Using other forms of authentication

What you should know about digital authentication

- What is Single Sign On (SSO)
- Integrating an application with a security provider
- Common sign-in protocols, authentication protocols, and token types

Interana authentication

- Password auth
- Google auth
- SAML auth
- Token auth
- Microsoft ADFS and Azure AD auth
- Okta auth
- OneLogin auth

What you should know about digital authentication

Digital authentication is used when accessing a non-public application. First you must authenticate, to show proof that you are who you say you are. Authentication is important for privacy and data security. The most common authentication method is with a username and password, also known as password auth. Password auth is the Interana default authentication method.

In some cases, an application allows you to authenticate once, then generates a token for subsequent interactions, using your stored token for authentication. If you forget your login credentials, such as a password, the application may prompt you to answer a previously established security question. A second method of authentication (two factor authentication, or 2FA) may also be required, such as entering a code sent to a mobile phone number.
What is Single Sign On (SSO)?

Single sign-on (SSO) is a session and user authentication service that allows users to have one set of login credentials (e.g., name and password) that access multiple applications. Many applications don't want to worry about security, which entails storing user credentials, dealing with flows for registration and password reset, and so on. Likewise, users don't want to have to remember separate passwords for every application. Finally, businesses want a single point of control that allows employees to access many systems at once.

SSO provides a solution to these problems. Centralized identity providers use a set of protocols that delegate authentication for many applications. A special type of secret handshake is used to ensure applications know the identity provider can be trusted to authenticate its users?

In a web SSO service, an agent module on the application server retrieves the authentication credentials for a user from the dedicated SSO policy server. The SSO service then authenticates the user against a user repository, such as a lightweight directory access protocol (LDAP) directory.

Integrating an application with an identity provider

To understand how to integrate an application with an identity provider, you need to know the sign-in protocol, the authentication protocol, and the token type.

Microsoft SME, David Gregory, used an excellent example in his ADFS deep dive blog post, comparing the authentication process to the process we go through when boarding an airplane:

**What is the Sign-in protocol?** I go to the one of those check-in kiosks, then print my boarding pass, then go through TSA line, then go to the boarding gate, and finally board the plane. It is expected that I interact with the airport in this fashion and performs these actions in sequence, otherwise, I will not get on my plane.

**What is the Authentication protocol?** While my boarding pass is part of this, I have to provide my ID or passport to prove who I am.

**What is the Token Type?** My boarding pass is my token to get on the plane.

This is a great analogy, because you use your ID or passport to authenticate as yourself before you can proceed to the security line at the airport. Once you're in the security line, your boarding pass becomes the token you need to actually board the plane. Unless you are flying internationally, your ID or passport won't be checked again.

There is another analogy with regards to a pre-existing relationship. Since we purchased a ticket (from the airline) before arriving at the airport, we had a pre-existing relationship with that airline (in the form of my purchased ticket). Similarly, an application you use may pre-register with an identity provider to acquire a private credential, so they know it's safe to be exchanging information with you.
Common sign-in protocols, authentication protocols, and token types

The most common sign-in protocols include WS-Fed, OAuth and SAML, with SAML also being a token type. Other sign-in protocols include OpenID, OpenID Connect, Google Authentication, and Facebook Connect.

The most common authentication protocols include the Password Authentication Protocol (PAP), and the Challenge-Handshake Authentication Protocol (CHAP).

- PAP initializes authentication when a user logs in with their username and password at the beginning of the connection.
- CHAP authentication is initialized by the server that sends a random string (usually 128B long) to the client. The client uses his password and the string received from the server as parameters for a MD5 hash function. The results are then sent together, along with his username in plain text.

The most common token types include SAML (Security Assertion Markup Language) and JWT (JSON Web Token). Google Auth uses a proprietary token which does not have a published standard.

Interana authentication

If you use an authentication method other than password auth, log in to the config node of the Interana cluster and disable password auth with this command: `ia settings update auth password_auth disabled`

Interana supports the following methods of authentication:

<table>
<thead>
<tr>
<th>Authentication method</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Password auth</strong></td>
<td>Interana fully manages password authentication. A web form appears that lets you register a username and password. The hashed password is stored locally on the cluster in the config DB. Once a user is authenticated, Interana stores a browser cookie for the user so they don't have to authenticate again.</td>
</tr>
<tr>
<td><strong>Google auth</strong></td>
<td>Interana delegates management to Google for authentication, which uses the OAuth2 sign-in protocol and token type. Google simply provides the client libraries with which Interana interacts. Once a user has authenticated, Interana stores a browser cookie for the user so they don't need to authenticate again.</td>
</tr>
<tr>
<td><strong>SAML auth</strong></td>
<td>Interana lets you register with any identity provider that complies with the SAML sign-in protocol. This is opaque to Interana, and uses a SAML token. Once a user has authenticated, Interana stores a browser cookie for the user so they don't need to authenticate again.</td>
</tr>
<tr>
<td><strong>Token auth</strong></td>
<td>Interana lets a user (who has already authenticated via one of the other protocols listed) generate a private token that can be used repeatedly to invoke the Interana query API. This token is stored in Interana's config DB.</td>
</tr>
<tr>
<td><strong>Microsoft ADFS and Azure AD auth</strong></td>
<td>Active Directory Federation Services (ADFS) is a software component developed by Microsoft that can be installed on Windows Server operating systems to provide users with single sign-on access to systems and applications across organizational boundaries. Azure Active Directory (Azure AD) is Microsoft's multi-tenant, cloud based directory and identity management service.</td>
</tr>
<tr>
<td>Authentication method</td>
<td>Notes</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-------</td>
</tr>
<tr>
<td><strong>Okta auth</strong></td>
<td>Okta provides reliable integration for SSO to web and mobile apps, with a full-featured federation engine and flexible access policy.</td>
</tr>
<tr>
<td><strong>OneLogin auth</strong></td>
<td>OneLogin is a cloud identity management platform that provides secure single sign-on, identity and access management (IAM) for an efficient and secure enterprise environment.</td>
</tr>
</tbody>
</table>

## Password auth

Password auth is the Interana default authentication method. When a user navigates to Interana in a browser, Interana first checks to see if the user is already logged in. Password auth requires email confirmation, and can be restricted to certain whitelisted email domains.

For password auth, Interana stores a cookie in the browser once the user has successfully logged in. If the user is not recognized, a login screen displays.

## Google auth

When a user navigates to Interana in a browser, Interana checks to see if the user is already logged in. For Google auth, Interana stores a cookie in the browser once a user has successfully logged in. This section explains how Google auth is implemented with Interana. For more information, Google documents is OAuth2 implementation at [https://developers.google.com/identity/protocols/OAuth2](https://developers.google.com/identity/protocols/OAuth2).

### Google auth implementation process

**Step 1: Google auth registration**  You must register with Google as a valid application to use Google auth with Interana. Google then gives you credentials that must be installed on the API node of the Interana cluster.

**Step 2: Interana configuration**  The Interana admin uses the CLI to configure the settings in Interana's config DB.

**Step 3: User attempts to access Interana**  When a user navigates to Interana for the first time and Google auth is enabled (and is not logged in), Interana displays a **Sign in with Google** button that (when clicked) redirects to the /api/
google endpoint on the API node of the Interana cluster.

**Step 4: Interana asks Google to authenticate** When Interana receives the /api/google request, it is redirected to Google to ask for a token. Google responds with a code, then Interana uses that code to get an actual token. Finally, Interana makes a direct Google API call to get the user information.

**Step 5: Find or create the Interana user** If Interana gets a login request from a previously unknown user, Interana creates a new user.

**Step 6: Save the user info in a cookie** Once user login credentials are verified and the user's presence in the config DB is confirmed, Interana sets a browser cookie that contains the user_id, customer_id, and user_name. The user browser cookie enables easy access, so the user doesn't need to sign in every time.

---

**SAML auth**

When working with a SAML Auth provider, the first thing is to make a connection and load the SAML configuration. The configuration can be stored locally, on the Interana cluster, or loaded remotely from a URL. For Managed Edition customers, Interana typically stores the remote URL for the SAML provider on the config node of the Interana cluster.

Once the desired IDP URL from the SAML config is loaded, Interana redirects authentication requests directly to the SAML provider. If Interana gets a login request from an unknown user, a new user is created.

---

**Token auth**

Interana supports API tokens that can be passed as part of the Authorization header. These tokens can be generated by admin users and are long-lasting.

---

**Microsoft ADFS and Azure AD auth**

You can use Microsoft ADFS or Azure AD as an authentication provider instead of the standard Interana password authentication. For more information, see the following articles:

- Add ADFS authentication to your Interana instance
- Add Azure AD authentication to your Interana instance

---

**Okta auth**

You can use Okta as an authentication provider instead of the standard Interana password authentication. For more information, see the following article:

- Add Okta authentication to your Interana instance
OneLogin auth

You can use OneLogin as an authentication provider instead of the standard Interana password authentication. For more information, see the following article:

- Add OneLogin authentication to your Interana instance