Identity Federation

Course 2023-2024 Q1

19/12/2023

Group 3:

Natalia Dai

Clàudia Giró Figueras

Mario Martín Sola

Javier Villarreal Arias

TABLE OF CONTENTS

1. Introduction

In an era where our online presence spans diverse platforms, from social media networks to various online services, each platform requires its own set of credentials and login information, resulting in a scattered web of identities. Managing user identities across these digital domains poses significant challenges.
Identity Federation addresses this issue by offering a unified approach to managing and authenticating user identities across different systems while enhancing security.

This project is dedicated to provide a comprehensive exploration of Identity Federation, navigating through mainly its concepts and components, mechanisms and protocols, recognizing how Identity Federation works.

2. Key concepts

2.1. Profiling

Identity refers to the information associated with or characterizes a physical or legal person, such as name, gender, work position, place of residence, and more.

On the other hand, Electronic Identity (eID) is the virtual manifestation of this real-world identity in the digital space. It encapsulates the ownership of information and dictates access rights to data and applications, serving as the bridge between the digital representation and the real identities.

Physical people may have multiple eIDs, creating different identities and profiles depending on the attributes that person wants to be associated with.

2.2. Single Sign-On

Single Sign-On (SSO) is a technology that simplifies the user experience by consolidating multiple application login screens into one, enabling a given user to access various systems and applications with a single identification instance or account. Simplifying, with SSO a user only has to enter his/her login credentials just once on a unified page in order to gain access to all their Software as a Service applications (SaaS).

One of SSO's primary advantages is the elimination of the need to repeat over and over authentications each time a user wants to access a service in case he/she has been disconnected from this given service. This convenience is particularly helpful when multiple systems share the same password: the users can authenticate once and maintain a valid session for all other applications using SSO, streamlining their access.

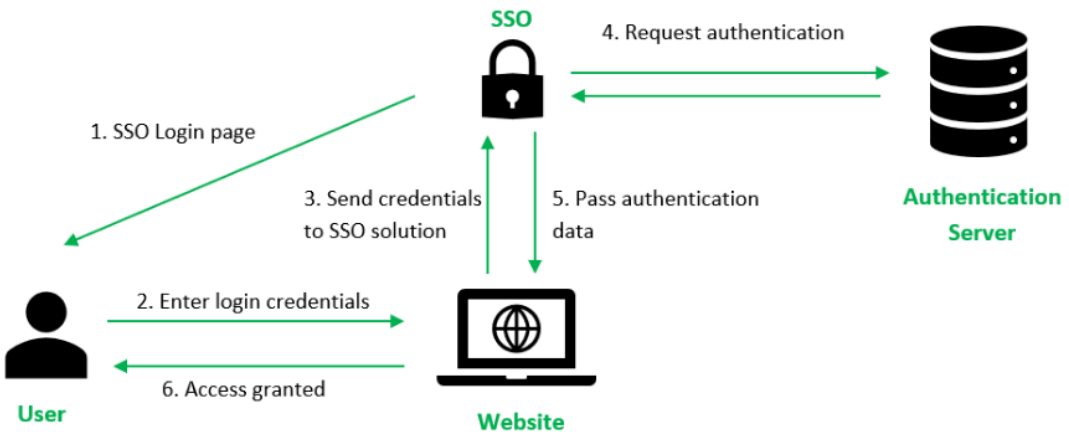
In addition, SSO plays a vital role in identity and access management (IAM) or access control solutions since it ensures the verification of a user's identity, and gives permissions accordingly, increasing security.

However, if SSO is compromised, all services linked to it may also be at risk. Or in case it becomes inaccessible, no user can access those applications.

How does it work?

The login steps usually are the following ones:

1. The user browses to the application or website they want to gain access to a SP, known as a Service Provider. He/She enters the login credentials and the website checks if this user has already been authenticated by SSO. In case it is affirmative, the SSO indeed gives the user access. Else, we shall follow the next steps.
2. The user enters username and password on the SSO solution (case 2 from above).
3. The login credentials presented are sent from the SP to the SSO system, known as the Identity Provider, as part of a request to authenticate the user.
4. In this step the Identity Provider verifies the user identity. Once it is verified, a verification token will be sent back to the Service Provider confirming a successful authentication.
5. This token is passed to the website where the user will be granted access to.
6. In this final step, the access is granted for the user to navigate in this website or application.



3. Identity Federation

3.1. What is an Identity Federation

Most of the nowadays organizations maintain user accounts and passwords for its own users and also manage the permissions of each user using different methods such as roles, attributes, etc. Sometimes these organizations are interested in collaborating with each other, requiring as a consequence that users from other organizations can access your organization. A simple approach to solve this issue is to just create a new account for that user from other organization, but this solution leads to considerable problems:

1. Now, users have to manage an account for each organization they need to access.
2. Organizations need a way to validate the identity of users that are not directly related to them. This means that every organization should have enough information about the rest of the organization's users for validation.

Identity federation technologies appear as an existing solution to address identity management in information systems. It refers to a relationship between two systems based on trust, in which one system uses authentication information in order to grant access to the other system without asking for authorization multiple times.

When it comes to defining what are the **key concepts** of how a good identity federation system should work, we encounter what Kim Cameron defined as “the seven laws of identity”. These laws are:

1. **User control and consent:** *“Technical identity systems must only reveal information identifying a user with the user’s consent”.*

   Users will give their permission to share data to a system that will reveal their identity attributes to others without explicit consent of the user. Therefore, a user needs to have confidence on any system that is provided with his identity information.

2. **Minimal disclosure:** "*The solution which discloses the least amount of identifying information and best limits its use is the most stable long term solution.*"

   All systems are vulnerable to attack and to get their confidential information stolen. Good practices are to store the smallest amount of identifying information that is needed, and ensure that it's stored securely and deleted quickly.

3. **Justification:** "*Digital identity systems must be designed so the disclosure of identifying information is limited to parties having a necessary and justifiable place in a given identity relationship.*"

   There is no use in giving information to parties that do not play any role in the communication that is being made. Only those who can prove they need access can get it.

4. **Direct identity:** *"A universal identity system must support both "omni-directional" identifiers for use by public entities and "unidirectional" identifiers for use by private entities, thus facilitating discovery while preventing unnecessary release of correlation random identifiers."*

   Protection of identity is paramount, and users should be assigned private identifiers for that purpose. Companies can't work together to build a more permanent view of someone working across platforms.

5. **Competition:** *"A universal identity system must channel and enable the inter-working of multiple identity technologies run by multiple identity providers."*

   Many identity providers should be supported, as competition breeds better performance. This necessitates that there is an overarching meta-identity system that uses a common protocol for the transport of identity credentials, whilst supporting an infinite variety in the types of credential technologies that are supported.

6. **Human integration:** *"The universal identity metasystem must define the human user to be a component of the distributed system integrated through unambiguous human-machine communication mechanisms offering protection against identity attacks."*

   A real person has a place in the process, reducing the risk of computer-to-computer hacks.

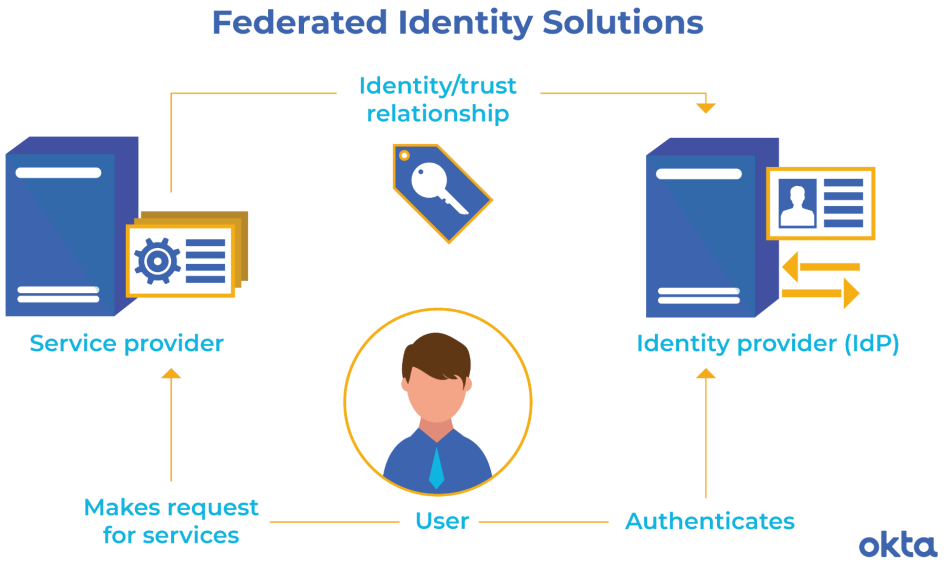
7. **Consistency: "***The unifying identity metasystem must guarantee its users a simple, consistent experience while enabling separation of contexts through multiple operators and technologies".*

   The users should have a simple, consistent experience among platforms regardless of the underlying credential technologies that are in use.

3.2. How does it work

3.2.1 Workflow

The idea goes as follows: first, a user tries to access a resource or service provided by a Service Provider (SP). Then, the SP asks for authorization, for which the user will have to introduce the credentials corresponding to the identity registered on the Identity Provider (IdP). The SP sends an authentication request to the IdP in order to verify the identity of the user that is trying to access their services. In case of correct authentication, the IdP will use a specific protocol to send the required information about the user (the different types of protocols will be explained later). Finally, the SP validates the information retrieved from the IdP and decides what action it should take.



3.2.2 Trust Frameworks

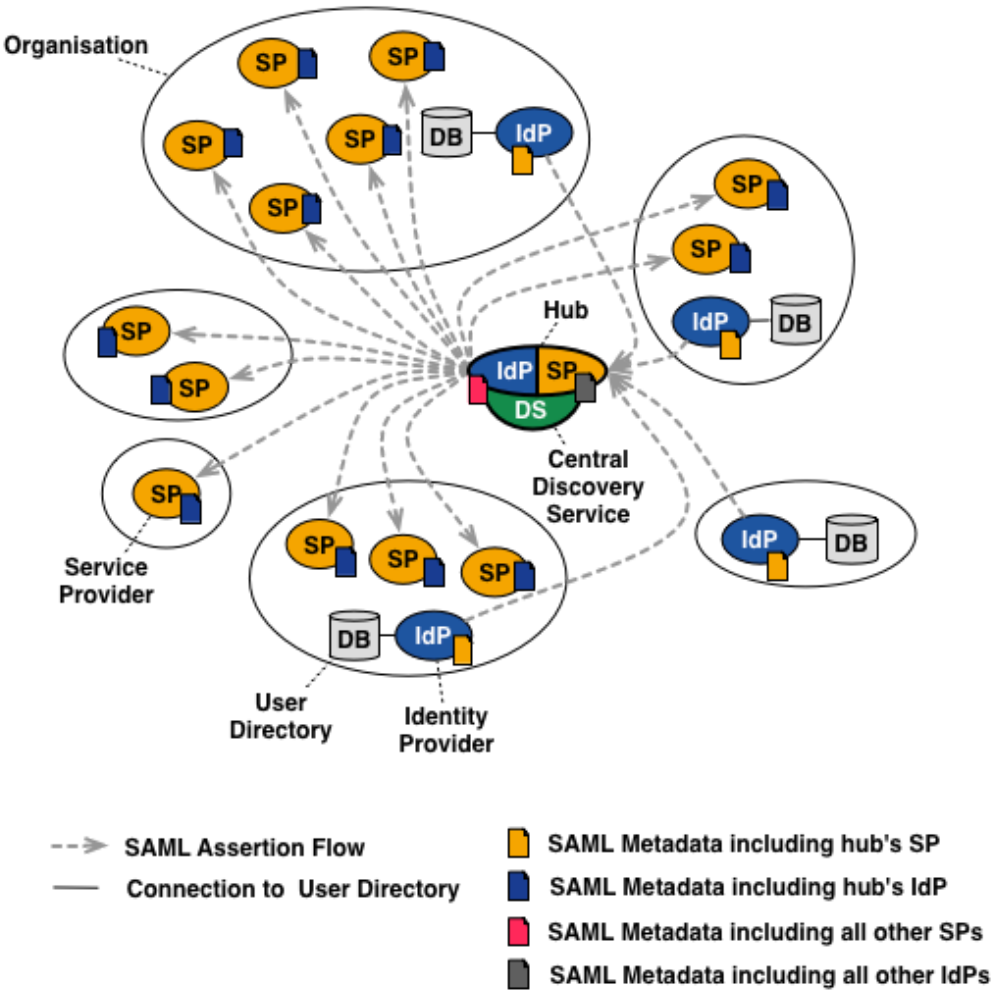
SPs and IDPs must establish a trust relationship between each other before participating in a federation. These relationships include details of the federation protocols, the exchange of cryptographic keys, the configuration of service endpoint locations, lines of communication, etc. They even need to define responsibilities of each party, administrative and legal agreements. All of this information about the relationship established between SPs and IDPs may be formalized in what is called a **trust framework.**

3.2.3 Architectures

There are different types of Identity Federations mechanisms. Those are:

FULL MESH

This type is the most common and straightforward to implement. Every component is distributed, which means that there will not be any central component that needs extra protection in case it fails. Each organization operates its own IdP that is connected to a local user database and an arbitrary number of Service Providers. There exists a metadata file that describes who takes part in the federation, distributed with SAML. The main problem of this distribution is to efficiently manage the metadata file and handle the attribute release at the IdPs. To solve this some federations use a web-based service to register all entities. Others, for example, use a set of scripts to compose the SAML2 metadata for the federation.



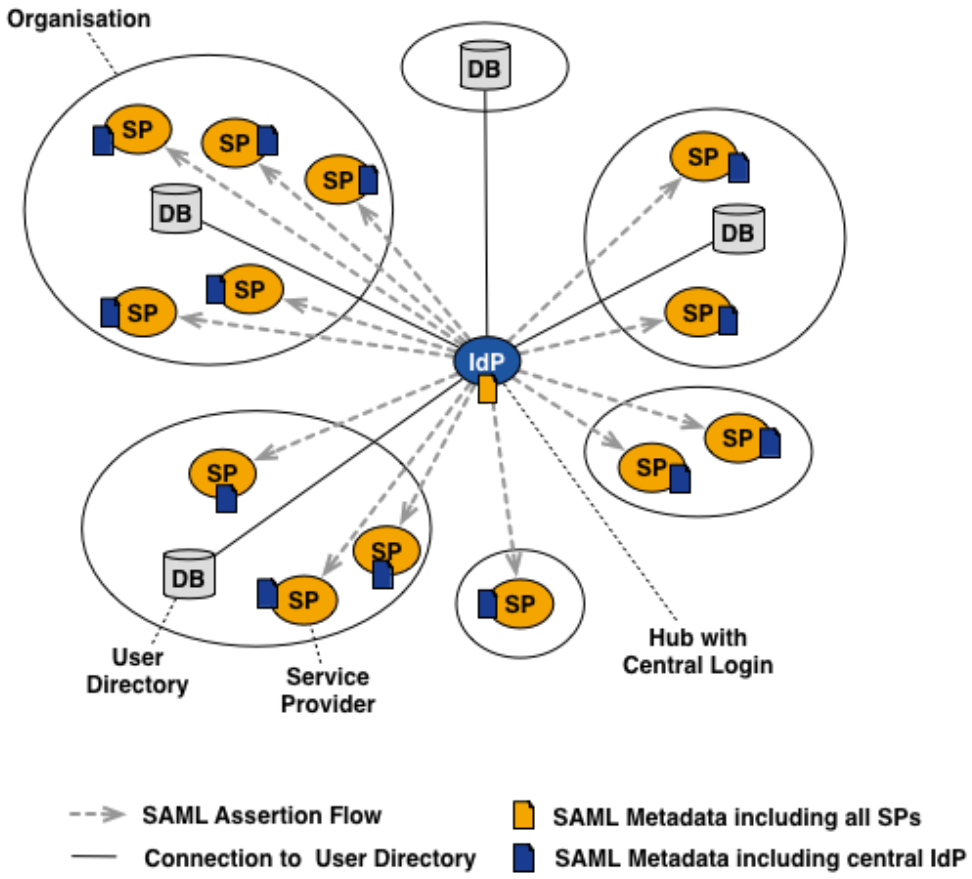
HUB-AND-SPOKE WITH DISTRIBUTED LOGIN

This type of federations rely on a central hub or proxy via which all SAML assertions are sent. The hub acts as a Service Provider for the Identity Providers in the federation, and acts as an Identity Provider for the Service Providers. Each organization still has their own Identity Provider connected to a local user database, but it typically only needs metadata from the hub. The Service Providers only need metadata from the hub.

The central hub has to be carefully secured and protected, as it is a single-point of failure. The hub only knows which entities are in the federation, it never learns the user's credentials.

The hub can be used to "connect" individual Service Providers and Identity Providers. This can be useful for example by using a web interface. It can also control, extend or transform

the attributes that are sent from the IdF to the SP. Even this, interfederation scenarios are non trivial to handle.

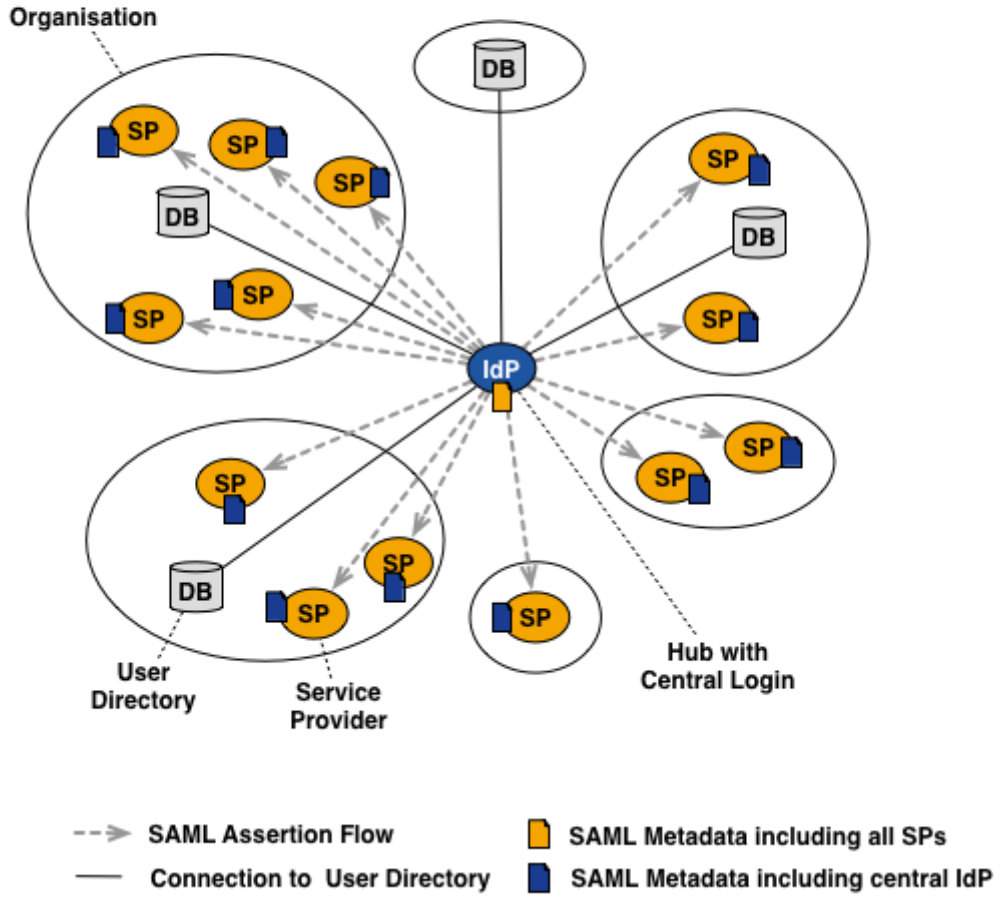


HUB-AND-SPOKE WITH CENTRALIZED LOGIN

In this type of federation only one single Identity Provider is present. All user databases are connected to a central IdP, and users enter their organization credential on this IdP. This means the IdP needs to be especially trusted by all the organizations that work with it.

It has the same problem as the previous federation explained: the IdP is a single point of failure, and it must have high availability to ensure the service is always provided. Depending on the number of logins, scalability issues may arise.

With this mechanism it is very easy to support new authentication protocols on the hub thanks to the central login.


Organisation
DB
SP
SP
SP
DB
SP
SP
SP
DB
SP
IdP
SP
SP
SP
DB
SP
SP
SP
User Directory
Service Provider
Hub with Central Login
SAML Assertion Flow
Connection to User Directory
SAML Metadata including all SPs
SAML Metadata including central IdP

3.3 Benefits of Identity Federation

This solution benefits both users and organizations in multiple ways:

- **Users:**
  - Their user experience is enhanced since a single credential is enough to access multiple accounts on different organizations.
  - Also, when authenticating to an organization, a session will be created with a defined lifetime. The access to other organizations that do pertain to the same Identity Federation will be automatic, which means that there will be no need to authenticate again. This mechanism is the definition of the **Single Sign-On**.
- **Organizations:**
  - Now the organization does not need to prove the identity of the user, nor manage their credentials, which reduces costs of time and management.
  - Flexibility is added since, if new authentication technologies come to the world, only IdPs need to update their system, not SPs.
  - SPs do not need to focus on authentication systems or credential management.

3.4 Examples

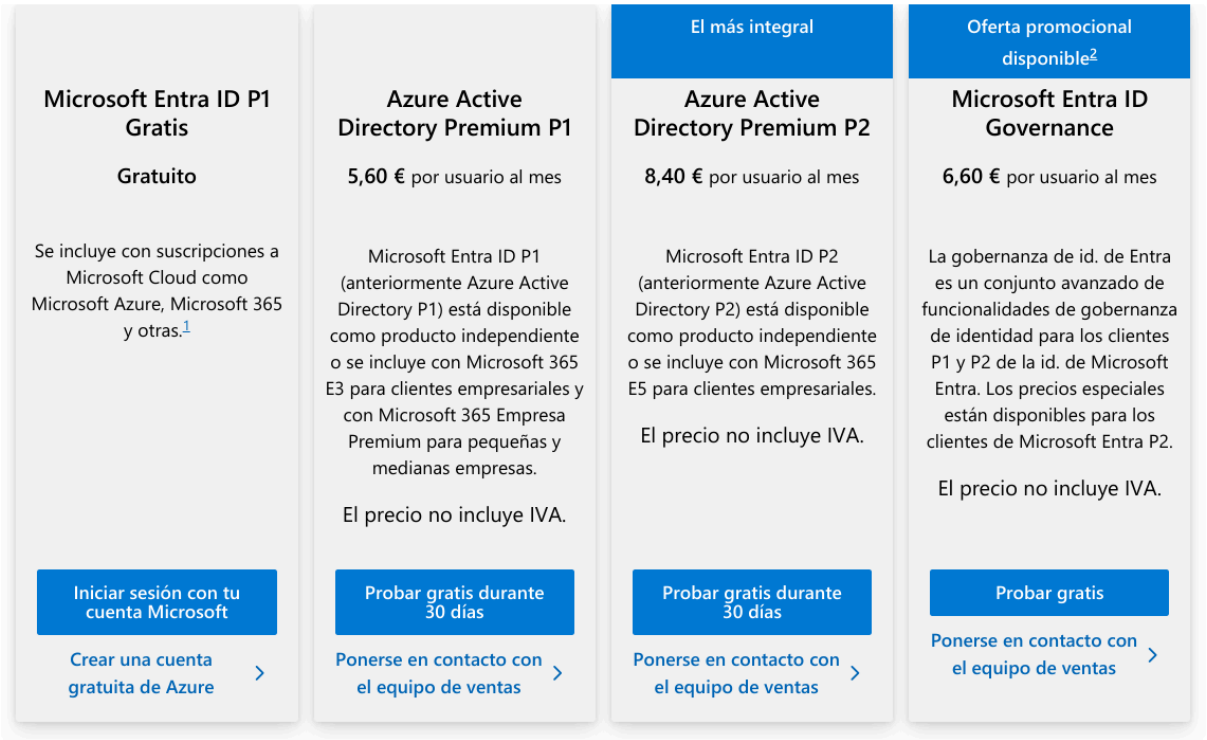
3.4.1 Microsoft Entra ID

It was known as Azure AD, but the name was changed to Microsoft Entra ID. It is a cloud-based service that offers authentication and authorization to various Microsoft services such as Microsoft 365, Dynamics 365 and Microsoft Azure. It uses single sign-on, regardless of whether their applications are cloud-based or on-premises.

It allows end users to manage all their identities and access to all their applications in a single central location, whether they are in the cloud or local, to improve visibility and control.

Entra ID offers different authentication methods: password-based, multi-factor, smart card, etc. It also includes several security features, such as Conditional Access policies, risk-based authentication, and identity protection. It protects and verifies every identity, provides only the needed privileges and simplifies the experience for the end user.

Its prices vary depending on the service the company needs. It has offers for small, medium and large businesses. The price shown is for users and billed monthly. The picture below shows the available plans and its prices:



3.4.2 ADFS

Active Directory Federation Services (ADFS) is a solution to provide single sign-on and identity federation capabilities. It is a feature of Microsoft Windows server and it allows users to access multiple applications or systems with a single set of credentials. It establishes a trust relationship and allows users to authenticate themselves to obtain access to a resource in another organization. The ADFS server issues a security token so the user can present it to the target organization to access its resources. Once trust is verified, the ADFS server in the target organization extracts the claims from the security token. These claims are used to make authorization decisions, determining whether the user has the required permissions to access the requested resource.

ADFS supports protocols such as Security Assertion Markup Language (SAML) and WS-Federation.

3.4.3 Google

Google works as an IdF with their platform Google Identity Platform. It allows users from a third-party application to authenticate themselves using their Google Accounts. They can use an already existing account or they can create a new one with just one click.

It works with Google Sign-In, which allows users to log in using their Google credentials. When they do so they may be asked to grant permissions to the applications, ensuring that they are giving consent to all the data they are sharing. Once authenticated the application can ask for extra information to complete their profile.

It uses the protocol OAuth 2.0, explained in another section in this document. After the user grants access the application receives an access token. This token is used to make API requests on behalf of the user, and it has a limited scope based on the permissions granted during the authorization process.

4. Protocols used in IdF and roles

Identity federation relies on protocols that streamline the authentication and authorization process, enabling users to access resources from multiple applications or services without having to create separate accounts or repeatedly enter their credentials.

These protocols define how messages sent between the IdPs and SPs should be structured, composed, protected and processed.

Three prominent protocols widely used in identity federation are: *OpenID*, *SAML* and *OAuth.*

4.1. OpenID

4.1.1 Concept

OpenID is an open standard that simplifies authentication and authorization by allowing users to log in to applications using their existing credentials from an identity provider "*idP*". This simplifies the authentication process and reduces the burden on users to create and manage multiple accounts. It also offers some other advantages, including:

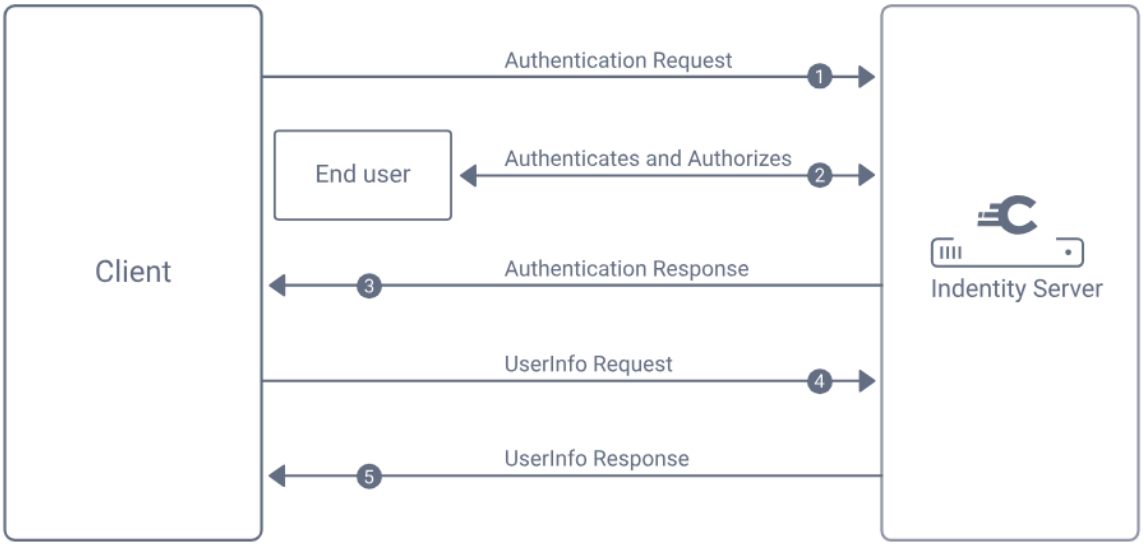
- **Uninterrupted User Experience**: Users can access multiple applications using their preferred IdP without having to repeat several times the processes of account creation and password management.
- **Reduced Security Risks:** This protocol mitigates the risk of credential exposure by allowing users to authenticate with trusted IdPs, rather than directly with each application.
- **Improved User Control:** Users retain control over their identity and authorization by granting access permissions to specific applications.

There have been many versions of the protocol since its creation. The first two versions, 1.0 and 2.0, are decentralized authentication protocols, both focused on user authentication.
Despite that, a more recent version replaced these two: *OpenID Connect.*

4.1.2 OpenID Connect

This protocol defines two federation participant terms, the **OpenId provider (OP)** as the IdP and **OpenID client** as the relying party.
Despite that, the most recent version, called OpenID Connect (OIDC), is an extension of OAuth 2.0 that also provides authentication and user information services introducing JSON Web Tokens, to exchange information in a secure way. It also offers an additional authentication layer, making it a more complete and versatile version.

In general, OpenID Connect works as shown in the following scheme:



First, the client initiates the process by sending a request to the identity server, seeking authorization and user authentication. Following this, the identity server authenticates the end-user and communicates the authentication (and authorization) result back to the client. Optionally, the client can then access user details via the UserInfo endpoint, with the identity server subsequently furnishing these details to the client.

4.2. SAML

SAML (Security Assertion Markup Language) is an XML-based protocol designed for exchanging authentication and authorization information between service providers (SPs) and IdPs. It facilitates secure communication between these entities, allowing users to access resources from SPs without having to re-authenticate every time.

This protocol is particularly suitable for enterprise environments with complex authentication requirements. SAML offers distinct benefits, including:

- **Centralized Authentication**: SAML allows IdPs to authenticate users and issue SAML assertions, which are cryptographically signed statements containing user information. SPs can verify these assertions to grant access to resources.

- **Efficient Authorization**: SAML enables SPs to request specific permissions from users instead of requiring them to provide all their credentials. This reduces the disclosure of sensitive data and enhances security.

- **Cross-Platform Interoperability**: SAML is widely adopted and interoperable across various platforms, making it a suitable choice for enterprise environments.

4.2.1 How does it work?

SAML works by passing information about the users, logins and attributes between the IdP and SPs. Users log in once to the Single Sign On with the identity provider and then, it passes SAML attributes to the service provider when the user tries to access those services. Then, the service provider requests the authorization and authentication from the IdP.

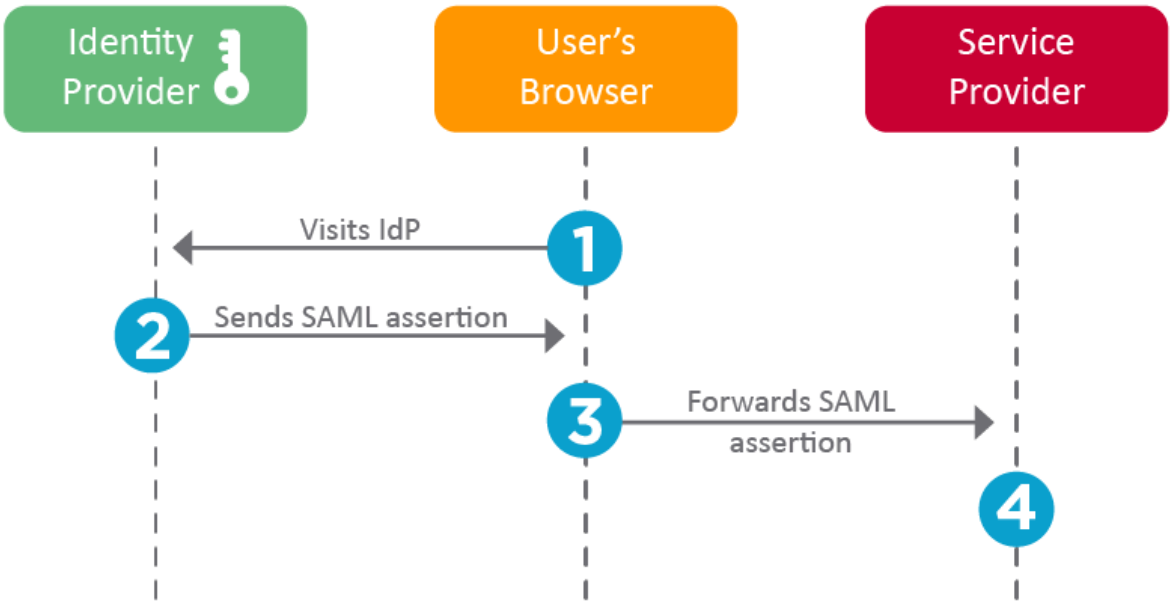
As both systems use SAML and should have the same configuration, users only have to log in one time.

Once integrated, the IdP can begin authenticating users on behalf of the SP. These types of messages are sent between the SP and the IdP. There are different types of messages that are sent between the SP and the IdP as part of the SAML process:

- **AuthnRequest (SP -> IdP)**: User authentication request embedded within a redirect.
- **SAML Assertion (IdP -> SP)**: Signed XML document that contains the authentication details for the end user.

Both the SP or IdP can initiate the process:

- **IdP initiated (SAML 1.1):** The IdP sends a SAML assertion to the SP when the user accesses the SP site. No AuthnRequest required.

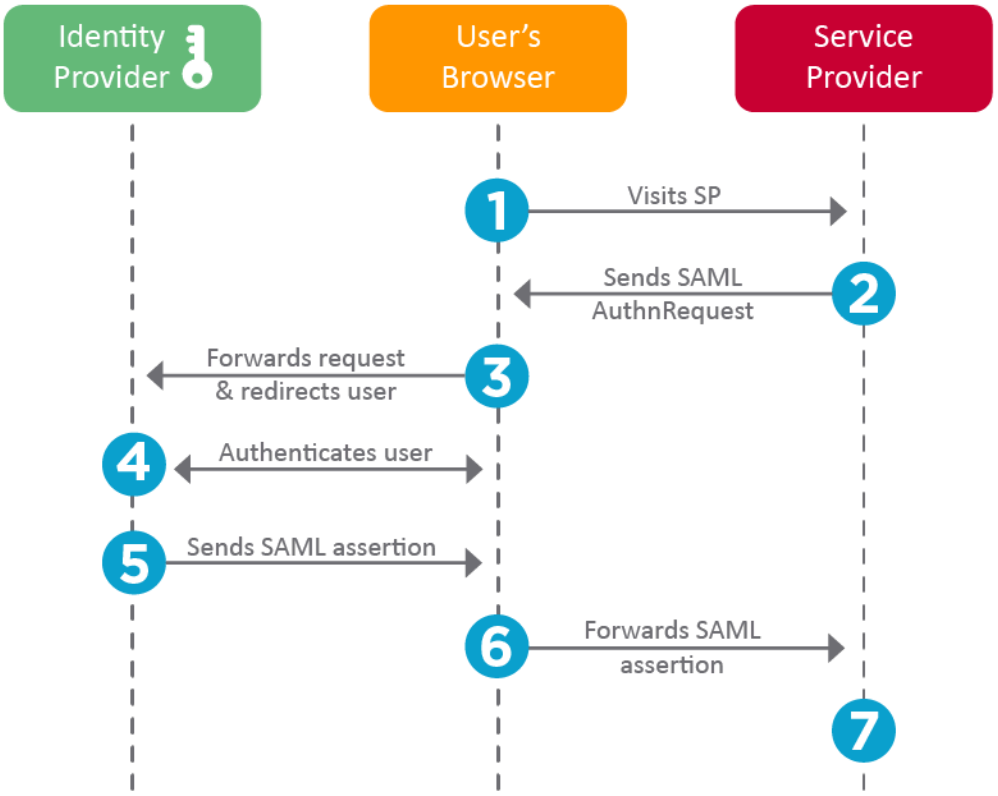


1. The user visits the IdP.
2. When the user is authenticated, the IdP generates a SAML assertion to the client with the following information:
    a. User's Federation ID.
    b. Additional attributes from the user.
    c. How and When the user authenticated.

d. X509 certificate public key of the IdP.

3. The client forwards the SAML assertion and redirects the user to the SP.
4. SP examines the public key and validates it. Then, it gives the end user access to their services.

- **SP initiated (SAML 2.0):** The SP sends an AuthnRequest to the IdP. The IdP authenticates the user and then returns a SAML assertion.



1. The end user visits the SP. This determines whether the user should authenticate with the IdP by checking the subdomain visited, the user's IP address or other…
2. The SP generates a SAML AuthRequest, and the X509 public key can optionally be included to authenticate the request.
3. The client both forwards the request and redirects the user to the IdP specified in the SAML AuthnRequest.
4. The IdP checks whether the user has a currently authenticated session open. If not, the IdP provides the end user with a login screen.

5. Once the user successfully authenticates, the IdP generates a SAML Assertion and sends this to the client. This contains:
    a. The user's Federation ID.
    b. Additional attributes about the user.
    c. How and when the user authenticated.
    d. The X509 certificate public key of the IdP.
6. The client both forwards the SAML Assertion and redirects the user to the SP.
7. The SP examines the assertion, validating the IdP's X509 certificate public key. The SP then gives the end user access to their services, using the Federation ID as the identifier for the end user.

4.3. OAuth

OAuth is another open standard focused on authorization, specifically on granting third-party applications access to user resources without exposing the user's credentials. It allows SPs to request specific permissions from users, enabling them to access limited data or perform specific actions on the user's behalf. This protocol is particularly useful for social media logins and other situations where users want to share specific information or grant limited access to applications. OAuth offers key advantages:

- **Secure Resource Access**: OAuth enables third-party applications to access authorized resources on an SP's platform without exposing the user's credentials.

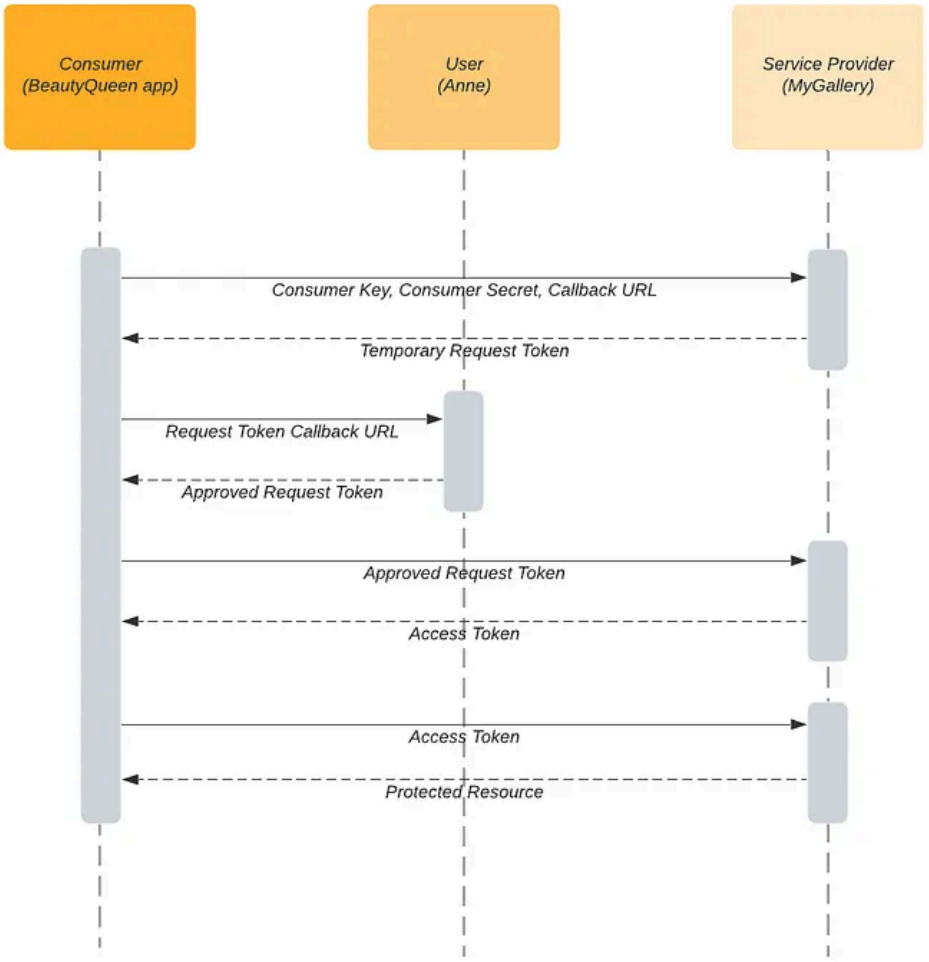
- **Granular Access Control**: It allows users to grant specific access scopes to applications, specifying the actions or data they can access.

- **Flexible Implementation**: It is flexible and can be adapted to various application scenarios, making it a versatile choice for authorization needs.

4.3.1: How does OAuth work?

Let's see how this protocol works with an example of a use case:

1. A user called Anne wants to retrieve some photos stored in the SP *MyGallery* using the application or consumer *BeautyQueenApp*.
2. The SP validates the consumer's request and sends it a temporary request token.
3. The Consumer redirects Anne to the ServiceProvider for login or provides the preformatted URL to log in.
4. When Anne successfully authenticates to the Service Provider, she is asked to grant permission to the Consumer to access her resources. When she finishes granting authentication, the Consumer is informed.
5. The Consumer requests the SP for an access token using its approved temporary request token
6. The SP validates the request and sends an access token.
7. The Consumer can now access Anne's photos from the SP server with the access token without knowledge of Anne's credentials.



4.3.2: OAuth 2.0

This protocol was developed as an open standard to address the need for the continuous exchange of credentials between the client and server. In fact, in 2012, it became the standard replacement for OAuth 1.0 for online authorization.

Specifically, it provides authorized access and restricts the actions that the client application can perform on resources on behalf of the user without sharing the user's credentials. Additionally, it provides the authorization flow for web applications, mobile applications, and desktop programs.

In the following table, we can appreciate the main differences and commonalities between these two versions:

OAuth 1.0	OAuth 2.0
Independent transport (security not delegated to HTTPS / TLS)	**Dependent** transport (security delegated to HTTPS / TLS)
Basic signature workflow	Basic signature workflow
Based on **shared secrets**	Basic on **shared secrets** and **tokens**
Only handles **web workflows**.	Can handle **non-web clients** in addition to **web workflows**.
Easy implementation.	**Complex** implementation.
Less secure and flexible.	**More** secure and flexible.

5. Conclusions

In conclusion, Identity Federation represents a vital solution nowadays. By addressing the challenges of the management of the many digital identities an user could have across multiple platforms, this technology also offers a secure user experience. It unifies credentials and establishes trusted connections between entities, ensuring security and reduces the administrative burden for organizations.

Continual evolution and strategic integration promise a future where identity management evolves into a more efficient, seamless, secure and user-focused system.

6. Bibliography

Examples of IdF:

- *https://www.microsoft.com/es-es/security/business/identity-access/microsoft-entra-id*
- *https://learn.microsoft.com/es-es/entra/fundamentals/whatis*
- *https://learn.microsoft.com/es-es/windows-server/identity/ad-fs/ad-fs-overview*
- *https://cloud.google.com/security/products/identity-platform?hl=es_419*
- *https://cloud.google.com/iam/docs/workload-identity-federation?hl=es-419*

SSO:

- *https://www.geeksforgeeks.org/introduction-of-single-sign-on-sso/*
- *https://www.cloudflare.com/es-es/learning/access-management/what-is-sso/*
- *https://www.chakray.com/what-is-single-sign-on-sso-definition-characteristics-and-advantages/*
- *https://www.onelogin.com/learn/how-single-sign-on-works*

Identity federation protocols:

- *https://www.onelogin.com/learn/federated-identity#:~:text=Federated%20identity%20allows%20authorized%20users,different%20applications%20securely%20and%20efficiently*

OpenID:

- *https://es.wikipedia.org/wiki/OpenID*
- https://openid.net/
- https://auth0.com/es/intro-to-iam/what-is-openid-connect-oidc
- https://openid.net/developers/how-connect-works/

SAML:

- https://www.seidor.com/blog/saml-que-es#:~:text=SAML%20(Security%20Assertion%20Markup%20Language,provider%20(proveedor%20de%20servicios)
- https://www.cloudflare.com/es-es/learning/access-management/what-is-saml/
- https://www.cloudflare.com/learning/access-management/what-is-saml/#:~:text=This%20is%20what%20a%20typical,a%20response%20to%20the%20principal

OAuth:

- https://www.loginradius.com/blog/identity/what-is-oauth/
- https://auth0.com/es/intro-to-iam/what-is-oauth-2
- https://oauth.net/2/

Crèdits transparències GCS Universitat Politècnica de Catalunya