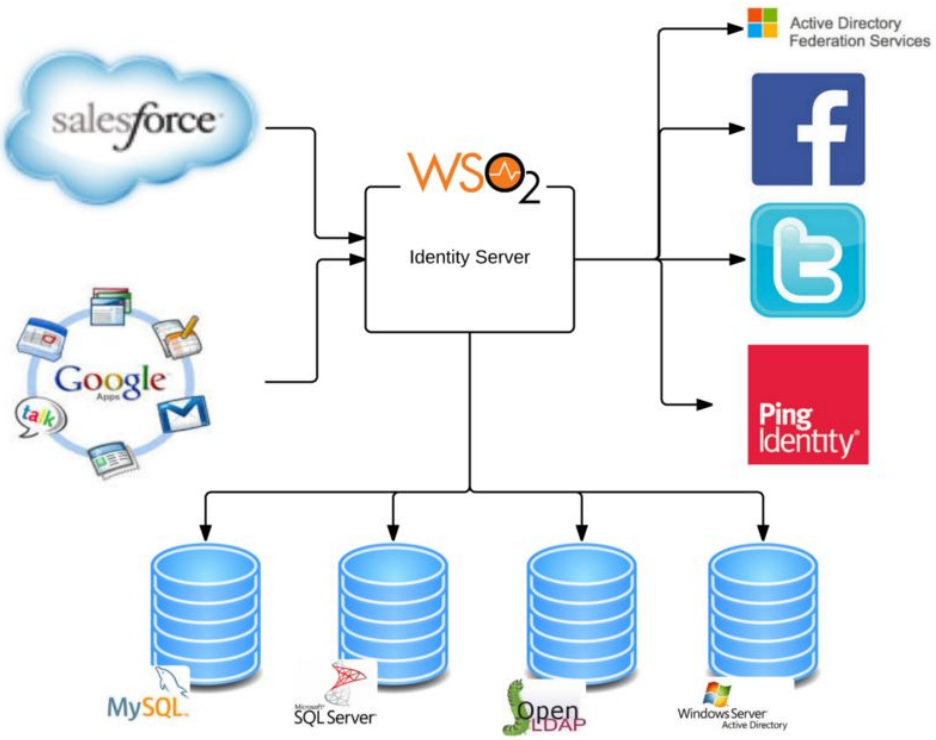
2. Follow this to setup the Identity Server https://docs.wso2.com/display/ISCONNECTORS/Configuring+Password+Policy+Authenticator

Solution :

1. There are multiple ways to secure an SPA and this presentation covers some options: http://www.slideshare.net/prabathsiriwardena/securing-singlepage-applications-with-oauth-20
2. This explains the SPA proxy pattern, where a proxy is introduced, and the calls from the SPA will be routed through the proxy.
3. Build an SPA proxy and deploy it in WSO2 Identity Server. A sample proxy app is available at https://github.com/facilelogin/aratuwa/tree/master/oauth2.0-apps.
4. The SPA proxy must be registered in the WSO2 Identity Server as a service provider, having OAuth inbound authenticator.
5. To make the SPA proxy stateless, the access_token and the id_token obtained from the WSO2 Identity Server (after the OAuth flow) are encrypted and set as a cookie. http://localhost:8080/solution-17-lebens-spa.com

---

[solution-18]Fine-grained access control for service providers



Problem :

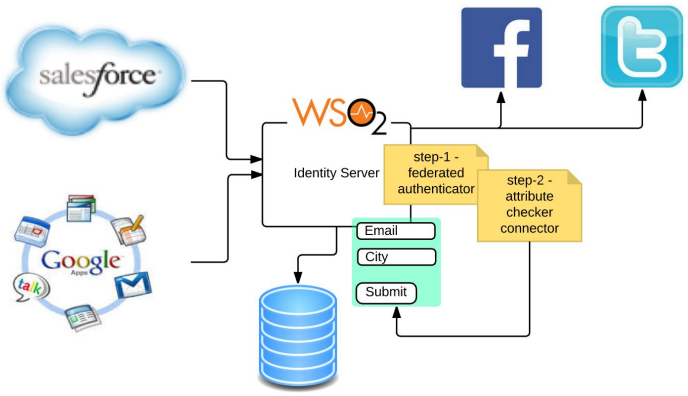
4. The business users need to access multiple service providers supporting multiple heterogeneous identity federation protocols.
5. Each service provider needs to define an authorization policy at the identity provider, to decide whether a given user is eligible to log into the corresponding service provider.
6. For example, one service provider may have a requirement that only the admin users will be able to login into the system after 6 PM.
7. Another service provider may have a requirement that only the users from North America should be able to login into the system.

Solution :

6. Install francesca.com -> SAML as the service providers.
7. Create finance and sales roles.
8. Create solution-18-user1 for finance and solution-18-user2 for sales with admin123 password for all.
9. Create XACML policies for user-store based authorization for Service Providers which is only allow to the finance users.
10. Can login to the francesca.com using finance user's credential
11. Can't login to the francesca.com using sales user's credential
    http://localhost:8080/solution-18-francesca.com

[solution-21] Enforce users to provide missing required attributes while getting JIT provisioned to the local system.

Problem :

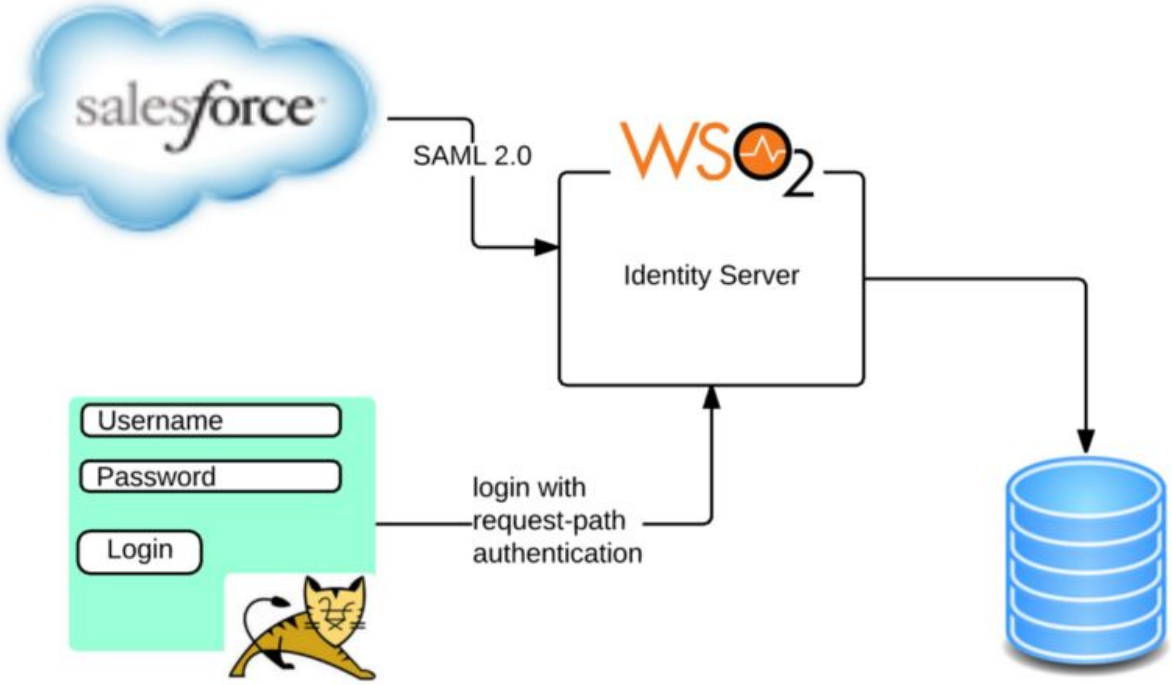


1. The business users need to access multiple service providers via federated identity provider (i.e Facebook, Yahoo, Google).
2. Need to JIT provision all the user coming from federated identity providers with a predefined set of attributes.
3. If any required attributes are missing in the authentication response from the federated identity provider, the system should present a UI to the user to provide those.

Solution :

1. Install francesca.com -> SAML as the service providers.
2. There are 2 additional claim requested by the Service Provider which is not provided by the Federated IDP. So once user get authenticated from the federated IDP, Identity Server will prompt an input screen to collect those additional attributes itself.
   http://localhost:8080/solution-21-francesca.com

[solution-23] Single Sign On between a legacy web app, which cannot change the user interface and service providers, which support standard SSO protocols.



Problem :

1. The business users need to access a service provider,where its UI cannot be changed. The users need to provide their user credentials to the current login form of the service provider.
2. Once the user logs into the above service provider, and then clicks on a link to another service (which follows a standard SSO protocol), the user should be automatically logged in. The vice-versa is not true.

Solution :

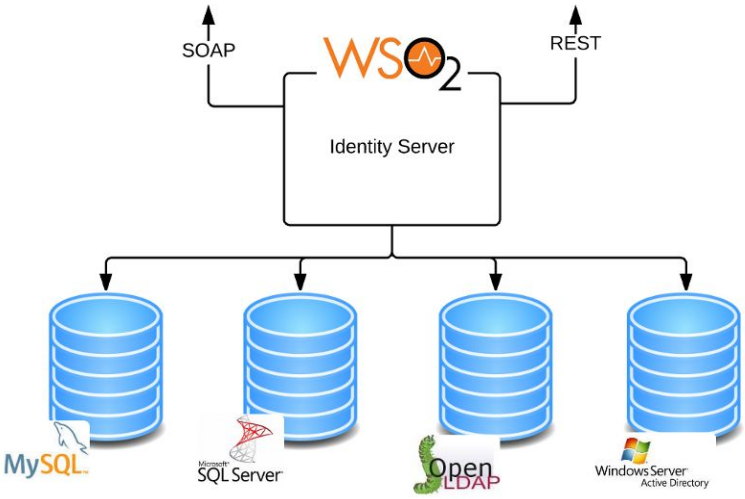
1. Deploy WSO2 Identity Server as the Identity Provider and register all the service providers with standard inbound authenticators (including the legacy app).
2. For the legacy web app, which does not want to change the UI of the login form, enable basic auth request path authenticator, under the Local and Outbound Authentication configuration.
3. Once the legacy app accepts the user credentials from its login form, post them along with the SSO request (SAML 2.0/OIDC) to the WSO2 Identity Server.
4. The WSO2 Identity Server will validate the credentials embedded in the SSO request and if valid, will issue an SSO response and the user will be redirected back to the legacy application. The complete redirection process will be almost transparent to the user.
5. When the same user tries to log in to another service provider, the user will be automatically authenticated, as the previous step created a web session for the logged in user, under the WSO2 Identity Server domain.
   http://localhost:8080/solution-23-lebens-legazy.com

[solution-26]

User administration operations from a third-party web app.

Problem :

1. A third party web app needs to perform all user management operations such as all CRUD operations on users and roles, user/role assignments and password recovery, without having to deal directly with underlying user stores (LDAP, AD, JDBC).

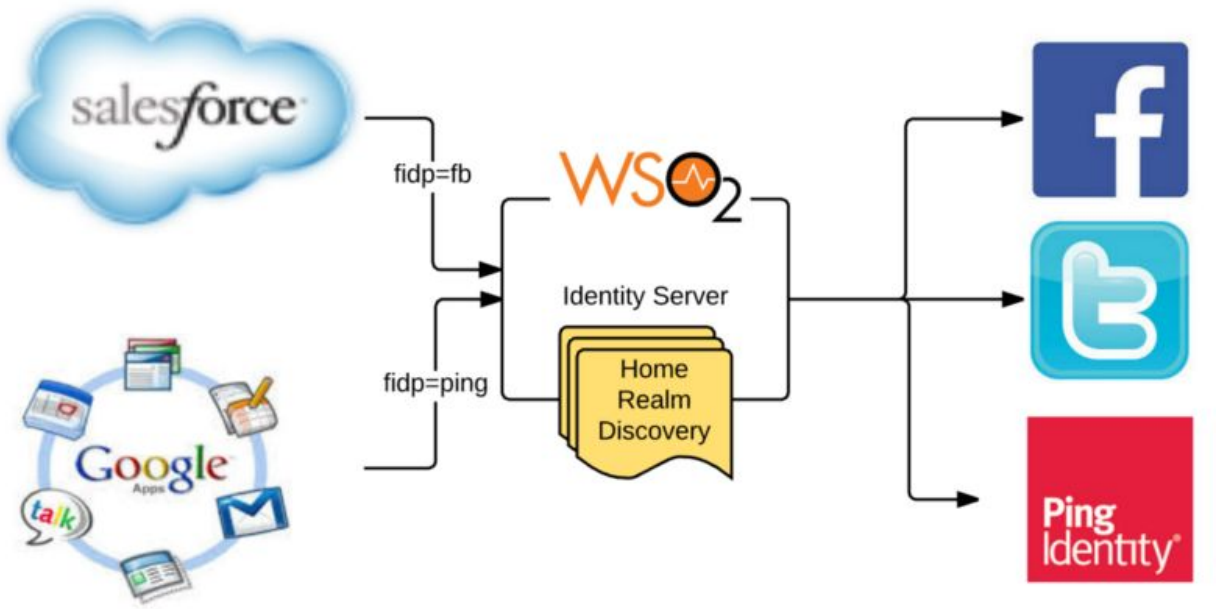


Solution :

1. Deploy the WSO2 Identity Server over the required set of user stores.
2. The WSO2 Identity Server exposes a set of REST endpoints as well as SOAP-based services for user management, the web app just need to talk to these endpoints, without having to deal directly with underlying user stores (LDAP, AD, JDBC).
   http://localhost:8080/solution-26-lebens.com

[solution-28]

Home realm discovery.



Problem :

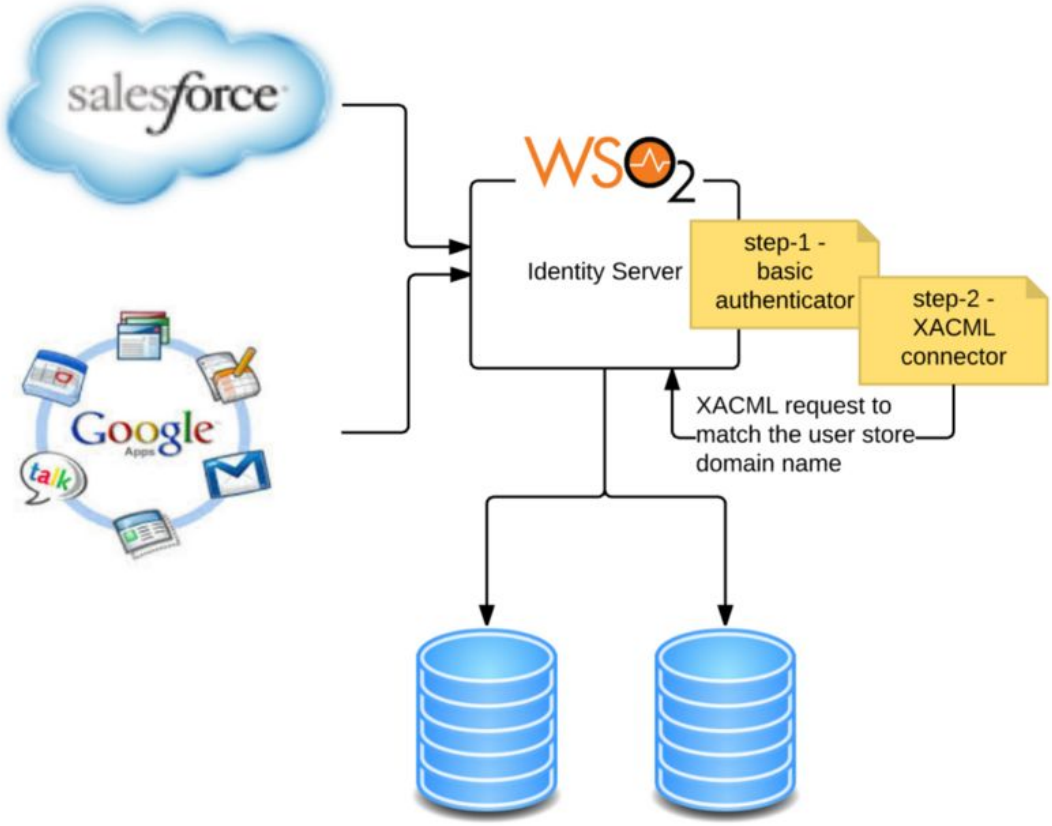
1. The business users need to login to multiple service providers via multiple identity providers.
2. Rather than providing a multi-login option page with all the available identity provider, once redirected from the service provider, the system should find out who the identity provider corresponding to the user and directly redirect the user there.

Solution :

3. Install francesca.com -> SAML and lebens.com -> OAuth2 as the service providers which is in two different protocols.
4. This lebens.com has the home realm discovery parameter for the request url and it will not show the multi-option once the user try to get authenticate.
5. But if we try to login to the francesca.com , it will show the multi-options.
   http://localhost:8080/solution-28-francesca.com

[solution-29]

Service provider-specific user stores



Problem :

1. The business users need to access multiple service providers supporting multiple heterogeneous identity federation protocols.
2. When the user gets redirected to the identity provider, the users only belong to the user stores specified by the corresponding service provider, should be able to login or get an authentication assertion.
3. In other words, each service provider should be able to specify from which user store it accepts users.

Solution :

1. Installed XACML policy to enforce this rules.
   http://localhost:8080/solution-29-francesca.com