Architecture Decision Record: authentication authorization options

Web application authentication and authorization are two crucial concepts in securing access to applications and services. Both deal with the identity of users and how permissions are granted, but they focus on different aspects:

- **Authentication** is the process of verifying the identity of a user or system.
- **Authorization** is the process of determining what resources or actions the authenticated user or system can access.

Now, let's dive into the specific protocols and technologies you've mentioned, which are commonly used in modern web applications for managing authentication and authorization.

1. **OAuth (Open Authorization)**

OAuth is an open standard for authorization. It allows a user to grant a third-party application limited access to their resources without sharing their credentials. The key idea is **delegated access**. OAuth is often used in situations where users can log in to a third-party service (e.g., signing in with Google) without directly providing their username and password to the third-party.

- **Flow**: OAuth typically follows a **token-based** flow, where an authorization server issues an access token to the third-party application. This token represents the user's permissions, and the application uses it to access the user's data or resources from an API.
- **Example**: A user signs into a third-party app using their Google account. Google verifies the user's identity and then grants a token that allows the third-party app to access some Google data (e.g., Google Calendar).

OAuth **does not** handle authentication directly; it’s about granting access. For authentication, OAuth is often paired with other protocols, like **OpenID Connect**.

2. **OpenID Connect (OIDC)**

OpenID Connect (OIDC) is an identity layer built on top of **OAuth 2.0** that adds authentication to OAuth’s authorization capabilities. Essentially, OpenID Connect extends OAuth to handle **user authentication** and provides a standardized way for applications to verify the identity of a user.

- **Flow**: When a user logs in using OpenID Connect, the third-party application requests an ID token (in addition to the OAuth access token). The ID token contains information about the user (such as their username, email, and other claims). This allows the application to know who the user is and whether they are authenticated.
- **Example**: Logging into a service like Slack using your Google account (Google being the OpenID Connect provider) involves authentication via OpenID Connect, while OAuth manages the access to your Google resources.

OIDC makes it easier for third-party apps to **authenticate users** while still allowing fine-grained control over what resources those apps can access.

3. **SAML (Security Assertion Markup Language)**

SAML is an older, XML-based standard used for exchanging authentication and authorization data between parties, particularly in **Single Sign-On (SSO)** scenarios. It's primarily used in enterprise environments to enable users to authenticate once and access multiple applications without re-entering credentials.

- **Flow**: The user first authenticates with an identity provider (IdP). The IdP generates a signed **SAML assertion** that includes the user's identity and related attributes. The assertion is sent to the service provider (SP), which uses it to authorize access to the application.
- **Example**: An employee logs into their corporate portal (the IdP) and is automatically logged into other systems like email, CRM, etc., without re-entering credentials. The authentication process is based on the SAML assertion sent by the IdP.

SAML is commonly used in **enterprise SSO solutions** and works well for web applications in corporate environments, but it’s less mobile-friendly compared to OAuth/OIDC.

4. **WS-Federation (Web Services Federation)**

WS-Federation is another protocol used for **Single Sign-On (SSO)**, especially in Microsoft-based enterprise environments. It's part of the *WS- (Web Services)** family of specifications and allows for identity federation across different security domains (such as between different organizations or between different services).

- **Flow**: WS-Federation allows a **trusted identity provider (IdP)** to authenticate users and issue tokens that the service provider can use for authorization. It’s similar to SAML but is often used in scenarios that rely heavily on Microsoft technologies.
- **Example**: A user logs into an enterprise application hosted by Microsoft Azure Active Directory (AD), and their identity can be used to access other federated services, including applications hosted by third-party vendors.

Although WS-Federation is largely replaced by newer protocols like OAuth2.0 and OpenID Connect in many modern web environments, it’s still used in legacy systems, especially in Microsoft-centric enterprises.

5. **LDAP (Lightweight Directory Access Protocol)**

LDAP is a protocol used to access and manage directory services, commonly used for **storing user credentials** and managing access control in a centralized directory (often called a **Directory Service**). LDAP isn't specifically about authentication or authorization but is used to store and retrieve identity data, which is then used in those processes.

- **Authentication**: LDAP allows an application to authenticate users by querying the directory service for credentials (like passwords).
- **Authorization**: It also manages user roles and permissions, helping determine whether a user has access to certain resources.
- **Example**: Many enterprises use LDAP-based directories (e.g., **Active Directory**) for authentication and authorization, especially within Windows environments.

LDAP is crucial for enterprises to manage user access across internal systems, but in a modern web context, LDAP is often integrated with other protocols like SAML or OAuth for more complete identity management.

6. **Social SSO Providers**

Social **Single Sign-On (SSO)** providers like **Facebook**, **Google**, **Twitter**, **GitHub**, and others allow users to authenticate into third-party applications using their social media credentials. This is a type of **OAuth-based authentication** where the third-party service (e.g., Google) is the identity provider.

- **Flow**: The user clicks "Log in with Google" (for example). The app redirects to Google, where the user logs in (if not already logged in). Google then provides an access token or ID token to the third-party app, which can be used to authenticate the user and possibly access their data.
- **Example**: Many applications allow you to log in using your Google or Facebook credentials. The app will use OAuth or OpenID Connect behind the scenes to verify your identity and, in some cases, access certain social media data.

Social SSO is a convenient and widely adopted method for authentication because it reduces friction for users, who may not want to create yet another username and password.

Summary of Differences:

- **OAuth**: Used for authorization, allows third-party apps to access user data without exposing credentials.
- **OpenID Connect**: Extends OAuth to provide authentication, enabling apps to verify user identity.
- **SAML**: XML-based protocol used for SSO, often in enterprise environments.
- **WS-Federation**: A Microsoft-specific protocol for identity federation, used in legacy systems.
- **LDAP**: A protocol for querying directory services to authenticate users and manage authorization.
- **Social SSO Providers**: OAuth-based systems (like Google, Facebook) that allow third-party apps to authenticate users using their social media credentials.

Each of these technologies has its own strengths and use cases, and in modern applications, you may see a combination of them being used for different aspects of security (e.g., OAuth/OIDC for API access, SAML for enterprise SSO).