|  |
| --- |
| In these Study Notes TOPIC to know to Pass the exam 🡪 is greenMaterial to study in Red |

TOPIC 🡪 Connected Apps (Knowing the settings is important, prepare well this section you will find many configuration questions)

<https://help.salesforce.com/articleView?id=connected_app_create.htm&type=0>

Key Settings

A)Basic Information --> connected app name / API NAme

B)API (Enable OAuth Settings)

--callback URL (e.g. https://axiomsso.herokuapp.com/OAuth2HandleAuthCode.action)

-- If use JWT OAuth flow --> select Use Digital Signatures

--OAuth scopes

-- Require Secret for Web Server Flow (in case web server flow can keep consumer secret)

C) Web App Settings

 -- start URL for your app to direct users to a specific location after they’ve authenticated

 --Enable SAML

D) Custom Connected App Handler

-- Customize the behavior of a connected app with Apex

E) Mobile App Settings

--Mobile Start URL- Enter the mobile start URL to direct users to a specific location when the app is accessed from a mobile device

--PIN Protect

-- App Platform

--Private App : Y/N

F) Canvas App Settings

-- Canvas apps that the org’s Salesforce admin installed.

-- Canvas personal apps that users installed across orgs

TOPIC 🡪 SAML related errors and troubleshooting

Troubleshooting Single Sign-On

If SSO is not working for you there are a few ways you can troubleshoot the issue.

1. Check the Force.com login history for errors by going to Setup -> Manage Users -> Login History. If you do not see a failed login attempt then you probably have something misconfigured with the Federation ID. The Force.com platform cannot match your user from the SAML assertion so the failed login attempt in not even recorded. Check the Force.com help for details on SSO error messages. Remember that an Assertion can contains
	* + A) Salesforce username
		+ B) Federation ID from the User object
		+ C) User ID from the User object
2. You can paste your SAML assertion into the SAML Assertion Validator on the Single Sign-On Settings page. This gives you a good view of what's going on and can possibly point you in the right direction.
3. Try downloading the Firefox add-on, Live HTTP Headers, so that you can easily view the packets that are being sent during the login process.

Also If a user fails to log in

Reviewing the SAML Login History

<https://developer.salesforce.com/docs/atlas.en-us.sso.meta/sso/sso_saml_login_history.htm>

Assertion Expired: An assertion’s timestamp is more than five minutes old.

Assertion Invalid: An assertion is not valid. For example, the <Subject> element of an assertion might be missing.

Audience Invalid: The value specified in <Audience> must be https://saml.salesforce.com.

Configuration Error/Perm Disabled: Something is wrong with the SAML configuration in Salesforce. For example, the uploaded certificate might be corrupted, or the organization preference might have been turned off. get a sample SAML assertion from your identity provider, and then click SAML Assertion Validator.

Issuer Mismatched: The issuer or entity ID specified in an assertion does not match the issuer specified in your Salesforce configuration.

Recipient Mismatched: The recipient specified in an assertion does not match the recipient specified in your Salesforce configuration.

Replay Detected: The same assertion ID was used more than once. Assertion IDs must be unique within an organization.

Signature Invalid: The signature in an assertion cannot be validated by the certificate in your Salesforce configuration.

Subject Confirmation Error: The <Subject> specified in the assertion does not match the SAML configuration in Salesforce.

* TOPIC 🡪 OAuth Scopes and their uses Types of OAuth Flows (know each of them really well since there will be tons of questions on these)
* api- Allows access to the current, logged-in user’s account using APIs, such as REST API and Bulk API.
* chatter\_api- Allows access to Chatter REST API resources only.
* custom\_permissions - Allows access to the custom permissions in an organization
* full - Allows access to all data accessible by the logged-in user, and encompasses all other scopes. full does not return a refresh token.
* id - Allows access to the identity URL service.
* openid - Allows access to the current, logged in user’s unique identifier for OpenID Connect apps.
* refresh\_token Allows a refresh token to be returned if you are eligible to receive one.
* visualforce Allows access to Visualforce pages.
* web Allows the ability to use the access\_token on the Web. This also includes Visualforce

<https://help.salesforce.com/articleView?id=remoteaccess_oauth_scopes.htm&type=0>

TOPIC 🡪 My Domain and its uses

Add a subdomain to your Salesforce org URL with the My Domain Salesforce feature.

<https://help.salesforce.com/articleView?id=domain_name_overview.htm&language=en&type=0>

My Domain is required before you can use these Salesforce features:

1. Single sign-on (SSO) with external identity providers
2. Social sign-on with authentication providers, such as Google and Facebook
3. Lightning components in Lightning component tabs, Lightning Pages, the Lightning App Builder, or standalone apps

Also Before you can enable Salesforce as an identity provider, you have to set up a domain.

TOPIC 🡪 Two Factor Authentication (2FA)

<https://help.salesforce.com/articleView?id=security_overview_2fa.htm&language=en_US&type=0>

We use the highest-priority verification method available for each user. In order of priority, the methods are:

* Verification via push notification or location-based automated verification with the Salesforce Authenticator mobile app (version 2 or later) connected to the user’s account.
* Verification via a U2F security key registered with the user’s account.
* Verification code generated by a mobile authenticator app connected to the user’s account.
* Verification code sent via SMS to the user’s verified mobile phone.
* Verification code sent via email to the user’s email address.

To Set up --> Create a permission set for two-factor authentication

System permission --> Select Two-Factor Authentication for User Interface Logins.

TOPIC 🡪 Identity Provider and Service Provider related scenarios

<https://help.salesforce.com/articleView?id=identity_provider_about.htm&language=en&type=0>

<https://help.salesforce.com/articleView?id=identity_provider_examples.htm&language=en_US&type=0>

Delegated Authentication

* manage delegated authentication at the permission level
* When a user tries to log in—either online or using the API—Salesforce validates the username and checks the user’s permissions and access settings.
* If the user has the “Is Single Sign-On Enabled” user permission, Salesforce doesn’t validate the username and password. Instead, a web services call is made to the user’s org asking it to validate the username and password.
* -->Salesforce doesn’t store, log, or view the password
* The web services call passes the username, password, and sourceIp to your web service. Source Ip is the IP address where the login request originated. You must create and deploy an implementation of the web service that Salesforce servers can access
* Your web service implementation validates the passed information and returns either true or false.
* If the response is true, the login process continues, a new session is generated, and the user proceeds to the app. If false, the user gets an error message that the username and password combination is invalid

<https://help.salesforce.com/articleView?id=sso_delauthentication.htm&language=en_US&type=0>

TOPIC 🡪 SAML Federated Authentication

Benefits of Implementing SSO

* Improved productivity - Not having to enter a password each time a user needs to access a resource saves time and makes users more productive.
* Reduce frustration of multiple log-on events and forgotten passwords - Users only have one password to remember and update,
* Increased adoption - SSO reduces the barriers of use for resources. Since it is easier to access applications, users will start using them more.
* Centralized user access control - A single registry of user identities allows quick and easy provisioning and deactivating of users.
* Improved reporting and monitoring - A single repository for auditing and logging access to resources provides streamlined regulatory compliance.
* Increased security - A secure, enterprise-wide infrastructure with common password and security policies that can be centrally managed and secured.
* Uniform security layer - SAML is platform agnostic allowing enterprise architects to implement a uniform security layer with existing assets.
* Reduced helpdesk costs - Fewer helpdesk calls for password resets relates directly to bottom-line savings.

<https://developer.salesforce.com/page/Single_Sign-On_with_SAML_on_Force.com>

TOPIC 🡪 Community Self Registration: Apex Registration Handler related scenarios

Update the CommunitiesSelfRegController to customize the default self-registration process for your community. You can use the same controller for the default self-registration page (CommunitiesSelfReg) or a custom Visualforce or Community Builder self-registration page.

* You can configure self-registration entirely in Community Management. This customization is recommended only if you want to modify the self-registration behavior beyond the defaults, if you have more then one community in your organization, or if you are using a custom self-registration page.
* Click Edit next to CommunitiesSelfRegController.
* Optionally, enter a value for ProfileId to define the type of profile the user should be assigned.
* Enter the account ID for the partner or customer account that users who self register should be associated with
* If you’re enabling self-registration for multiple communities, add code to create appropriate types of users for each community

TOPIC 🡪 Login Flows

Custom Login Flows: Login flows allow administrators to build post-authentication processes to match their business practices, associate the flow with a user profile, and send the user through that flow when logging in. Use login flows to collect registration information from users, provide a terms of service acceptance form, prompt the user for a second factor of authentication, and other customization.

Use the Flow Designer to create login flows, and then associate those flows with specific profiles in your organization

1. Create a Login Flow: Use the Cloud Flow Designer to build a login flow process, then associate the finished flow with a profile.
2. Connect a Login Flow to a Profile: After you create a login flow in Flow Designer and activate the flow, you associate it with a profile in your organization. Users with that profile are then directed to the login flow.

TOPIC 🡪 Just-in-Time Provisioning

With Just-in-Time provisioning, you can use a SAML assertion to create regular and portal users on the fly the first time they try to log in. This eliminates the need to create user accounts in advance

 Just-in-Time provisioning works with your SAML identity provider to pass the correct user information to Salesforce in a SAML 2.0 assertion.

 Because Just-in-Time provisioning uses SAML to communicate, your organization must have SAML-based single sign-on enabled

 Just-in-Time provisioning can offer the following advantages

* -Reduced Administrative Costs
* -Increased User Adoption
* -Increased Security

Just-in-Time Provisioning Requirements and SAML Assertion Fields

To correctly identify which object to create in Salesforce, you must use the User. prefix for all fields passed in the SAML assertion

 Just-in-Time Provisioning for Communities

With Just-in-Time (JIT) provisioning for Communities, you can use a SAML assertion to create customer and partner community users on the fly the first time they try to log in from an identity provider.

but there are two basic approaches that can be followed for deprovisioning

Deprovisioning through a custom attribute:

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.

TOPIC 🡪 Salesforce1 and SSO

Single Sign-On for Desktop and Mobile Applications using SAML and OAuth

Combining SAML and OAuth: layering them together in order to achieve SAML based single sign-on for OAuth enabled desktop and mobile applications

<https://developer.salesforce.com/page/Single_Sign-On_for_Desktop_and_Mobile_Applications_using_SAML_and_OAuth>

TOPIC 🡪 External Identity, Identity, Customer Community, Customer Community Plus, Salesforce Platform licenses and uses

<https://help.salesforce.com/articleView?id=users_license_types_communities.htm&language=en_US&type=0>

TOPIC 🡪 CA-Signed Certificates and Self-Signed Certificates

<https://help.salesforce.com/articleView?id=security_keys_about.htm&type=0>

Choose an API client certificate based on the remote endpoint you connect to. Some endpoint servers require a certificate chain that is trusted by a certificate authority; others are fine with directly trusting a self-signed certificate.

* Generate a Self-Signed Certificate: Generate a certificate signed by Salesforce to show that communications purporting to come from your organization are really coming from there.
* Generate a Certificate Signed by a Certificate Authority: A certificate authority-signed (CA-signed) certificate can be a more authoritative way to prove that your org’s data communications are genuine. You can generate this type of certificate and upload it to Salesforce.

<https://help.salesforce.com/articleView?id=security_keys_about.htm&type=0>

TOPIC 🡪 Active Directory (AD) and Identity Connect

* Identity Connect
* Identity Connect integrates Microsoft Active Directory with Salesforce via a service that runs on either Windows or Linux platforms. It gives AD users single sign-on access to Salesforce. When syncing AD users, the identity service provider can be either Salesforce or Identity Connect.
* <https://developer.salesforce.com/docs/atlas.en-us.identityImplGuide.meta/identityImplGuide/identityconnect_about.htm>

TOPIC 🡪 Concepts like Access Token, Refresh Token, Relay State, Start URL, Callback URL

Authorization Code: An authorization code is a short-lived token created by the authorization server and passed to the client application via the browser. only for (Web Server Flow)

Access Token - The access token is used by the client to make authenticated requests on behalf of the end user .

Refresh Token - The refresh token may have an indefinite lifetime, persisting for an admin-configured interval or until explicitly revoked by the end-use

ID Token

OpenID Connect defines the ID token, a signed data structure that contains authenticated user attributes including a unique identifier for the end-user, the time at which the token was issued, and an identifier for the client application that requested the token. The ID token is encoded as a JSON Web Token (JWT).

Relay State -

RelayState will be sent as an HTTP Parameter along side a SAML AuthnRequest ( per the SAML standard ).RelayState originally sent by the Service Provider. The echoing of RelayState is critical to the success of the protocol, as this is what allows the user to be returned to the originally requested resource

Start URL - the start URL is the page the user attempted to access before they were authenticated.

Callback URL - The 'Callback URL' field, also known as the 'redirect URI', is an endpoint in your application to which Force.com can redirect the user's browser with an authentication code or access token

TOPIC 🡪 Identity and Access Management:

* Know details of each OAuth flow. Know details of differences between different flows. I found Axiom tool very useful for Web Server flow. Read about all attributes that are passed in OAuth flows, specially Scope and grant type.

Configure an OpenID Connect Authentication Provider

You can use any third-party web app that implements the server side of the OpenID Connect protocol, such as Amazon, Google, and PayPal, as an authentication provider.

Do the following to set up a custom authentication provider for SSO.

A) Configure the service provider website.

- Before you can configure a web app for your Salesforce org, you must register it with your service provider.

B) Configure an Aouth. Provider from the list, Additionally configure :

Scope—Customizes the permissions requested from the third party.

Site—Enables the provider to be used with a site.

StartURL—Sends the user to a specified location after authentication.

C) Create a registration handler using Apex.

You must implement a registration handler to use authentication providers for SSO. The Apex registration handler class must implement the Auth.RegistrationHandler interface, which defines two methods. Salesforce invokes the appropriate method on callback, depending on whether the user has used this provider before or not. When you create the authentication provider, you can automatically create an Apex template class for testing purposes

Define the authentication provider in your org.

<https://help.salesforce.com/articleView?id=sso_provider_google.htm&type=0&language=en_US>

* **TOPIC 🡪 Read about JIT and registration handler.**

SamlJitHandler Interface

Use this interface to control and customize Just-in-Time user provisioning logic during SAML single sign-on.

To use custom logic for user provisioning during SAML single sign-on, you must create a class that implements Auth.SamlJitHandler. This allows you to incorporate organization-specific logic (such as populating custom fields) when users log in to Salesforce with single sign-on. Keep in mind that your class must perform the logic of creating and updating user data as appropriate, including any associated account and contact records.

* **TOPIC 🡪 Know different authentication methods, specially Delegated Authentication and best practices.**

# Best Practices for Implementing Single Sign-On

Delegated Authentication Best Practices

* Your org’s implementation of the web service must be accessible by Salesforce servers, so you must deploy the web service on a server in your DMZ
* If Salesforce and your system can’t connect, or if the request takes longer than 10 seconds to process, the login attempt fails
* Namespaces, element names, and capitalization must be exact in SOAP request
* For security reasons, make your web service available by TLS.A certificate from a trusted provider, such as Verisign or Thawte, is required.
* The IP address that originated the login request is sourceIp. Use this information to restrict access based on the user’s location
* We recommend that you don’t enable SSO for Salesforce admins
* We recommend that you use a Developer Edition account or a sandbox when developing a SSO solution before implementing it in your org

Federated Authentication Using SAML Best Practices

* -Get the Salesforce login URL from the Single Sign On Settings configuration page and enter it in the corresponding configuration parameter of your identity provider. Sometimes, the setting is called the recipient URL.
* -Salesforce allows a maximum of 3 minutes for clock skew with your IDP server. Make sure that your server’s clock is up to date.
* If you can’t log in with SAML assertion, check the login history and note the error message. Use the SAML Assertion Validator on the Single Sign On Settings configuration page to troubleshoot.
* -Map your orgs internal usernames and Salesforce usernames. To map the names, you can add a unique identifier to the FederationIdentifier field of each Salesforce user.
* -Use the My Domain feature to prevent users from logging in to Salesforce directly
* -Sandbox copies are made with federated authentication with SAML disabled. Any configuration information is preserve

-Your identity provider must allow you to set the service provider’s audience URL. The value must match the Entity ID value in the SSO configuration.

* **TOPIC 🡪 Identity Connect and its capabilities.**

<https://org62.my.salesforce.com/_ui/core/userprofile/UserProfilePage?u=0050M00000D0YAK&tab=sfdc.ProfilePlatformFeed&fId=0D50M00002z3MA4>

TOPIC 🡪  **Prepare well and understand the oAuth flows completely including scope, tokens etc. There were some questions on scope. Understand clearly the various flows: Web Server, User Agent, Username/Password, JWT Bearer and SAML Bearer.**

Web Server - users can authorize your web application to access their data. This is the OAuth 2.0 authorization code grant type.

User-Agent - users can authorize your application to access their data, leveraging browser for authentication - the OAuth 2.0 implicit grant type.

JWT Bearer Token Flow - your app can re-use an existing authorization by supplying a signed JSON Web Token (JWT)

 grant\_type Set this to urn:ietf:params:oauth:grant-type:jwt-bearer

SAML Bearer Assertion Flow - your app can also re-use an existing authorization by supplying a signed SAML 2.0 Assertion

 urn:ietf:params:oauth:grant-type:saml2-bearer

SAML Assertion Flow - your app can federate with the API using a SAML Assertion, in much the same way as you would configure SSO -- > In contrast with the SAML Bearer Assertion Flow, you do not have to create a connected app to use this assertion flow. The SAML Assertion Flow uses the SAML settings configured in your org for single sign-on

 grant\_type Set this to assertion

Username and Password - your application can authenticate using an 'API user's credentials - the resource owner password credentials grant type.

 grant\_type Set this to password.

Refresh Token flow, for renewing tokens issued by the web server or user-agent flows,

 grant\_type=refresh\_token

**TOPIC 🡪 Identity connect and what attributes it can synch between AD and Salesforce**

https://www.youtube.com/watch?v=Vl3SSydtwi0

Icentity connect server Admin Console use Oauth --> need connected app

Connect App to Identity connect plugin installed on-premises

Map USER Attribute in the Identity Connect Admin console (on premise server) - Synch users

Use SAML to log users into Salesforce

**TOPIC 🡪 App launcher and what you need to do, in terms of settings & steps etc.**

The App Launcher presents users with logos that link to their on-premise applications, connected apps,

and Salesforce apps, all from a unified user interface. Administrators can set the default app order for

their organizations.

Single Sign On

Profile Based

- Authorize user via Profile and Permission Set

- Tabset Automatically show up in App Launcher

**TOPIC 🡪 Understand clearly the canvas app authentication: Signed request, oAuth, and SAML?**

Signed request—The default method of authentication for canvas apps. The signed request authorization flow varies depending on whether you configure the canvas app so that the administrator gives users access to the canvas app or if users can self-authorize. The signed request containing the consumer key, access token, and other contextual information is provided to the canvas app if the administrator has allowed access to the canvas app for the user or if the user has approved the canvas app via the approve/deny OAuth flow.

OAuth 2.0—Canvas apps can use the OAuth 2.0 protocol to authenticate and acquire access tokens.

Force.com Canvas supports OAuth 2.0 for authorization. When using OAuth, you have two options:

A) Web Server OAuth Authentication Flow

B) User-Agent OAuth Authentication Flow

SAML Single Sign-On for Canvas Apps

Whether you use signed request or OAuth authorization, you can use SAML-based single sign-on (SSO) to provide your users with a seamless authentication flow

**TOPIC 🡪 Describe why Two Factor Authentication is important and strategies for implementing it in Salesforce.**

<https://help.salesforce.com/articleView?id=security_overview_2fa.htm&language=en_US&type=0>

We use the highest-priority verification method available for each user. In order of priority, the methods are:

1. Verification via push notification or location-based automated verification with the Salesforce Authenticator mobile app (version 2 or later) connected to the user’s account.
2. Verification via a U2F security key registered with the user’s account.
3. Verification code generated by a mobile authenticator app connected to the user’s account.
4. Verification code sent via SMS to the user’s verified mobile phone.
5. Verification code sent via email to the user’s email address.