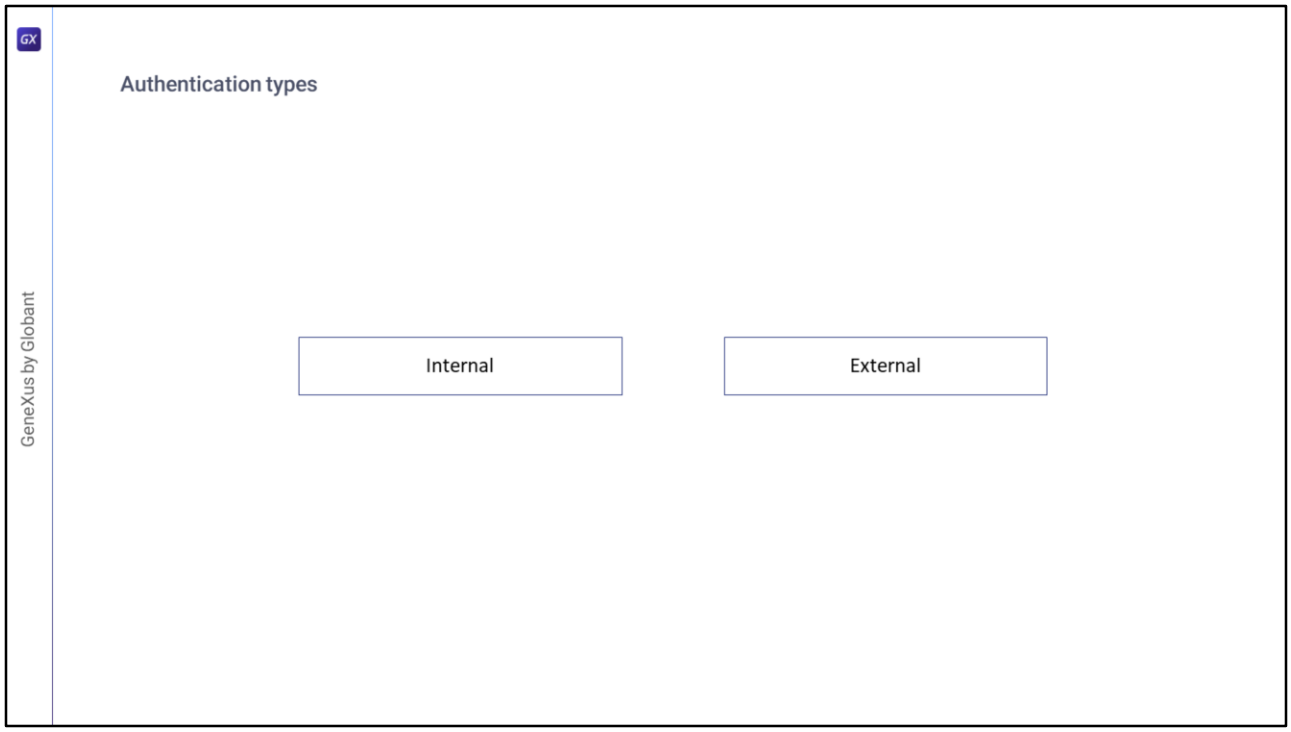

GeneXus by Globant
GeneXus™
by Globant
training.genexus.com

GeneXus by Globant

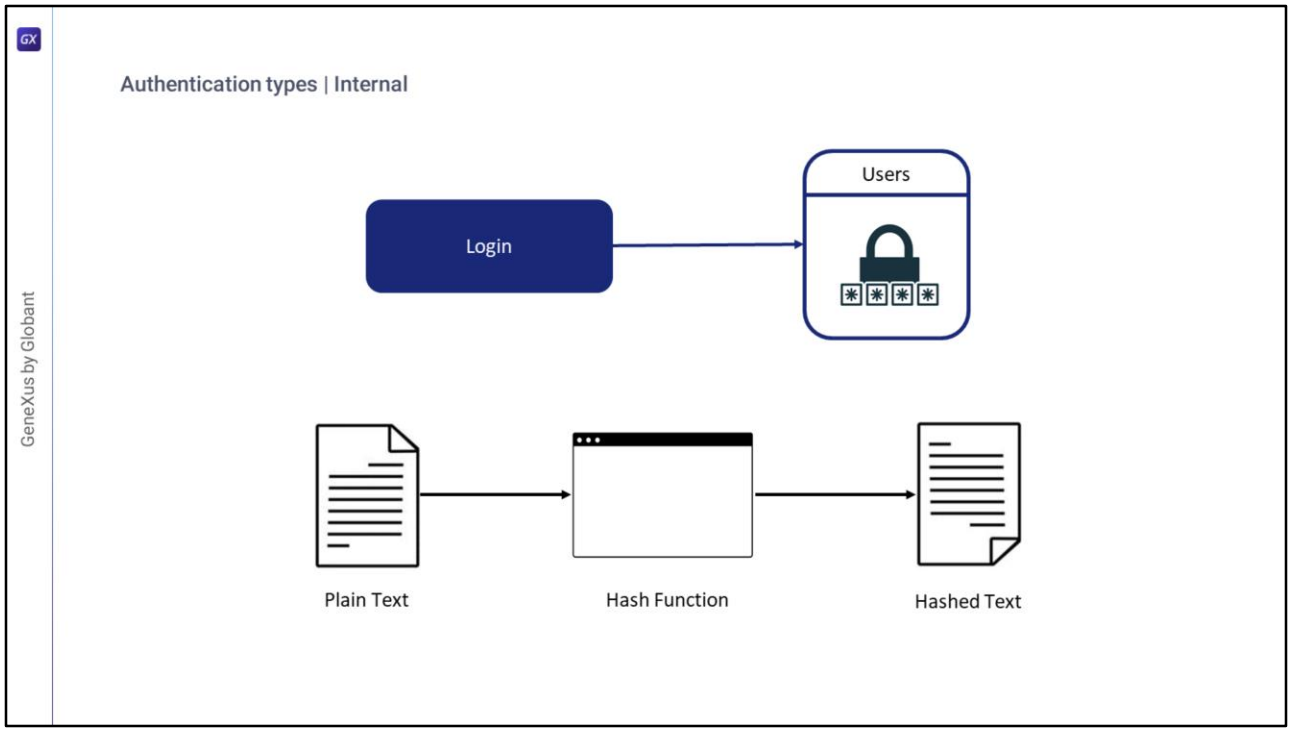
Authentication Types

Nicolas Adrién



As we said in previous videos, GAM offers different types of authentication, both internal (against the GAM database) and external (such as web services, social networks, or Google, also called Remote).

Let's go into detail on them.

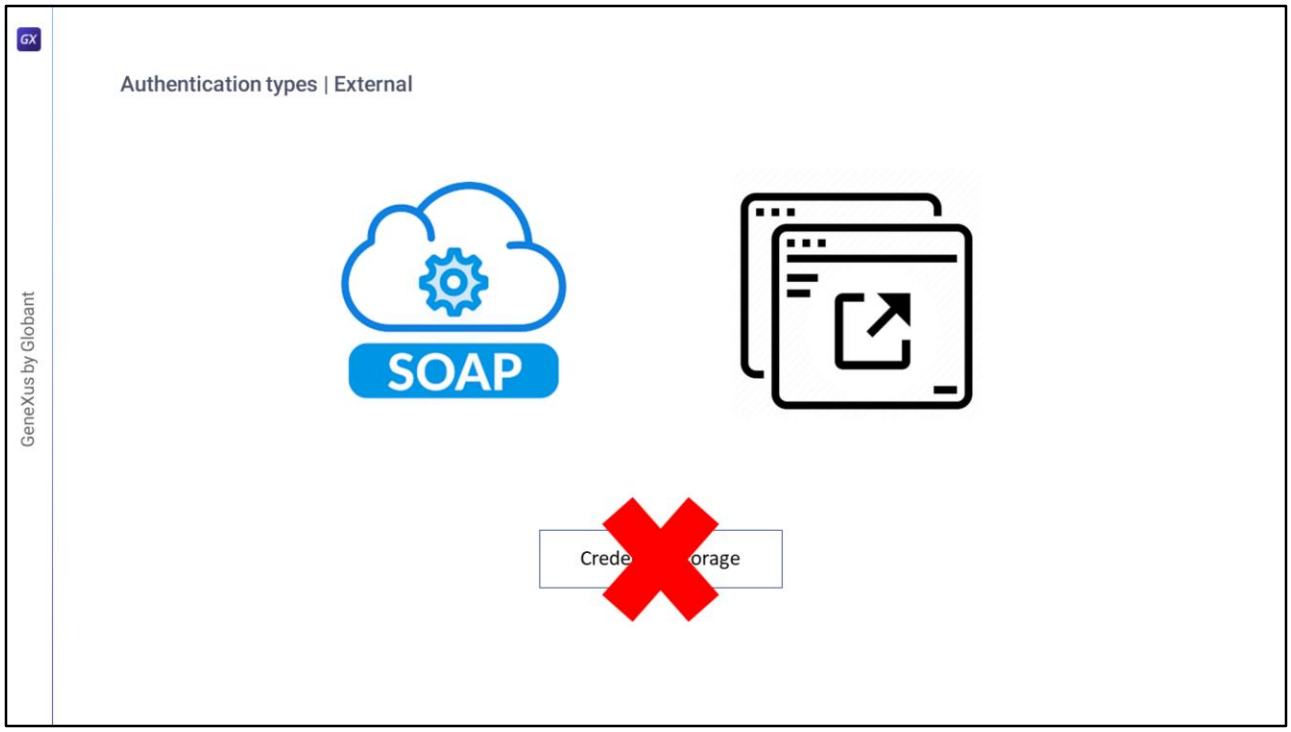


As for Internal Authentication Types, we have **Local Authentication** where user credentials are stored in the "Users" table of GAM.

GAM does not store the users' password but rather stores a hash of it. A hash is an algorithm that, given a plain text string, always results in the same string. Given this string, the original string cannot be obtained.

The hash is obtained from a unique key for each user and an algorithm named SHA-512, which will not be discussed in detail here.

This means that when retrieving GAM Users from the repository, the password property always has an empty value.



When integrating one application with another to exchange information, the first essential aspect is to solve the authentication issue.

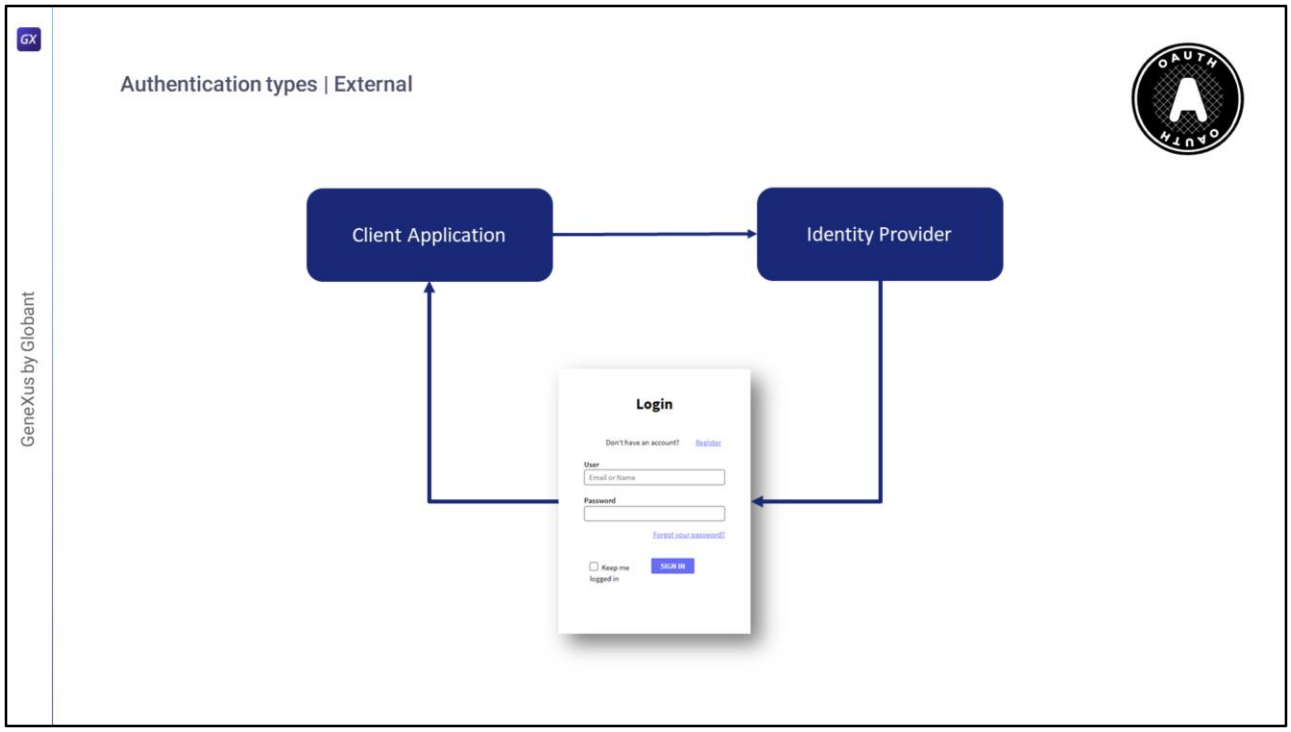
In the External authentication type, a first solution is to have the application we need to integrate expose a SOAP web service that performs the authentication.

Another scenario involves an external program that meets the authentication needs and is not necessarily a SOAP service.
The solution for this scenario is to configure the GAM custom authentication type in the GAM repository.

In both cases, GAM must be configured to accept the external program as the Identity Provider.

When using either of these authentication types, the GAM Client is not the owner of the user's credentials; only the username and other information that depends on the output of the external program will be stored in the Repository.

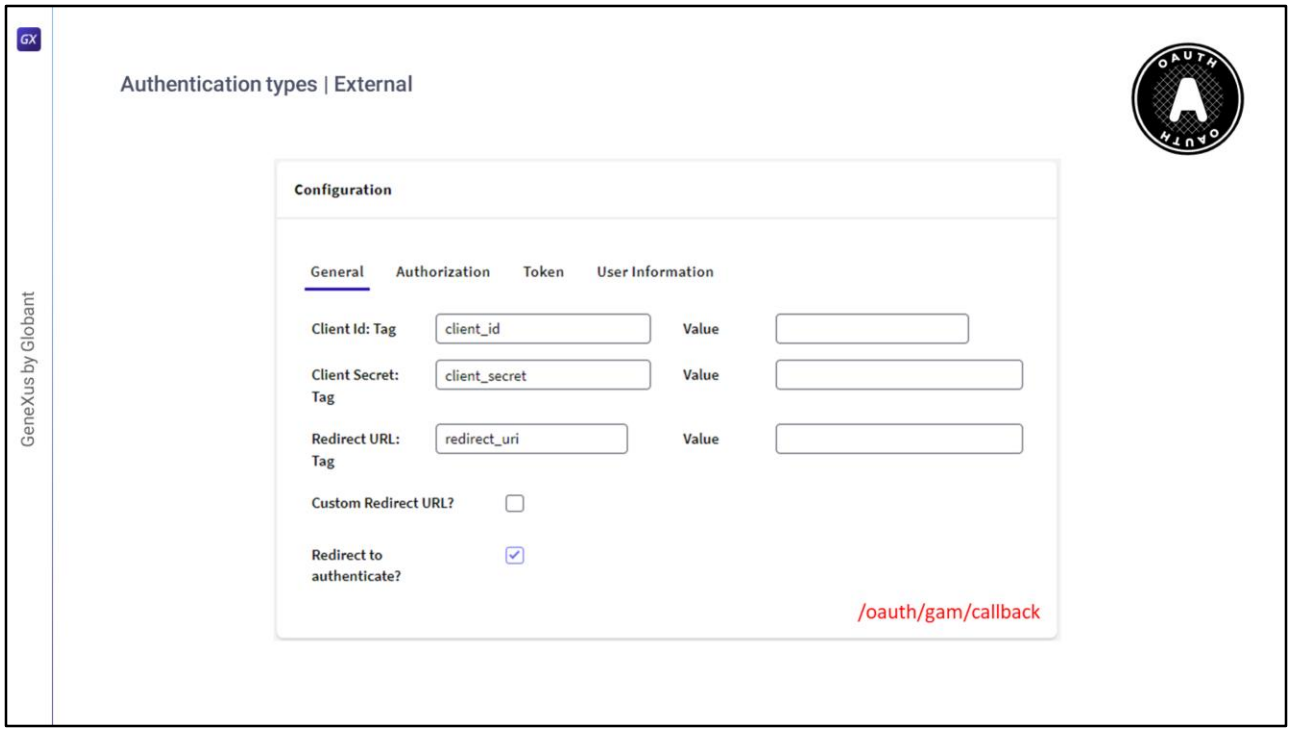
When authenticating to other external services, such as LDAP, an external program or web service can be used to act as a bridge between the GAM application and LDAP.



Among the other authentication types, first we find Oauth 2.0.
GAM allows authenticating with any OAuth 2.0 provider just by following a few simple steps.
When this type of authentication is selected, an application login is redirected to the configured identity provider.
The login is displayed by the provider; there, users enter their credentials and are redirected back to the application.

This Authentication type is defined in the same way as any of the other GAM types we have already mentioned, only that it requires a detailed configuration of the protocol used by the Provider. Therefore, to configure the OAuth authentication type in GAM, you must follow the documentation of the Identity Provider you wish to connect to.

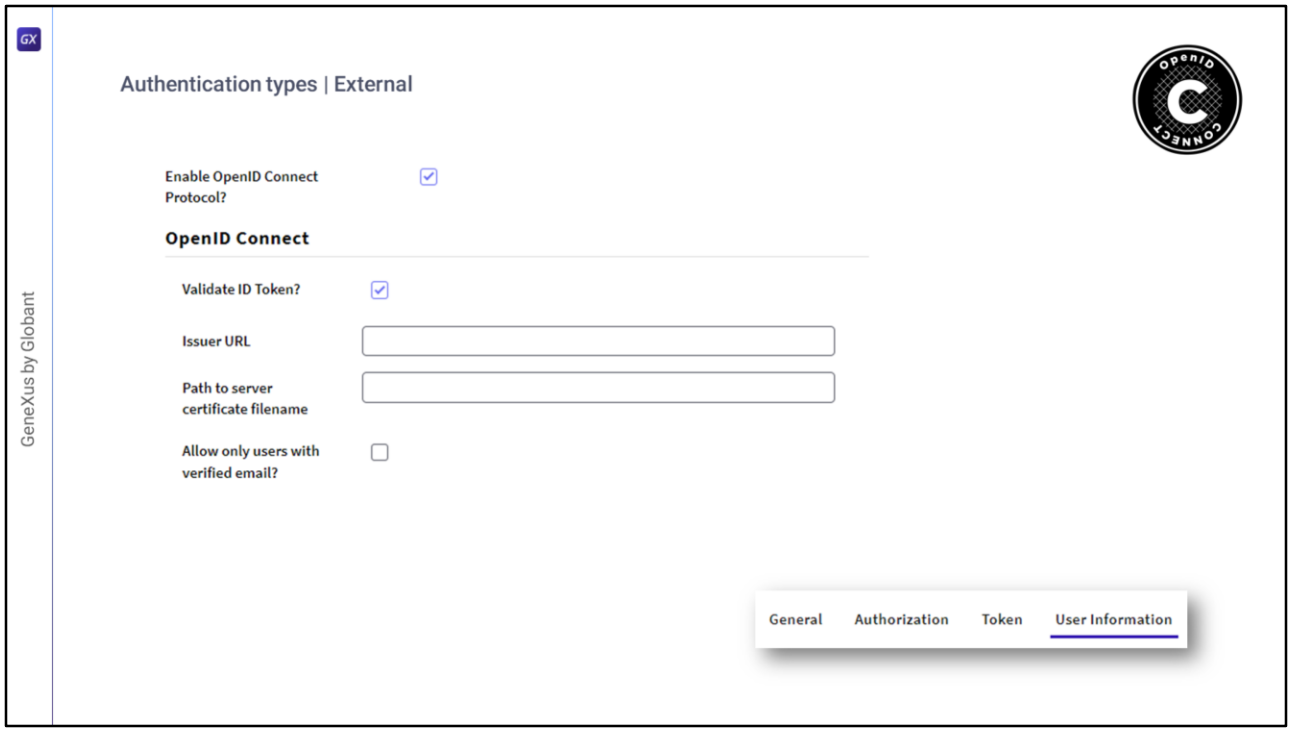
This protocol also provides SSO between different client applications.



Oauth 2.0 has a second authentication flow. With the option "Redirect to authenticate?" set to False, it provides OAuth 2.0 authentication using REST without redirecting to the identity provider, since GAM skips the redirection configured in the Authorization tab.

The other option (Custom Redirect URL?) indicates GAM that the specified return URL is customized. If set to False, it will then concatenate "/oauth/gam/callback". Otherwise, if it is set to True, the developer must implement this URL and read the IDP responses.

Both properties can be configured from the OAuth authentication type in the GAM back end.



Then we have OpenID Connect.

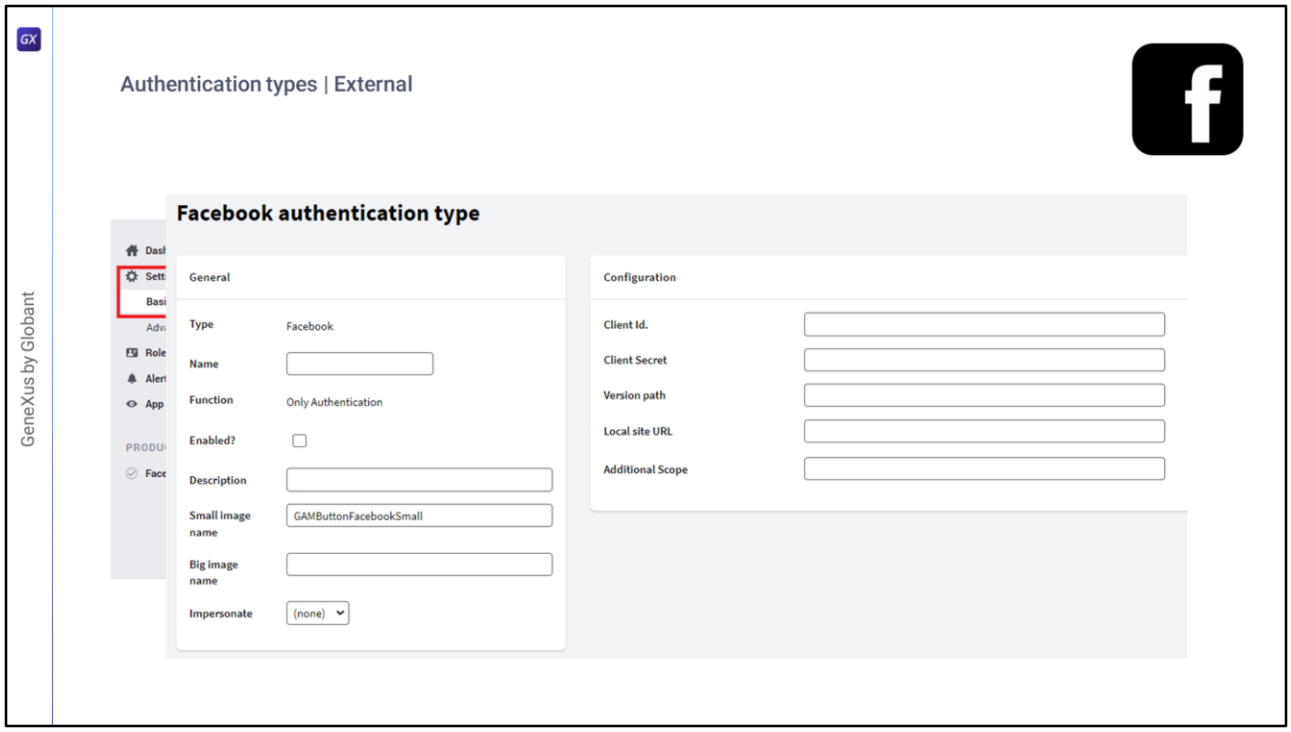
This is an authentication protocol that works with OAuth 2.0 by implementing authentication as an extension of the OAuth authorization process and is becoming one of the most common today.

The advantage it gives compared to OAuth is that this protocol allows us to obtain the user's information, unlike the OAuth standard that doesn't enable us to do so.

For this reason, it is no longer necessary to configure the User Information section in the Authentication Type.

For the protocol to work, you must activate the Validate ID Token property and include the provider and the local public certificate on a server.

With this information, a JSON Web Token signed and returned by the provider, called ID Token, is obtained.

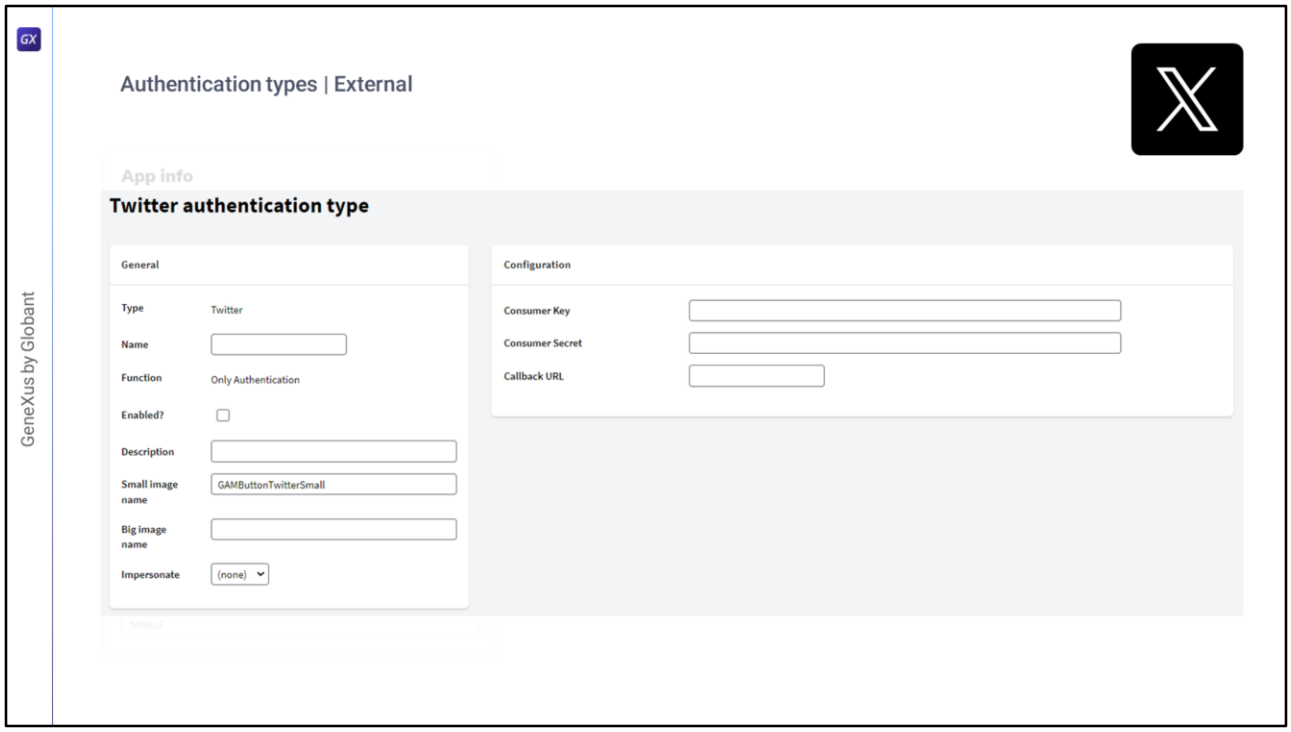


In second place, we have Facebook.

For this type, two steps must be followed:
First, you need to create a "Facebook client application" on your site and obtain an ID and key (called "Secret") for your application.
Second, define the "Facebook Authentication Type" in the GAM back end or API.

By following these steps thoroughly you will have the authentication type configured correctly.

This type can be used in web applications and native mobile applications, and in the background it is handled by Oauth 2.0.

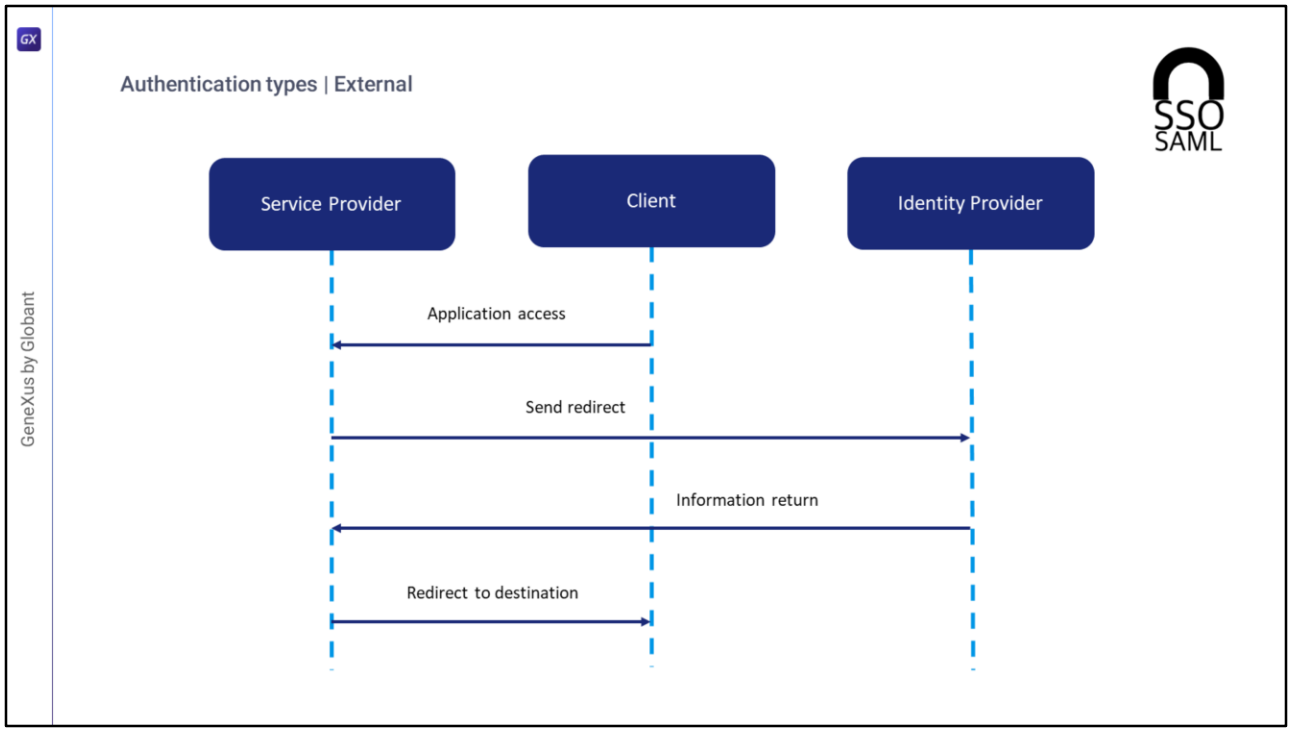


Then we have Twitter.

This case is executed in the same way as Facebook.
Step 1 is to create a Twitter application on your site, and obtain the consumer's key and secret for that application.
Step 2 is to define the Twitter authentication type using the GAM backend.

Once again, just like with Facebook, this type of authentication can be used in web apps and also in native mobile apps.

In the GeneXus Wiki, you can find detailed information about this and all the existing authentication types for GAM.

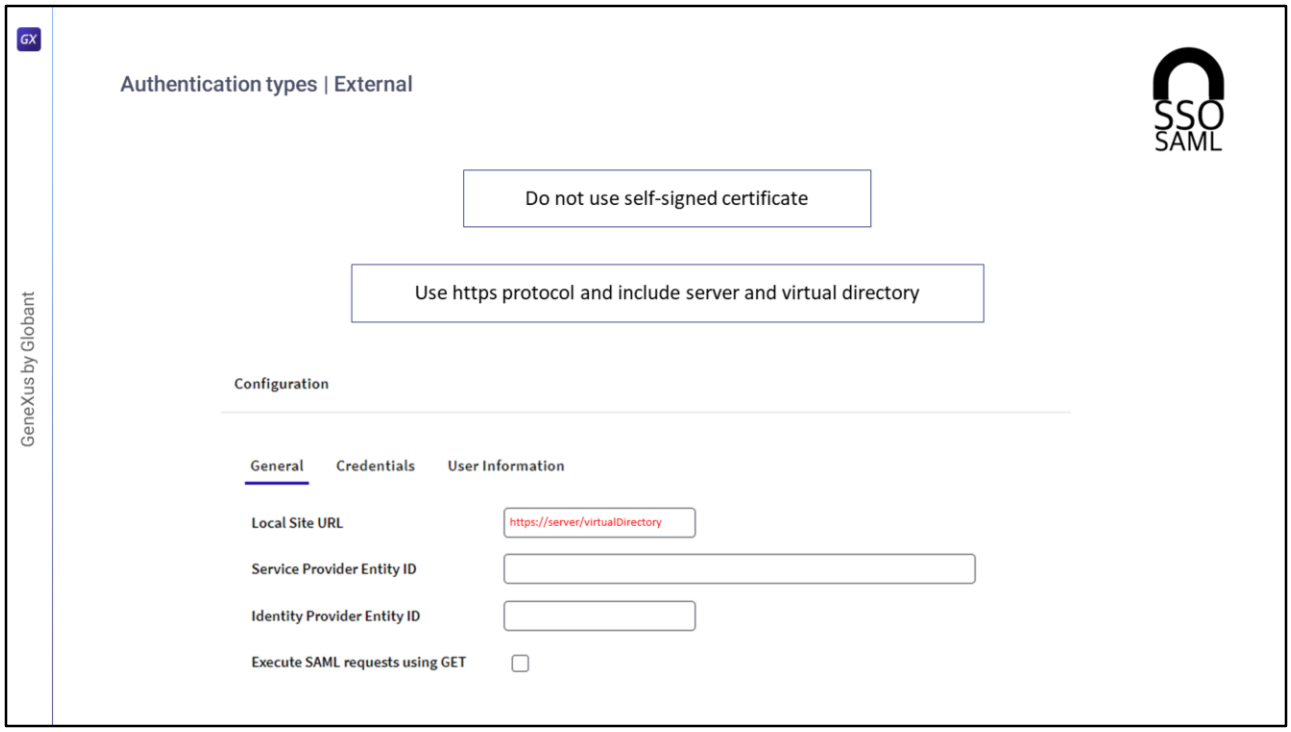


GAM provides authentication using any SAML 2.0 provider.

SAML is a secure XML-based communication mechanism for exchanging identities between organizations.

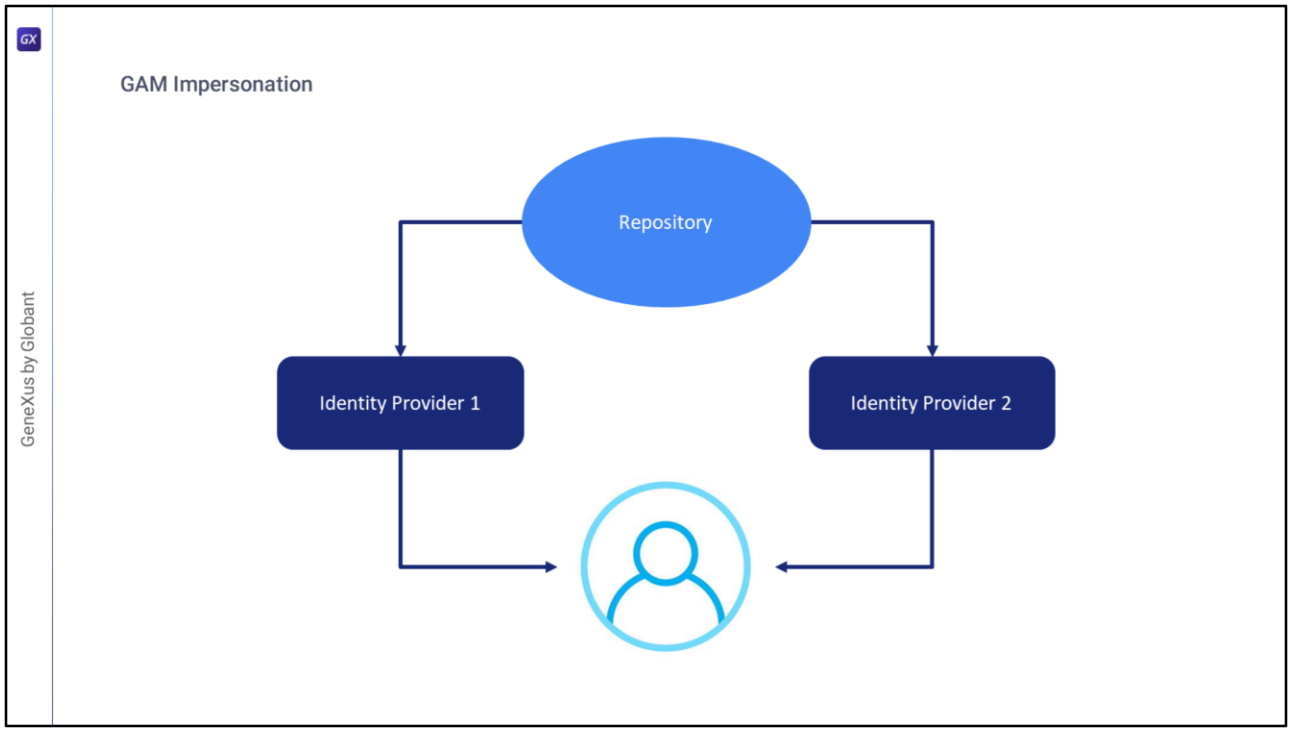
One of the use cases addressed by SAML is also SSO, so it avoids the need to maintain several credentials in multiple locations and increases security while reducing time-consuming administration tasks.

SAML involves two entities besides the client: a service provider and an identity provider.
A login flow is roughly as follows:
First, the user tries to access an application hosted by a service provider.
This provider generates an authentication request and redirects it to the user's browser.
Next, the identity provider receives the request, authenticates the user by requesting valid login credentials or checking for correct session cookies, and generates the response to be returned to the user's browser.
Lastly, the user is redirected to the target URL.



The following should be mentioned about SAML:

- Regarding certificates, it is recommended not to use self-signed certificates.
- The Local Site URL property must have HTTPS protocol and include a server and virtual directory as shown on the screen.



When the GAM Repository allows end users to authenticate with different identity providers, by default they are assigned to different GAM Users. For security reasons, users can authenticate using different mechanisms depending on the login source being used.
However, the login information must be assigned to the same logical GAM user.

Impersonation allows the repository to have two different authentication mechanisms that converge on the same user.
This is useful, for example, when it is not possible to use the same type of authentication from the intranet and from the internet, but the user should be the same.
It is also used for migrating from one type of authentication to another, where the "impersonated" authentication type is the one being migrated.

Depending on the type of authentication, there are different criteria for mapping users, which are described in the GeneXus Wiki.

To end this topic, we will move on to a series of demos in order to show the cases in practice in more detail.


GeneXus by Globant
GeneXus™
by Globant
training.genexus.com