OAuth 2.0

<https://developer.okta.com/docs/concepts/auth-overview/#oauth-2-0>

OAuth 2.0 è uno standard utilizzato dalle app per fornire accesso alle applicazioni client.

Tramite il protocollo OAuth 2.0 si consente l'accesso ai dati dell'applicazione in modo sicuro.

OAuth 2.0 ha quattro ruoli importanti:

* Il "Authorization Server", ovvero il server che emette il token di accesso.

In questo caso il server di autorizzazione è Okta.

* Il "Resource Owner", normalmente è l'utente finale dell'applicazione, che concede l'autorizzazione ad accedere al server delle risorse con un token di accesso.
* Il "Client", che è l'applicazione che richiede il token di accesso da Okta e quindi lo passa al server delle risorse.
* Il "Resource Server ", che accetta il token di accesso e deve verificarne la validità. In questo caso questa è la tua applicazione.

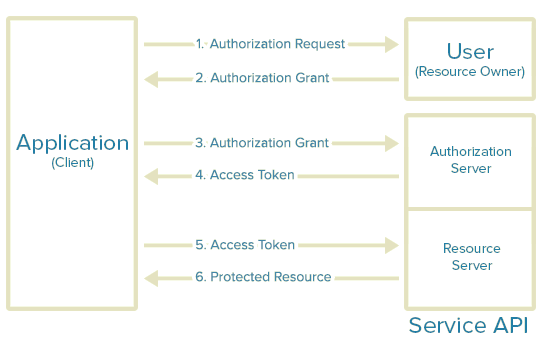
Altri termini importanti:

* Un "Grant" OAuth 2.0 è l'autorizzazione data (o "concessa") al Client da parte dell’utente.

Esempi di grants sono "authorization code" e "client credentials".

Ogni grant OAuth ha un flusso corrispondente, spiegato di seguito:

* Il "token di accesso" viene emesso dal server di autorizzazione (Okta) in cambio della GRANT.
* Il "token di aggiornamento" è un token opzionale che viene scambiato con un nuovo token di accesso se il token di accesso è scaduto.



<https://habr.com/en/post/449182/>

1. L'applicazione richiede l'autorizzazione per accedere alle risorse del servizio da parte dell'utente. L'applicazione deve fornire l'ID client, il secret client, l'URI di reindirizzamento e gli ambiti richiesti.
2. Se l'utente autorizza la richiesta, la domanda riceve una GRANT di autorizzazione
3. L'applicazione richiede un token di accesso dal server di autorizzazione presentando l'autenticazione della propria identità e la GRANT di autorizzazione
4. Se l'identità dell'applicazione è autenticata e la concessione dell'autorizzazione è valida, il server delle autorizzazioni emette il token di accesso e aggiornamento (se richiesto) all'applicazione. L'autorizzazione è completa.
5. L'applicazione richiede la risorsa dal server delle risorse e presenta il token di accesso per l'autenticazione.
6. Se il token di accesso è valido, il server delle risorse serve la risorsa all'applicazione

Ecco alcuni dei principali vantaggi e svantaggi di OAuth 2.0:

* OAuth 2.0 è più facile da usare e implementare (rispetto a OAuth 1.0)
* Ampia diffusione e continua crescita
* Token di breve durata
* Token incapsulati
* Nessuna firma (si basa esclusivamente su SSL / TLS), token al portatore
* Nessuna sicurezza integrata
* Può essere pericoloso se utilizzato da persone non esperte
* Integrazione mobile (visualizzazioni Web)
* Le specifiche Oauth 2.0 non sono un protocollo, ma piuttosto un framework - RFC 6749

OpenID Connect

OpenID Connect è uno standard di autenticazione basato su OAuth 2.0. Aggiunge un token aggiuntivo chiamato token ID.

OpenID Connect standardizza anche le aree che OAuth 2.0 lascia a scelta, come ambiti, rilevamento degli endpoint e registrazione dinamica dei client.

Okta è certificata OpenID.

Sebbene il protocollo OpenID Connect (OIDC) sia basato su OAuth 2.0, la specifica utilizza termini leggermente diversi per i ruoli nei flussi:

* Il "provider OpenID", che è il server di autorizzazione che emette il token ID è il provider OpenID, in questo caso Okta.
* L '"utente finale" le cui informazioni sono contenute nel token ID.
* "Relying party", ovvero l'applicazione client che richiede il token ID da Okta.
* Il "token ID" viene emesso dal provider OpenID e contiene informazioni sull'utente finale sotto forma di reclami.
* Un "reclamo" è un'informazione sull'utente finale.

Il flusso di alto livello ha lo stesso aspetto sia per OIDC che per i flussi OAuth 2.0 regolari, la differenza principale è semplicemente che un flusso OIDC si traduce in un token ID, oltre a qualsiasi token di accesso o di aggiornamento.

Choosing an OAuth 2.0 Flow

<https://developer.okta.com/docs/concepts/auth-overview/#choosing-an-oauth-2-0-flow>

Esempio Spring:

<https://www.baeldung.com/spring-security-openid-connect>