Webbtjänster med API er

Mål med lektionen!

- Titta på Oauth
- Titta på OpenID
- Titta på OWASP TOP 10

Vad lektionen omfattar

- Vad är Oauth?
- Vad är OpenID?
- Vad är OWASP?

Oauth2?

Oauth2 är ett autentiserings ramverk som låter applikationer ge en begränsad tillgång med till användarkonton på en HTTP tjänst såsom Facebook, Google, GitHub och DigitalOcean.

Det fungerar så som att det delegerar användar autentiseringen till tjänsten som hostar användarkontot, och tillåter tredjeparts applikationer att komma åt kontot.

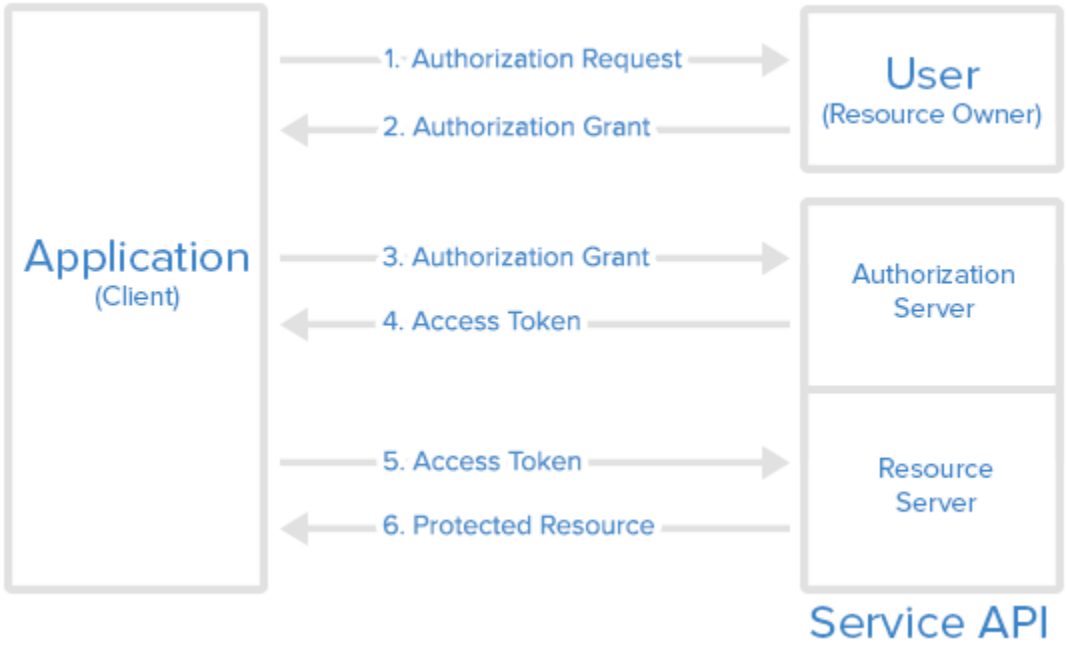
Oauth2 tillför autentiseringsflöden för web, desktop och mobila applikationer.

4 roller

Oauth definierar 4 roller:

- Resurs ägare.
- Klient.
- Resurs server.
- Autentiserings server.

Protokoll flödet



Registrering av applikationen

Innan du använder OAuth med din applikation, måste du registrera din applikation med tjänsten. Detta görs via en registreringsformulär i delen"utvecklare" eller "API" på tjänstens webbplats där du kommer att skicka in följande information (och förmodligen detaljer om din applikation):

- Applikationsnamn.
- Applikationswebbplats.
- Omdirigerings URI eller callback URL.

Klient ID och Klient hemlighet

När din applikation är registrerad kommer tjänsten att utfärda "klientuppgifter” (client credentials) i form av en klientidentifierare och en klienthemlig.

- Klient ID
- Klient hemlighet

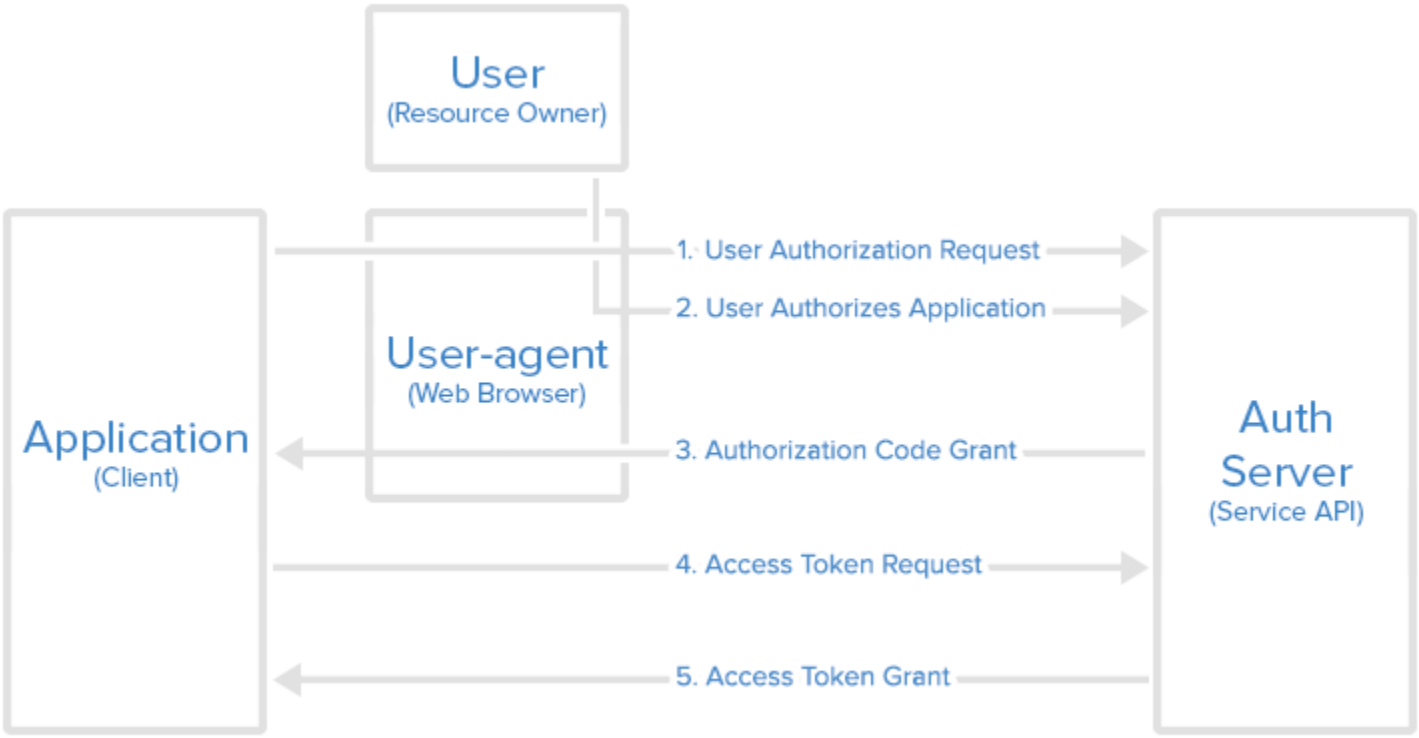
Tillstånd beviljas

I det abstrakta protokollflödet i sliden innan där de fyra första stegen täcker att man erhåller ett beviljat tillstånd och accesstoken.

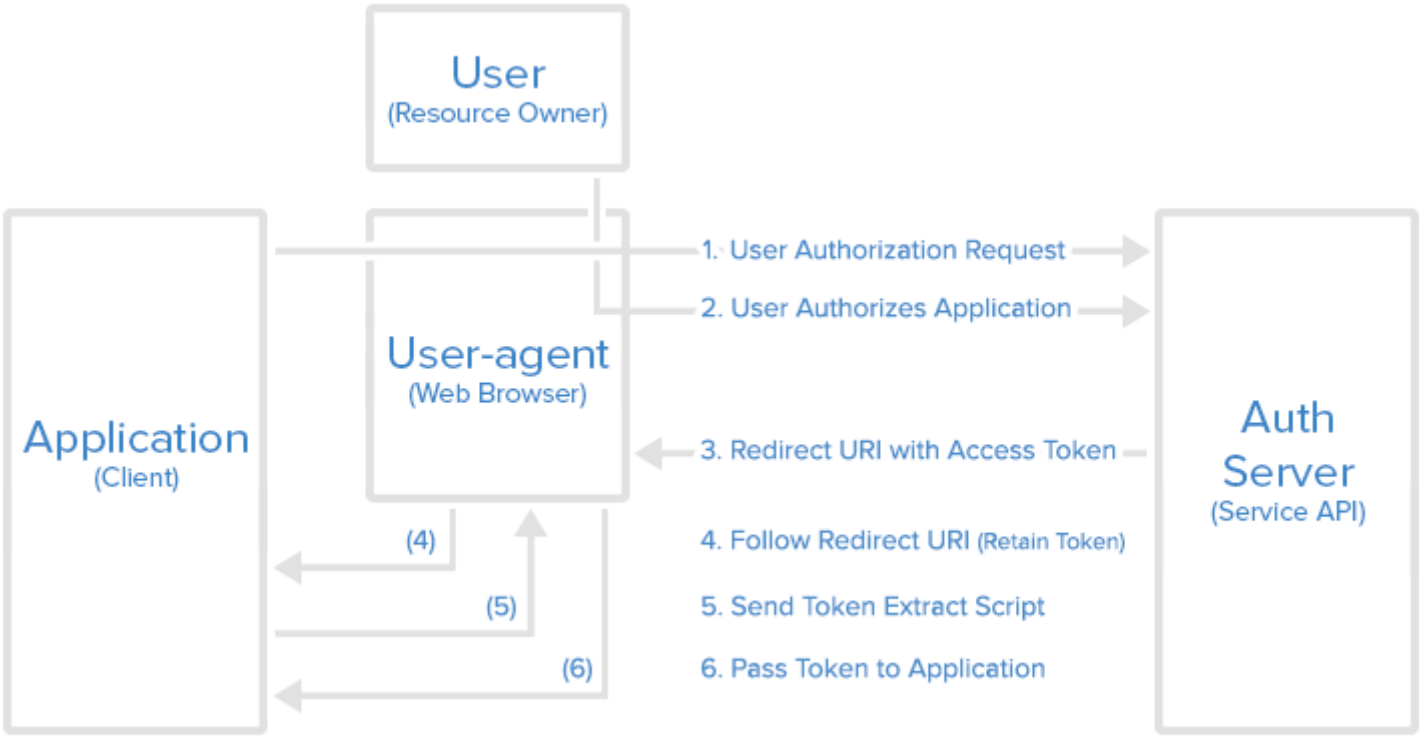
Typen för beviljandet av tillståndet beror på vilken metod som används av applikationen för att begära tillstånd och de tillståndstyper som stöds av API:et.

OAuth 2 definierar fyra bidragstyper, vilka var och en är användbara i olika fall:

Typ av beviljan: Tillståndskod (med flödet)



Typ av beviljan: Implicit (med flödet)



Typ av beviljan: Resurs ägarens lösenords referenser

Med hjälp av resursägarens lösenords behörighetstyp, tillhandahåller användaren sina användaruppgifter (användarnamn och lösenord) direkt till programmet, som använder referensuppgifterna för att få accesstoken från tjänsten. Denna tillståndstyp bör endast aktiveras på auktoriseringsservern om andra flöden inte är genomförbara.

Typ av beviljan: Klient referenser

Klientens legitimationstyp (referenser) ger en applikation ett sätt att komma åt sitt eget servicekonto.

Exempel på när det här kan vara användbart är om en applikation vill uppdatera sin registrerade beskrivning eller redirect URI, eller få åtkomst till annan data som lagras i sitt servicekonto via API:et.

Applikationen begär en accesstoken genom att skicka sina uppgifter, dess klient-ID och klienthemlighet till auktorisationsservern. Om applikationsuppgifterna stämmer, returnerar behörighetsservern en accesstoken till programmet. Nu är applikationen behörig att använda sitt eget konto!

Fördelar med token-baserad autentisering

- **Skalbarhet för servrar**
- **Lös koppling**
- **Mobilvänlig**

Local user authentication vs Identity Providers

Applikationer måste ofta identifiera sina användare. Det enklare sättet är att skapa en lokal databas för användarnas konton och referenser.

Om vi tänker på hur vi skall hantera det tekniska och underhålla detta sedan, så kan man göra detta bra. Lokal autentisering kan dock vara dålig för företag:

Local user authentication vs Identity Providers

Den etablerade lösningen på dessa problem är att delegera användarautentisering och tillhandahålla en dedikerad, specialbyggd tjänst, kallad en Identity Provider (IdP).

Google, Facebook och Twitter, där många personer på internet är registrerade, erbjuder sådana IDP-tjänster till sina användare. En konsumentwebbplats kan kraftigt effektivisera att användaren ”kommer ombord” genom att integrera inloggning med dessa IdPs.

OpenID Connect

OpenID Connect, publicerad första gången 2014, är inte den första standarden för IdP, utan enligt en del det bästa när det gäller användbarhet och enkelhet, efter att ha tagit lärdomar ifrån tidigare insatser som SAML och OpenID 1.0 och 2.0.

Vad är OpenID Connect’s formel för framgång?

- Lätt att konsumera identitetstoken.
- Baserat på OAuth 2.0-protokollet.
- Enkelhet

Identity token

ID-token liknar konceptet med ett identitetskort, i ett standard JWT-format, signerat av OpenID leverantören. För att få en måste klienten skicka användaren till sin OpenID leverantör med en autentiseringsförfrågan.

ID token features

- Anger användarens identitet, som heter subject i OpenID (sub).
- Anger utfärdande bemyndigandet (iss).
- Är genererad för en viss publik, dvs klient (aud).
- Kan innehålla ett tillfäle (nonce).
- Kan ange när (auth_time) och hur, med avseende på styrka (acr), användaren var autentiserad.

ID token features

- Har ett utfärdande (iat) och ett utgångsdatum (exp).
- Kan innehålla ytterligare begärda detaljer om ämnet (subject), till exempel namn och e-postadress.
- Är digitalt signerad, så det kan verifieras av avsedda mottagare.
- Kan eventuellt krypteras för konfidentialitet.

ID token objekt

ID-token-påståenden, eller claims, är förpackade i ett enkla JSON-objekt:

```
{
  "sub"       : "alice",
  "iss"       : "https://openid.c2id.com",
  "aud"       : "client-12345",
  "nonce"     : "n-0S6_WzA2Mj",
  "auth_time" : 1311280969,
  "acr"       : "c2id.loa.hisec",
  "iat"       : 1311280970,
  "exp"       : 1311281970,
}
```

```
eyJhbGciOiJSUzI1NiIsImtpZCI6IjFlOWdkazcifQ.ewogImlzcyI6ICJodHRw
Oi8vc2VydmVyLmV4YW1wbGUuY29tIiwKICJzdWIiOiAiMjQ4Mjg5NzYxMDAxIiw
KICJhdWQiOiAiczZCaGRSa3F0MyIsCiAibm9uY2UiOiAibi0wUzZfV3pBMk1qIi
wKICJleHAiOiAxMzExMjgxOTcwLAogImlhdCI6IDEzMTEyODA5NzAKfQ.ggW8hZ
1EuVLuxNuuIJKX_V8a_OMXzR0EHR9R6jgdqrOOF4daGU96Sr_P6qJp6IcmD3HP9
9Obi1PRs-cwh3LO-p146waJ8IhehcwL7F09JdijmBqkvPeB2T9CJNqeGpe-gccM
g4vfKjkM8FcGvnzZUN4_KSP0aAp1tOJ1zZwgjxqGByKHiOtX7TpdQyHE5lcMiKP
XfEIQILVq0pc_E2DzL7emopWoaoZTF_m0_N0YzFC6g6EJbOEoRoSK5hoDalrcvR
YLSrQAZZKflyuVCyixEoV9GfNQC3_osjzw2PAithfubEEBLuVVk4XUVrWOLrL10
nx7RkKU8NXNHq-rvKMzqg
```

ID-tokenhuvudet, fodrar JSON och signaturen kodas in i en bas 64-URL-säker sträng för enkelt skicka runt, till exempel som URL-parameter.

Hur man begär en ID token

Nu när vi vet vad ett ID-token är, hur kan en klient, kallad den litande parten (RelyingParty) i OpenID Connect, begära en?

Autentisering måste ske hos identitetsleverantören, där användarens session eller referenser kan kontrolleras. För det krävs en betrodd agent, och denna roll spelas vanligtvis av webbläsaren.

Hur man begär en ID token

ID-tokens begärs via OAuth 2.0-protokollet, vilket har varit en enorm framgång på egen hand. OAuth utformades ursprungligen som en enkel auktoriseringsmekanism för appar för att få access tokens för webb-API eller andra skyddade resurser. Den har flöden som är utformade för alla apptyper: traditionella server baserade webbapps, endast appar (javascript) i webbläsaren och inbyggda/mobilappar.

För detta kan man användaa någon av tre flöden/vägar.

De tre flödena:

Flow property	Code	Implicit	Hybrid
Browser redirection step	✓	✓	✓
Backend request step	✓	×	✓
Tokens revealed to browser	×	✓	✓
Client can be authenticated	✓	×	✓

ID token användningsområden

ID-token är mångsidig och användningen är säkerligen inte begränsad till att bara logga in användare i appar:

- **Stateless sessions**
- **Passing identity to 3rd parties**
- **Token exchange**

Hur går autentiseringen till?

Vi kommer nu att gå igenom ett minimalt exempel på hur man får en ID-token för en användare från en OP med hjälp av authorisation code flödet. Detta är det vanligaste flödet av traditionella webbapps och inbyggda mobilappar.

	Step 1	Step 2
Purpose	1. Authenticate user 2. Receive user consent	1. Authenticate client (optional) 2. Exchange code for token(s)
Via	Front-channel request (browser redirection)	Back-channel request (app to web server)
To	Authorisation endpoint	Token endpoint
Result on success	Authorisation code (step 2 input)	ID token (+ OAuth 2.0 access token)

Step 1

LitandeParten (Relying Party) initierar användarautentisering genom att redirecta webbläsaren till OAuth 2.0 godkännandeendpointen för OpenID Provider.

Exempel på hur en autentiserings redirect till OP ser ut:

```
HTTP/1.1 302 Found
Location: https://openid.c2id.com/login?
          response_type=code
          &scope=openid
          &client_id=s6BhdRkqt3
          &state=af0ifjsldkj
          &redirect_uri=https%3A%2F%2Fclient.example.org%2Fcb
```

Step 2

Tillståndskoden är en mellanliggande referens, som enkodar(krypterar) den auktorisation som erhölls i steg 1. Den är därför oklarför RP och har endast betydelse för OP-servern. För att hämta ID-token måste RP skicka in koden till OP, men den här gången krävs en direkt back-channel-begäran.

Allt detta görs av två skäl:

- Att autentisera konfidentiella klienter med OP innan de avslöjar tokens.
- För att leverera tokens direkt till RP, och därmed undvika att exponera dem för webbläsaren.

Claims (användar info)

OpenID Connect anger en uppsättning standardkrav eller användarattribut. De är avsedda att tillhandahålla klientapplikationen med samtyckta användaruppgifter, såsom e-post, namn och bild, på begäran. Språkkoder aktiverar lokalisering (dynamiskt hämta valuta, tidszoner osv).

Scope value	Associated claims
email	email, email_verified
phone	phone_number, phone_number_verified
profile	name, family_name, given_name, middle_name, nickname, preferred_username, profile, picture, website, gender, birthdate, zoneinfo, locale, updated_at
address	address

Claims (användar info)

Klienter kan begära claims på två sätt:

- En hel claims kategori med dess räckvidds värde (se tabell på sliden innan för räckvidd för att claim mappings)
- Individuellt med frivillig claims request parameter.

Om du är en apputvecklare, respekterar användarens integritet och behåll de begärda claims:en till det väsentliga. Detta ökar dina chanser för användarkonvertering och kommer också att säkerställa att de senaste lagarna om integritetsskydd följs.

OpenID connect endpoints

Core endpoints	Optional endpoints
■ Authorisation ■ Token ■ UserInfo	■ WebFinger ■ Provider metadata ■ Provider JWK set ■ Client registration ■ Session management

Valfria endpoints

OpenID Connect-leverantörer kan ha dessa ytterligare endpoints:

- **WebFinger**
- **Provider metadata**
- **Provider JWK set**
- **Client registration**
- **Session management**

Lite enkelt skillnaden mellan Oauth-OpenID

OpenID är ett protokoll för autentisering medan OAuth är för tillstånd/bemyndigande. Autentisering handlar om att se till att killen du pratar med är verkligen vem han hävdar att vara. Bemyndigande handlar om att bestämma vad den här killen ska få lov att göra.

I OpenID delegeras autentisering: server A vill verifiera användare U, men U’s referenser (t.ex. U: s namn och lösenord) skickas till en annan server, B, som A litar på (åtminstone litar på för att verifiera användare). Faktum är att server B ser till att U verkligen är U, och berättar sedan till A: "Okej, det är den äkta U".

Lite enkelt skillnaden mellan Oauth-OpenID

I OAuth delegeras behörighet: Enhet A erhåller från enhet B en "åtkomsträtt" som A kan visa för server S för att beviljas åtkomst; B kan således ge tillfälliga, specifika åtkomstnycklar till A utan att ge dem för mycket kraft. Du kan tänka dig en OAuth-server som chefen i ett stort hotell. Han ger till anställda nycklar som öppnar dörrarna till de rum som de ska komma in i, men varje nyckel är begränsad (det ger inte tillgång till alla rum). Dessutom förstörs nycklarna efter några timmar.

Lite enkelt skillnaden mellan Oauth-OpenID

I viss utsträckning kan auktorisering missbrukas i någon pseudo-autentisering på grund av att om enhet A erhåller från B en åtkomstnyckel via OAuth och visar den till servern S, kan servern S härleda att B verifierade A innan denne beviljades åtkomstnyckeln. Så vissa använder OAuth där de ska använda OpenID.

Jag tycker att denna pseudo-autentisering är mer förvirrande än vad något annat.

OWASP 10

OWASP topp tio är ett kraftfullt medvetenhetsdokument för webbapplikations säkerhet. Det representerar en bred enighet om de mest kritiska säkerhetsriskerna för webbapplikationer. Projektmedlemmar inkluderar en mängd säkerhetsexperter från hela världen som har delat med sig av sin kompetens för att producera denna lista. Vi uppmanar alla företag att anta detta medvetenhetsdokument inom sin organisation och starta processen för att se till att deras webbapplikationer minimerar riskerna.

Vad är OWASP topp tio?

OWASP topp tio ger:

- En lista över 10 mest kritiska säkerhetsapplikationer för webbapplikationer.
- För varje risk det ger:
  - En beskrivning.
  - Exempel attacker.
  - Vägledning om hur man undviker.
  - Referenser till OWASP och andra relaterade resurser.