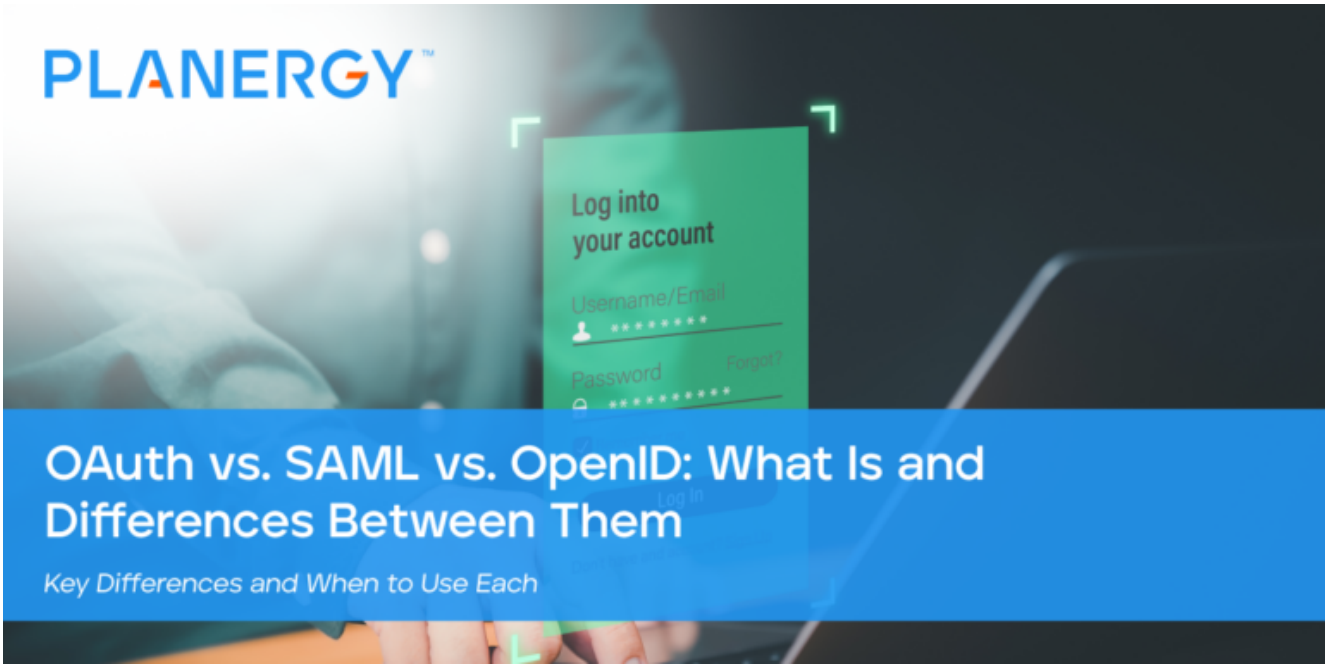
OAuth vs SAML vs OpenID: What Is and Differences Between Them



KEY TAKEAWAYS

- OAuth is best for secure authorization, allowing third-party apps to access user data without exposing credentials.
- SAML is ideal for enterprise Single Sign-On (SSO), enabling authentication across multiple applications with XML-based assertions.
- OpenID simplifies user authentication by letting users log in to multiple sites with one set of credentials.
- Choosing the right protocol depends on the use case: OAuth for delegated access, SAML for enterprise SSO, and OpenID for decentralized authentication.

In online authentication, OAuth, SAML, and OpenID are three key protocols for identity and access management that ensure secure access and streamlined user experiences.

Each protocol serves a unique purpose, has different mechanics, and is suitable for various use cases.

Let’s dive into what each protocol is, how it works, and its main differences to

help you decide which one to use.

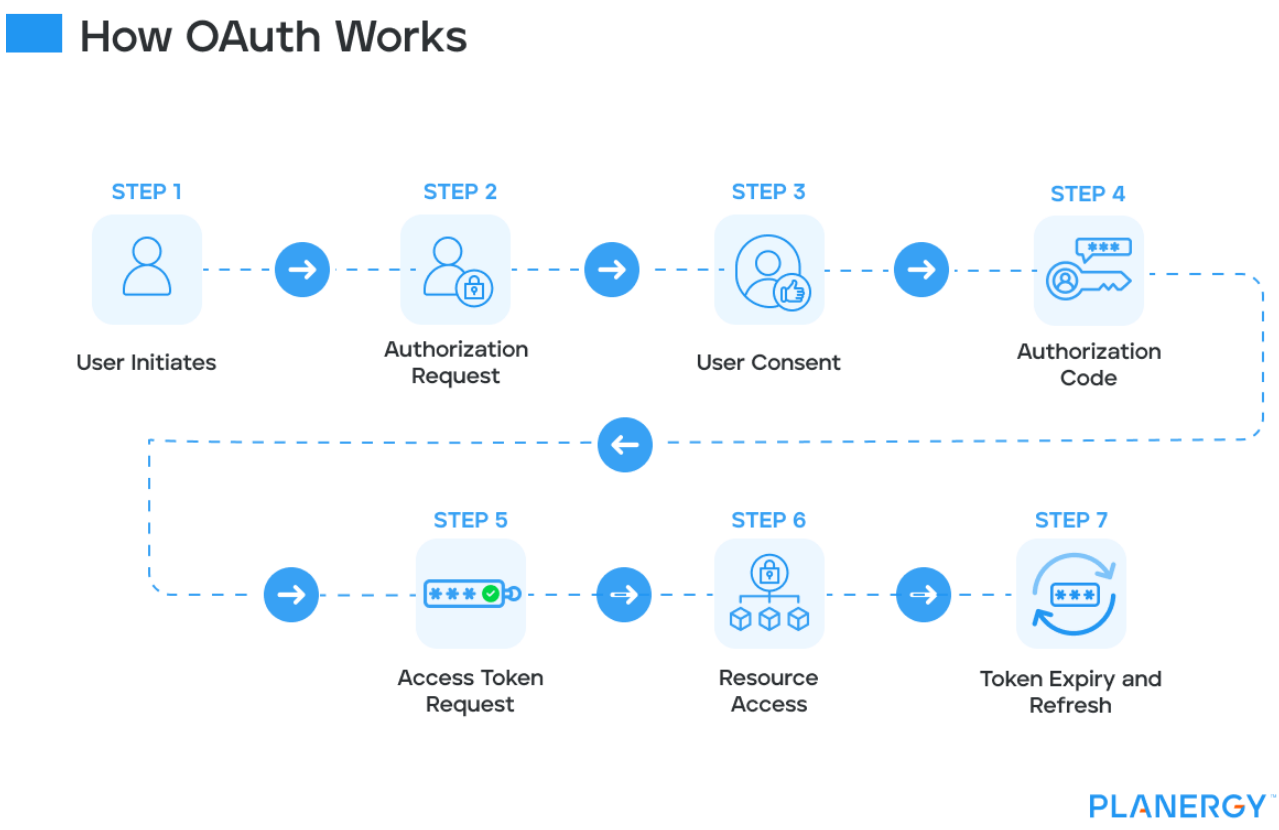
What Is OAuth?

OAuth (Open Authorization) is a token-based authorization framework that allows third-party services to exchange data on a user's behalf without exposing their credentials.

It is commonly used to grant websites or applications limited access to a user's information from another service.

Although it involves multiple steps, the token-based approach ensures privacy and security while allowing seamless integration between services.

How OAuth Works



1. **User Initiates:** The user initiates a request to access a resource on the service provider's site.

For example, a user wants to grant third-party access to their Google Drive or Microsoft OneDrive files.

2. **Authorization Request:** The service provider redirects the user to the authorization server.

   Continuing the above example, the app redirects to Google's or Microsoft's OAuth 2.0 authorization endpoint.

3. **User Consent:** The user logs in and consents to third-party access. If the user isn't already logged in, they're prompted to log in and consent.

   The user logs into their Google account and is presented with a screen asking permission to allow the app to view and manage their Google Drive files.

4. **Authorization Code:** The authorization server redirects back with an authorization code.

   After the user consents, Google redirects them to the app's direct URI with an authorization code in the query string.

5. **Access Token Request:** The client application exchanges the authorization code for an access token.

   The client app sends a POST request to Google's token endpoint.

6. **Resource Access:** The client application uses the access token to access the user's data from the resource server.

   The authorization server responds with an access token and, optionally, a refresh token.

   When requesting the Google Drive API, the app includes the access token in the authorization header.

7. **Token Expiry and Refresh:** If the access token expires, the client application can use the refresh token to obtain a new access token without requiring the user to reauthorize.

   The client app sends a POST request to the token endpoint with the

request for a refresh token, the client ID token, and the client secret.

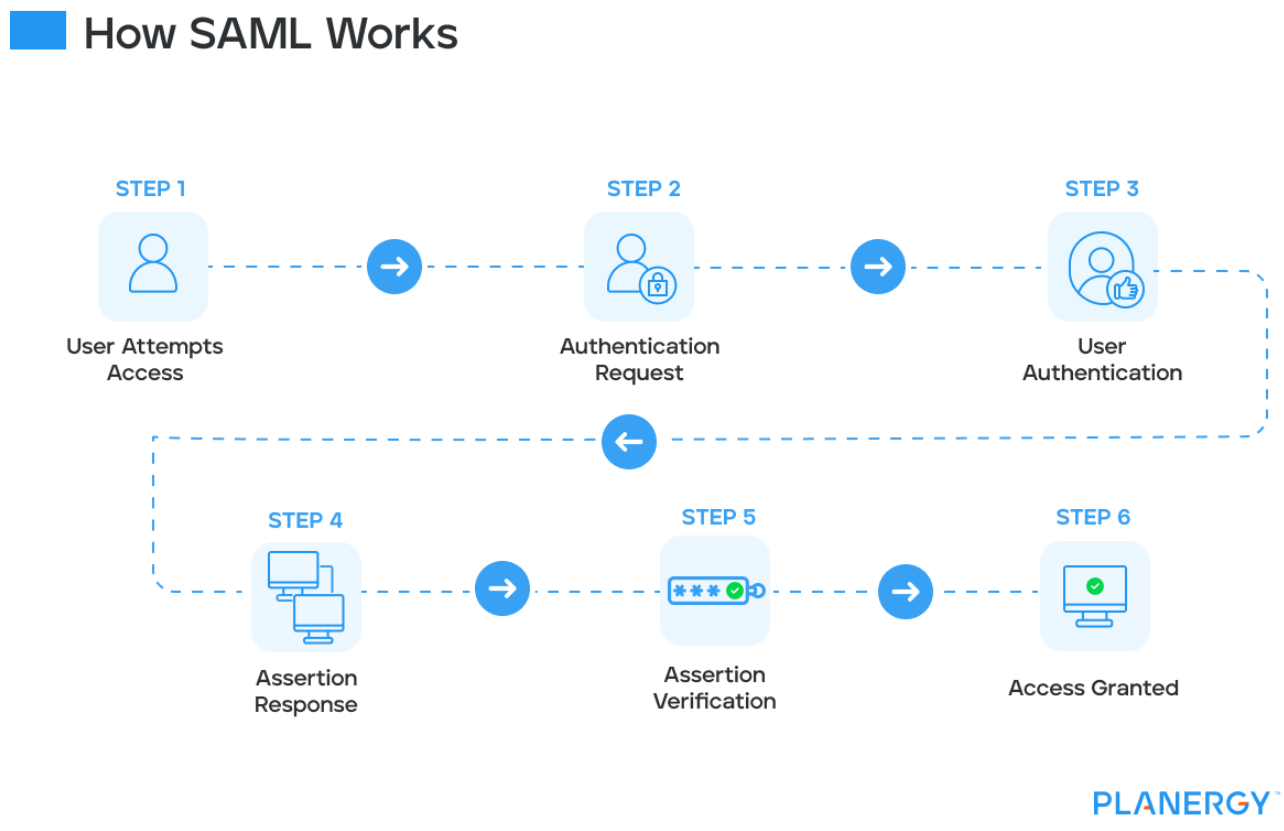
What Is SAML?

SAML (Security Assertion Markup Language) is an XML-based Single Sign-On (SSO) protocol. It allows users to authenticate once and access multiple applications through assertions about their identity.

Its structured approach ensures secure authentication and seamless access to multiple applications, reducing the need for repetitive logins and enhancing security management.

Although it involves multiple steps, it ultimately gives a streamlined and secure user experience.

How SAML Works



1. **User Attempts Access:** The user tries to access an application (Service

Provider).

For example, an employee attempts to access the company's internal HR portal, which acts as the SP.

2. **Authentication Request:** The Service Provider identifies that the user needs to be authenticated and redirects the user to the Identity Provider (IdP) with an authentication request.

   The HR portal redirects the user to the company's IdP, for example, Okta, with a SAML authentication request.

3. **User Authentication:** The user is directed to the Identity Provider, where they login using their credentials.

   If the user is already authenticated because they're already logged into the corporate network, this step may be bypassed.

4. **Assertion Response:** Upon successful authentication, the Identity Provider creates a SAML Assertion, which contains the user's authentication status and attributes (like username, email, and roles).

   A SMAL Assertation is an XML document that includes:

   - **Authentication Statement:** Confirms that the user has been authenticated.
   - **Attribute Statement:** Provides additional information about the user (e.g., roles, group membership).
   - **Authorization Decision Statement:** Specifies the user's authorization to access resources.

   The Identity Provider sends the SAML Assertion back to the Service Provider as part of a SAML Response.

   This response is typically sent via the user's browser (HTTP POST binding) or through a direct connection (HTTP Artifact binding).

5. **Assertion Verification:** The Service Provider receives the SAML Response and verifies the SAML Assertion.

   The SP checks the assertion's digital signature to ensure it was issued by

a trusted Identity Provider. It also validates the assertion's conditions, such as time validity and audience restrictions.

6. **Access Granted:** Upon successful verification, the Service Provider grants the user access to the requested resource.

   The HR portal grants the employee access to the internal resources based on the information provided in the SAML Assertion.

What Is OpenID?

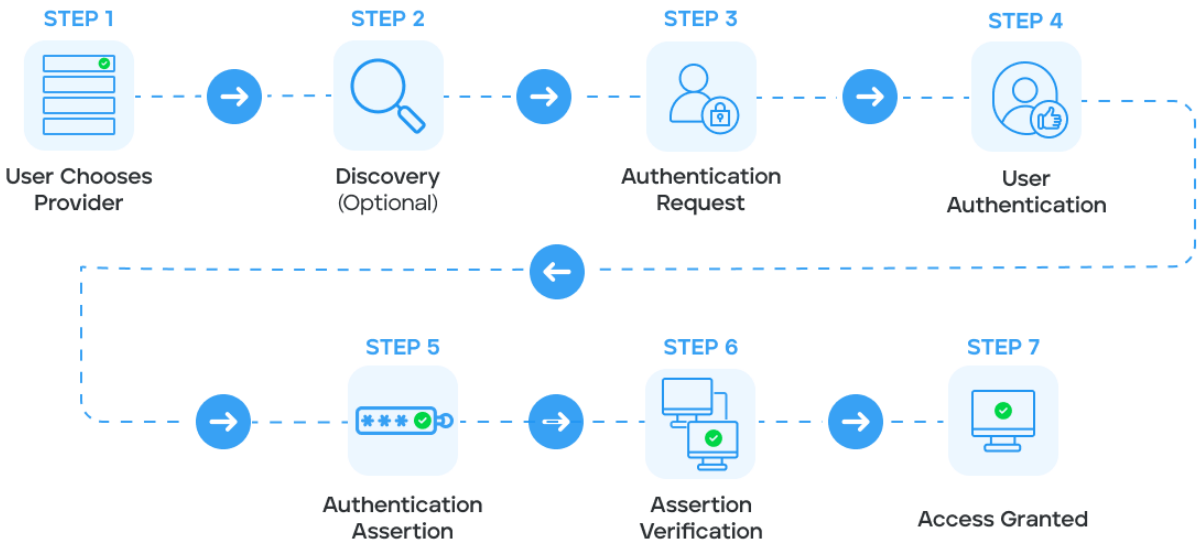
OpenID is an authentication protocol that allows users to log in to multiple sites using a single set of credentials maintained by an OpenID provider.

It provides decentralized authentication to simplify the login process and enhance security by reducing the number of passwords users need to manage.

The protocol uses a series of redirections and assertions to ensure that authentication is seamless and secure.

How OpenID Works

How OpenID Works



1. **User Chooses Provider:** The user selects an OpenID provider. This step can involve the user entering their OpenID identifier (a URL) on the relying party's login page.

   For example, a user who wants to log in to a forum chooses Google as their OpenID provider.

2. **Discovery (Optional):** The relying party performs discovery on the OpenID identifier provided by the user to find the location of the OpenID provider.

   This step involves performing a Yadis or XRDS discovery to find the OpenID provider endpoints. The forum identifies Google as the OpenID provider for the given identifier.

3. **Authentication Request:** The relying party (the site where the user wants to log in) redirects the user to the OpenID provider with an authentication request.

   The forum redirects users to Google's authentication endpoint with an

OpenID authentication request.

4. **User Authentication:** The user is directed to the OpenID provider's login page, where they authenticate using their credentials.

   In this case, the user logs into their Google account.

5. **Authentication Assertion:** Upon successful authentication, the OpenID provider creates an authentication assertion and sends it back to the relying party.

   The assertion includes information like the user's OpenID identifier and may include additional attributes if requested.

6. **Assertion Verification:** The relying party verifies the authentication assertion received from the OpenID provider.

   It checks the digital signature to ensure the response is from a trusted OpenID provider and validates the returned identifier.

7. **Access Granted:** Once the assertion is verified, the relying party grants the user access to the requested resource.

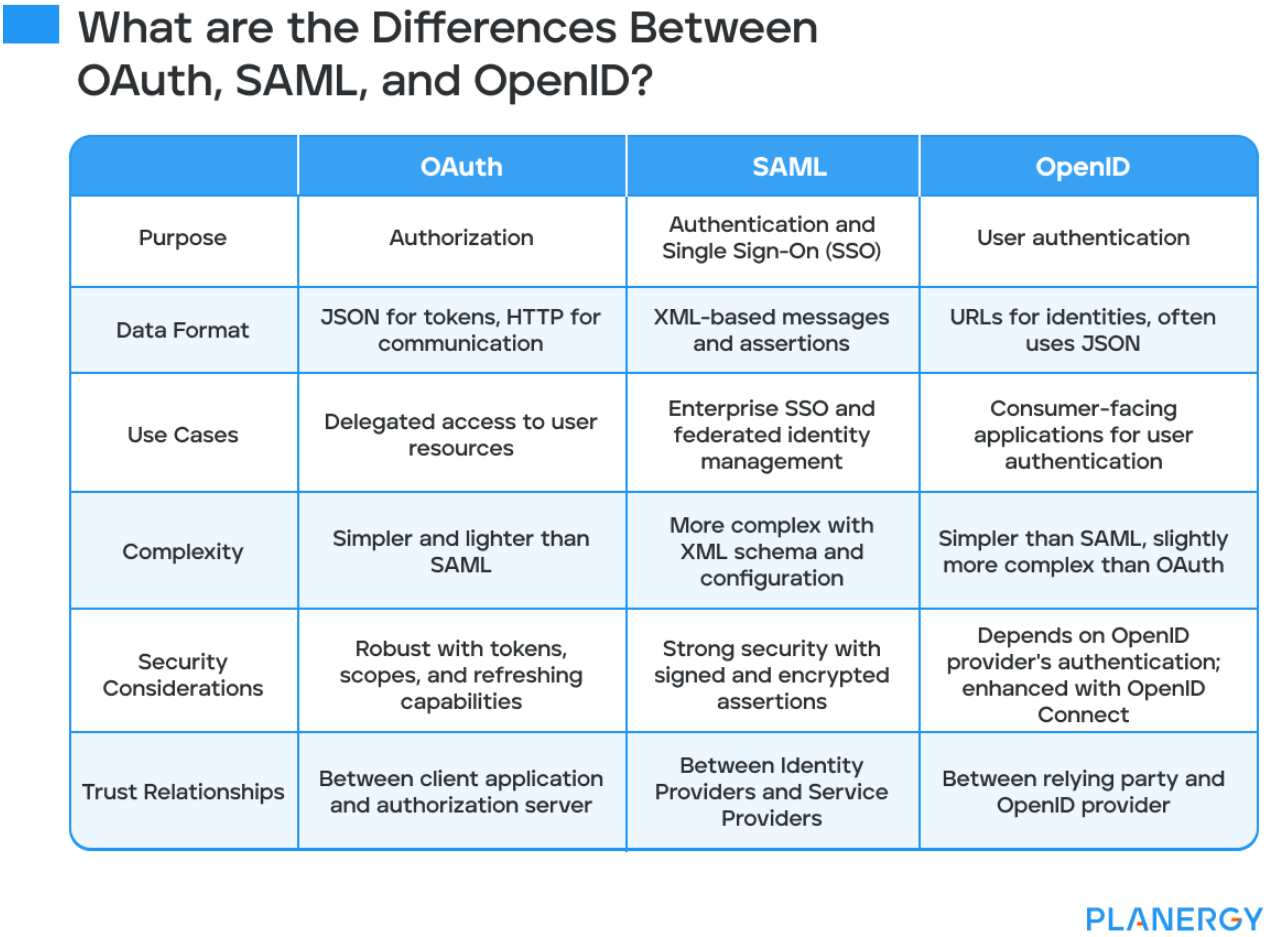
   The forum logs users in and grants them access based on the authenticated OpenID identifier.

How is OIDC used?

OIDC (OpenID Connect) is an identity layer built on OAuth 2.0.

It adds authentication to OAuth's authorization capabilities, with an additional JSON web token (JWT) allowing both authentication and authorization to be used using the same framework.

What are the Differences Between OAuth, SAML, and OpenID?



Purpose

- **OAuth:** Primarily designed for authorization, OAuth allows third-party applications to access a user's resources without exposing their credentials.

  It focuses on granting limited access to resources.

- **SAML:** Primarily designed for Single Sign-On (SSO) within enterprise environments, SAML focuses on providing authentication and authorization across multiple applications using assertions.

- **OpenID:** Primarily designed for user authentication, OpenID enables users to log in to multiple websites using a single set of credentials provided by an OpenID provider.

Data Format

- **OAuth:** Uses JSON for its token format and HTTP for communication. The tokens are typically short-lived and can be refreshed.
- **SAML:** Uses XML-based messages for communication. Assertions are structured in XML and include detailed information about the user.
- **OpenID:** Utilizes URLs to represent user identities and often uses JSON for its data format. The protocol involves exchanging tokens and assertions.

Use Cases

- **OAuth:** Ideal for scenarios where applications need delegated access to user resources without handling user credentials directly.
- **SAML:** Best suited for enterprise SSO solutions where secure, federated identity management is required across various internal systems.
- **OpenID:** Suitable for consumer-facing applications requiring simple and decentralized user authentication.

Complexity

- **OAuth** is generally simpler and lighter than SAML. It requires implementing token handling mechanisms but is straightforward to deploy.
- **SAML** is more complex due to its XML schema and extensive configuration requirements. It involves setting up metadata, certificates, and intricate trust relationships.
- **OpenID:** Simpler than SAML but slightly more complex than OAuth due to the additional discovery and identifier mechanisms.

Security Considerations

- **OAuth:** Provides robust security mechanisms through tokens, scopes, and refreshing capabilities.

  However, improper implementation can lead to security vulnerabilities like token leakage.

- **SAML** offers strong security due to the use of signed and encrypted XML assertions.

  However, its complexity can make it harder to implement securely without expertise.

- **OpenID:** Security depends on the strength of the OpenID provider's authentication mechanism. Combining OpenID with OAuth 2.0 (OpenID Connect) enhances security by adding standardized authentication.

Trust Relationships

- **OAuth:** Establishes trust relationships between the client application and the authorization server. The resource server trusts the authorization server to validate tokens.

- **SAML:** Establishes trust relationships between Identity Providers and Service Providers through metadata exchange and certificate validation.

- **OpenID:** Establishes trust between the relying party and the OpenID provider. The relying party must trust the OpenID provider to authenticate users accurately.

OAuth 2.0 vs. OpenID Connect

OAuth 2.0 focuses primarily on authorization and does not provide any user authentication mechanism.

OpenID Connect, however, builds an identity layer on top of OAuth 2.0 to offer a standard way to authenticate users while still leveraging OAuth 2.0 for

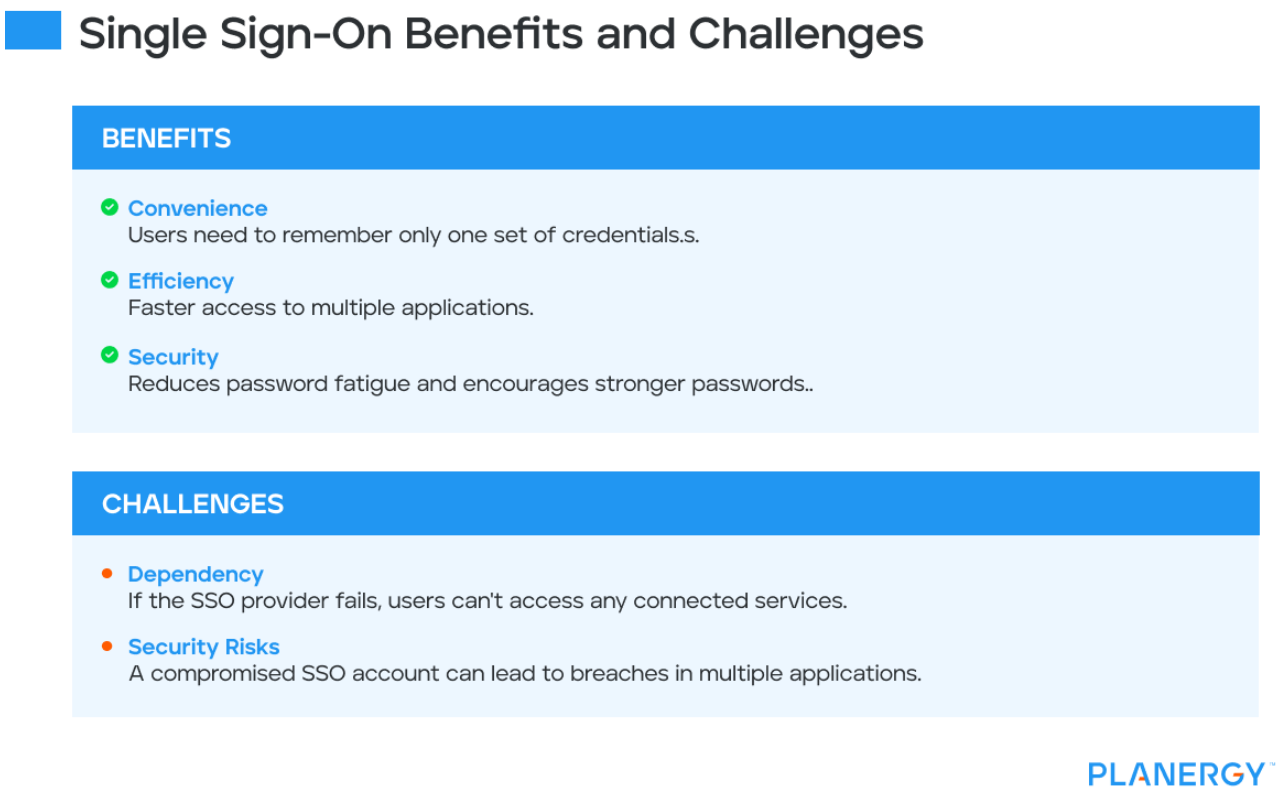
authorization.

SSO vs. OpenID

Single Sign-On (SSO) allows users to authenticate once and gain access to multiple systems without re-authenticating.

OpenID is a decentralized authentication protocol that enables users to log in to multiple unrelated sites using a single set of credentials.

Single Sign-On Benefits and Challenges



- **Benefits:**
  - **Convenience:** Users need to remember only one set of credentials.
  - **Efficiency:** Faster access to multiple applications.
  - **Security:** Reduces password fatigue and encourages stronger passwords.

- **Challenges:**
    - **Dependency:** If the SSO provider fails, users can't access any connected services.
    - **Security Risks:** A compromised SSO account can lead to breaches in multiple applications.

Which Is More Secure: SAML 2.0 or OAuth2?

SAML 2.0 and OAuth2 have robust security measures, but their security depends heavily on implementation.

SAML is often considered more secure for enterprise SSO due to its stringent protocols, while OAuth2's token-based system is excellent for API security when properly implemented.

Which of the Three Protocols To Use?

Choosing between OAuth, SAML, and OpenID depends on your specific requirements:

- **Use OAuth:** When you need secure, delegated access to user resources without handling credentials directly, ideal for mobile applications.
- **Use SAML:** For enterprise SSO, where robust, fine-tuned access control, federated authentication, and authorization are needed across multiple applications.
- **Use OpenID:** For consumer-facing applications that require simple, decentralized authentication.

Choosing the right protocol hinges on your specific needs, the environment, and the level of security required.

Each protocol offers unique strengths tailored to different scenarios, ensuring secure and efficient authentication and authorization.

What’s your goal today?

1. Use PLANERGY to manage purchasing and accounts payable

We’ve helped save billions of dollars for our clients through better spend management, process automation in purchasing and finance, and reducing financial risks. To discover how we can help grow your business:

- Read our case studies, client success stories, and testimonials.
- Visit our Workflow Automation Software page to see how PLANERGY can digitize and automate your processes saving you time and money.
- Learn about us, and our long history of helping companies just like yours.

Book a Live Demo

2. Download our guide “Preparing Your AP Department For The Future”

Download a free copy of our guide to future proofing your accounts payable department. You’ll also be subscribed to our email newsletter and notified about new articles or if have something interesting to share.

download a free copy of our guide

3. Learn best practices for purchasing, finance, and more

Browse hundreds of articles, containing an amazing number of useful tools, techniques, and best practices. Many readers tell us they would have paid consultants for the advice in these articles.

Related Posts