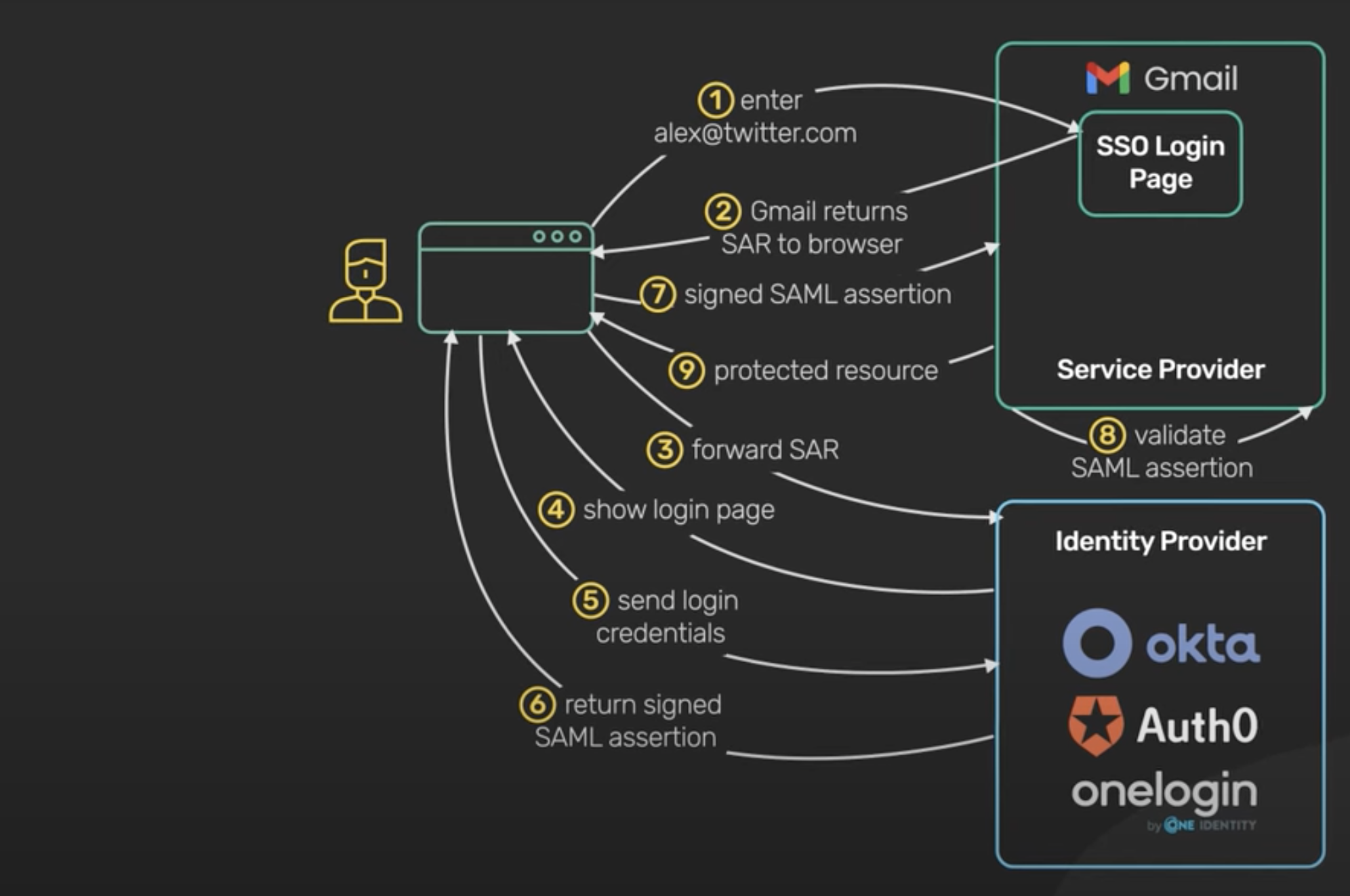
# How SSO Works?

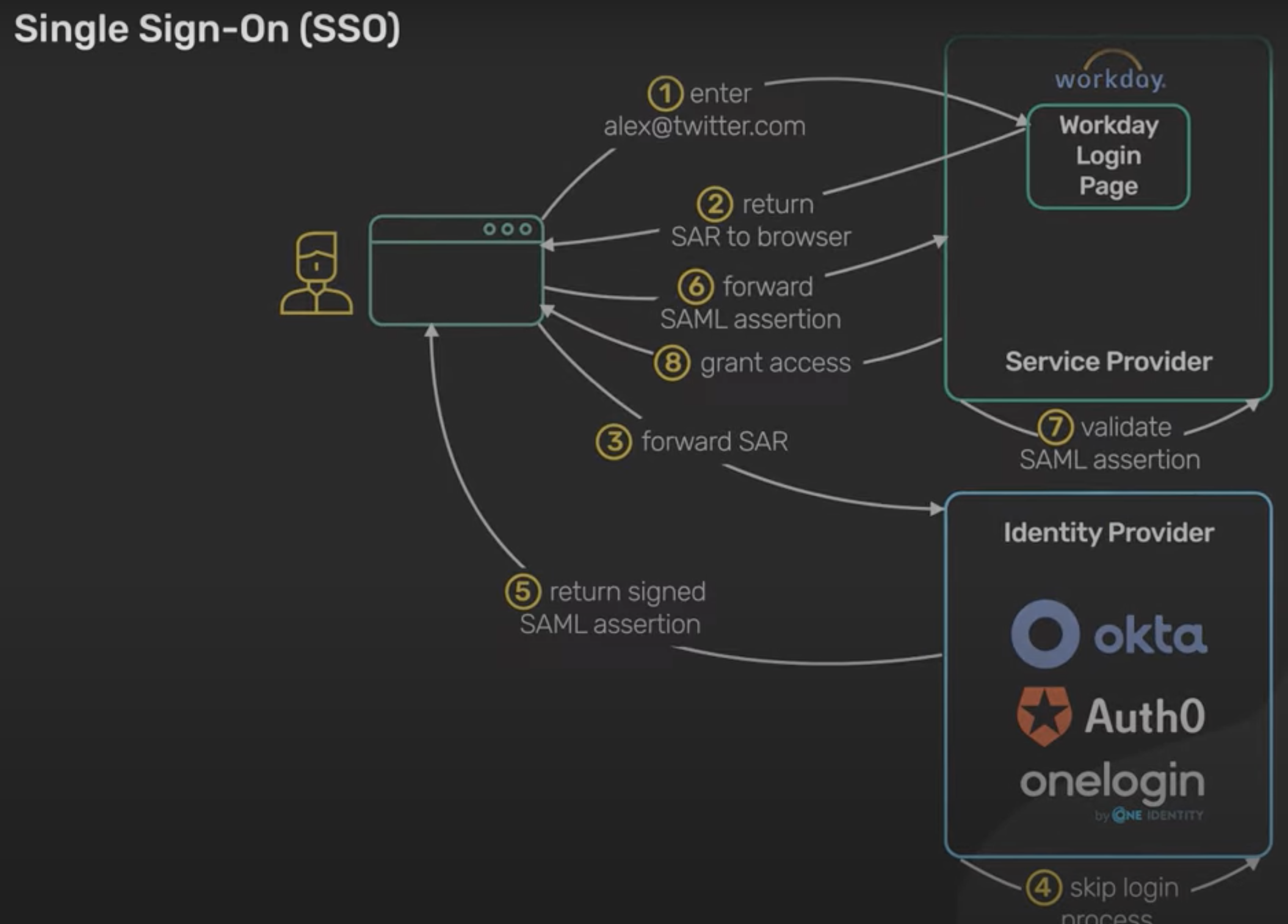
Imagine a large organization with various web applications used by employees, such as email, intranet, and project management tools. Instead of requiring employees to log in separately to each application, the organization implements SSO.

1. **Authentication Request**:
   * An employee attempts to access the email application.
   * The email application recognizes that the user is not authenticated and redirects them to the SSO service provider.
2. **Authentication Process**:
   * The SSO service provider prompts the employee to log in using their username and password.
   * The SSO service provider authenticates the user against the organization's identity provider (e.g., Active Directory, LDAP).
3. **Token Generation**:
   * Upon successful authentication, the SSO service provider generates a SAML token containing information about the user and their authentication status.
   * This token is digitally signed by the SSO service provider to ensure its integrity.
4. **Token Response**:
   * The SSO service provider sends the SAML token back to the employee's browser.
5. **Access Request**:
   * The employee's browser forwards the SAML token to the email application.
6. **Token Verification**:
   * The email application validates the SAML token's digital signature with the SSO service provider.
   * The email application extracts user information from the SAML token to determine the user's identity and authentication status.
7. **Access Granted**:
   * If the token is valid and the user is authorized to access the email application, access is granted without requiring the user to log in again.

When user is login to the 1st app:



When User is already logged-in and trying to login to 2nd app:



Let’s compare different technology :

| **Technology** | **Description** | **Use Cases** | **Authentication Protocol** | **Key Features** |
| --- | --- | --- | --- | --- |
| Security Assertion Markup Language (SAML) | XML-based protocol for SSO between an identity provider (IdP) and a service provider (SP). | Enterprise environments, Cross-domain SSO | XML-based messages (Assertions) | Cross-domain SSO, Wide adoption in enterprise systems |
| OpenID Connect (OIDC) | Authentication layer built on OAuth 2.0, providing an identity layer for web applications. | Consumer-facing applications, Modern authentication | OAuth 2.0, JSON-based tokens | Modern authentication features, User-centric design |
| OAuth 2.0 | Authorization framework allowing third-party applications limited access to a user's resources. | API authorization, Access delegation | Authorization framework | Access delegation, Wide adoption in API ecosystems |
| JSON Web Tokens (JWT) | Compact, URL-safe format for transferring claims between parties, often used as SSO tokens. | OIDC, Custom SSO implementations | Custom | Self-contained, Verifiable, Flexible |
| Lightweight Directory Access Protocol (LDAP) | Protocol for accessing and maintaining directory services, used for centralizing user authentication. | Centralized authentication, Legacy systems | N/A | Centralized user management, Integration with directory services |
| Federation Services | Services facilitating secure SSO across different security domains, often based on standards like SAML or OIDC. | Inter-domain SSO, Trust relationships | SAML, OIDC | Inter-domain SSO, Trust establishment, Integration with existing systems |