Fundusze Europejskie
Wiedza Edukacja Rozwój

Politechnika Wrocławska

Unia Europejska
Europejski Fundusz Społeczny

„ZPR PWr – Zintegrowany Program Rozwoju Politechniki Wrocławskiej”

Politechnika Wrocławska

Wydział Informatyki i Zarządzania

Zaawansowane Technologie Webowe

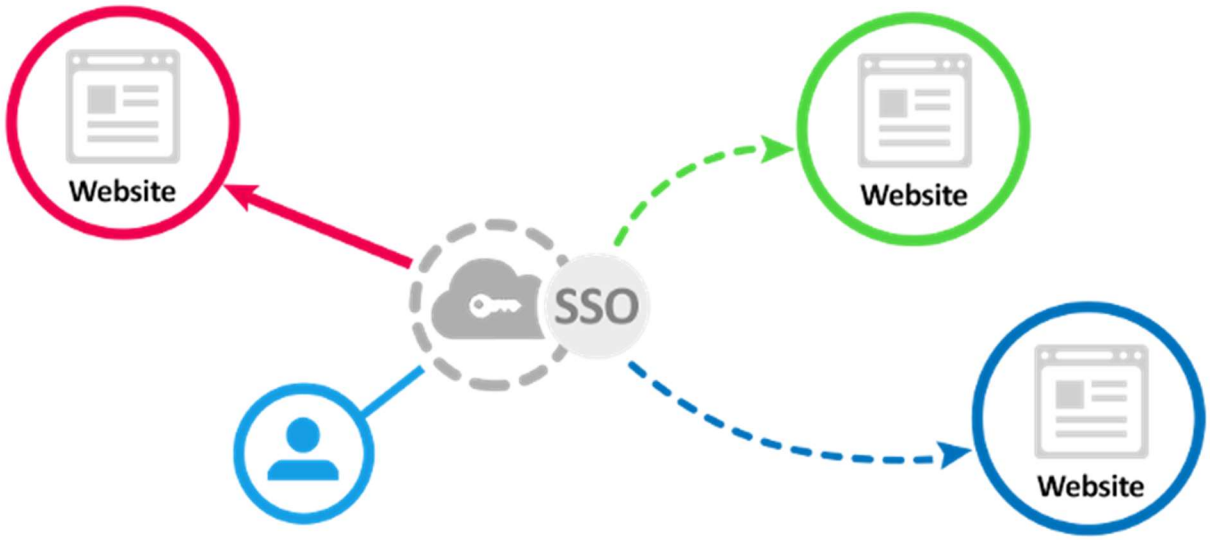
Laboratorium

Temat:	Projekt grupowy – implementacja Single Sign On
Opracował:	mgr inż. Piotr Jóźwiak
Data:	lipiec 2020
Liczba godzin:	2 godziny

Table of Contents

Wstęp

Duże organizacje poszukiwały sposobu na scentralizowanie swoich systemów uwierzytelniania, celem łatwiejszego zarządzania użytkownikami i bezpieczeństwem. Rozwiązaniem tego problemu jest Single Sign On (SSO). SSO jest scentralizowanym serwisem zarządzającym sesją oraz autentykacją, w taki sposób, że pojedynczy zestaw login/hasło może być wykorzystany do uzyskania dostępu do wielu aplikacji. Piękno tego rozwiązania jest ukryte w jego prostocie. Serwis autentykuje użytkownika na dowolnej platformie, a tym samym w sposób przeźroczysty dla użytkownika dostarcza mu dostęp do wszystkich innych serwisów bez ponownego procesu logowania.



Naturalnym rozwinięciem technologii SSO była chęć skorzystania z tego API przez zewnętrznych programistów. Chcieli oni zapenić usługi autoryzacji do własnych serwisów w oparciu o istniejące rozwiązania SSO wdrożone w dużych firmach. Oczywistym jest, że takie podejście staje się sporym wyzwaniem. Do tego wszystkiego pojawiły się różnego rodzaju sieci społecznościowe, które jeszcze bardziej komplikowały sprawę.

Obecnie mamy tysiące aplikacji, które wykorzystują uwierzytelnianie za pośrednictwem sieci społecznościowych jak Facebook, Google, Twitter czy LinkedIn. Największy problem w danej architekturze polega na utrzymaniu prostoty systemu bez narażania poziomu zabezpieczeń. Rozwiązaniem w tej sytuacji okazuje się koncepcja Federated Identities (FI).

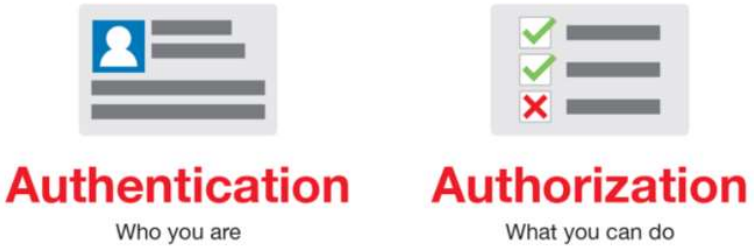
Aktualnie posiadamy trzy główne protokoły FI: SAML, Oauth2 oraz OpenID Connect. Zanim pokrótce się im przyjrzymy, omówmy kilka powszechnych pojęć, które są czesto mylone.

Uwierzytelnianie vs Autoryzacja

Problematyka bezpieczeństwa dostępu do zasobów definiuje dwa terminy: uwierzytelnianie i autoryzacja, które często rozumiane są zamiennie. Uwierzytelnianie to weryfikacja tożsamości, podczas gdy autoryzacja to weryfikacja tego, do czego uzyskuje się dostęp.

Uwierzytelnianie polega na sprawdzaniu danych logowania przed udzieleniem użytkownikowi dostępu do systemu. Zazwyczaj w najprostszym ujęciu jest to formularz logowania monitujący o podanie nazwy użytkownika oraz hasła.

Autoryzacja następuje po procesie uwierzytelniania poprzez weryfikację praw przed udzieleniem dostępu do wymaganych zasobów, takich jak bazy danych, pliki, repozytoria itp. Praktycznym przykładem może być sprawdzenie czy aktualnie zalogowany użytkownik może edytować posty na blogu.



Zarówno autoryzacja, jak i uwierzytelnianie mają funamentalne znaczenie dla bezpieczeństwa i zarządzania dostępem. Te dwie koncepcje, mają kluczowe znaczenie dla bezpieczeństwa infrastruktury systemu, a ich zrozumienie jest niezbędne do zarządzania tożsamością i dostępem.

SAML

Security Assertion Markup Language (SAML) to oparty na języku XML otwarty standard używany do implementacji logowania Single Sign On (SSO). SAML 2.0 został opracowany w 2005 roku i jest aktualną wersją tego standardu.

SAML jest używany zarówno do uwierzytelniania, jak i autoryzacji między dwiema stronami: dostawcą usług (Office365, Salesforce, G Suite itp.) i dostawcą tożsamości (Okta, OneLogin, Ping Identity itp.). Z założenia Service Provider (SP) ufa dostawcy tożsamości (Identity Provider- IdP) w procesie uwierzytelniania. Odbywa się to za pomocą dokumentu SAML XML wysłanego przez IdP, zawierającego autoryzację i uwierzytelnienie użytkownika, a następnie przekierowanie do usługodawcy SP.

Przyjrzyjmy się procesowi uwierzytelniania przedstawionemu na poniższym diagramie. Jako Identity Providera (IdP) możemy przyjąć serwis Okta. Natomiast jako Service Provider (SP) występuje aplikacja webowa – np. nasza z projektu.

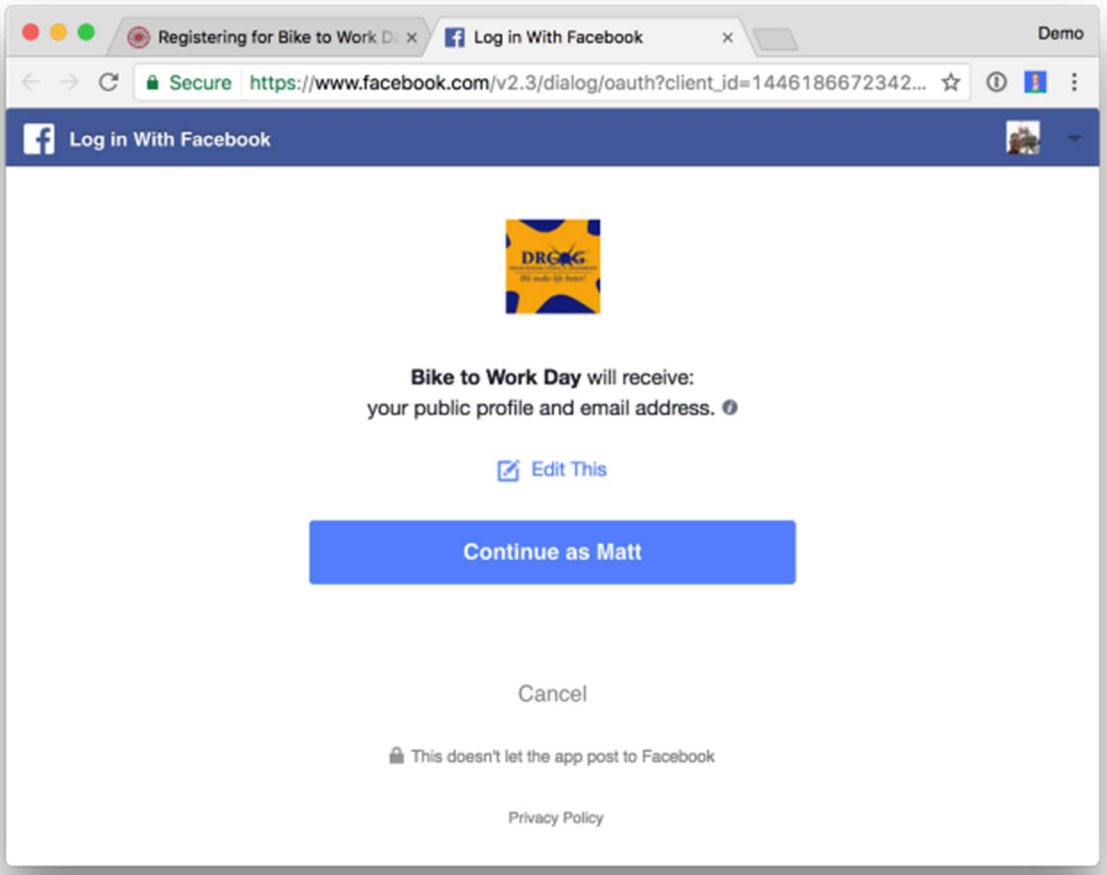
1. Użytkownik próbuje zalogować się do aplikacji webowej z przeglądarki.
2. Aplikacja webowa odpowiada, generując żądanie SAML.
3. Przeglądarka przekierowuje użytkownika na adres URL do logowania jednokrotnego, Okta analizuje żądanie SAML, uwierzytelnia użytkownika (może to być za pomocą nazwy użytkownika i hasła, uwierzytelniania dwuskładnikowego lub MFA, jeśli użytkownik nie znajduje się w wewnętrznej sieci firmy; jeśli użytkownik jest już uwierzytelniony w Okta, ten krok zostanie pominięty) i generuje odpowiedź SAML.
4. Okta ponownie wysyła zakodowaną odpowiedź SAML do przeglądarki klienta.
5. Przeglądarka przekierowuje odpowiedź SAML do aplikacji webowej.
6. Jeśli weryfikacja się powiedzie, użytkownik zostanie zalogowany do aplikacji webowej i uzyska dostęp do zasobów.

OAuth2

OAuth2 to otwarty standard używany do autoryzacji, który umożliwia aplikacjom dostarczanie funkcjonalności z „delegowaną autoryzacją”. W przeciwieństwie do innych struktur, które zapewniają uwierzytelnianie, OAuth2 autoryzuje tylko urządzenia, API, serwery z tokenami dostępu, a nie poświadczeniami. Działa przez HTTPS.

Technologia OAuth2 powszechnie wykorzystywana jest w takich serwisach jak Facebook czy Google. Serwisy te umożliwiają skorzystanie z ich wewnętrznych mechanizmów autoryzacji do logowania w zewnętrznych serwisach, nie związanych z dostawcą tożsamości.

Zwróćmy uwagę na poniższy przykład. Przykładowa strona z pierwszej zakładki korzysta z Facebookowego mechanizmu logowania, widocznego na drugiej zakładce.

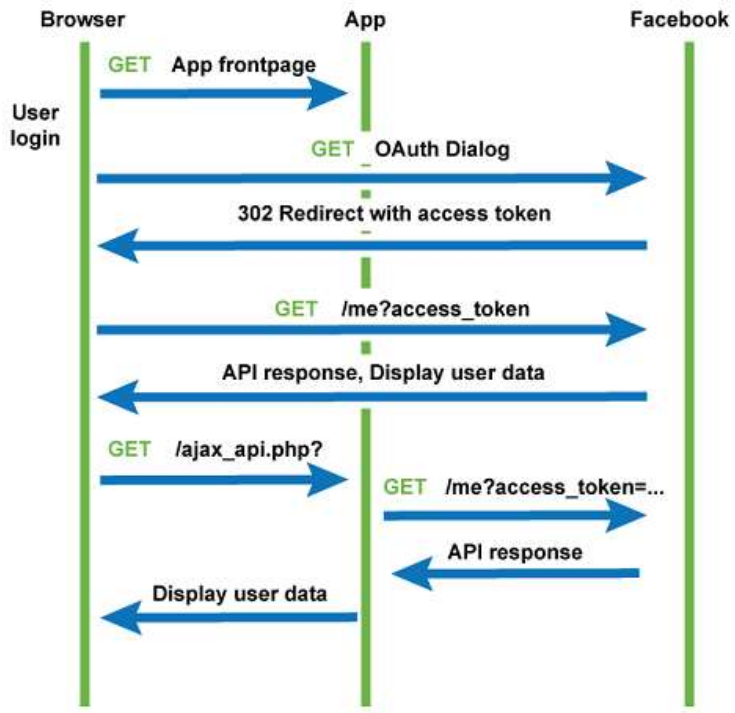


Przypomina to system kart hotelowych, ale dla aplikacji. Jeśli dysponujemy hotelową kartą-kluczem, możemy uzyskać dostęp do pokoju. Jak zdobyć kartę hotelową? Aby ją otrzymać, musimy przeprowadzić proces uwierzytelniania w recepcji. Po uwierzytelnieniu i uzyskaniu karty-klucza możemy uzyskać dostęp do zasobów w hotelu.

Oauth2 wyróżnia cztery role:

- Właściciel zasobu: ogólnie sam użytkownik.
- Klient: aplikacja żądająca dostępu do serwera zasobów.
- Serwer zasobów: serwer obsługujący chronione dane (na przykład Facebook hostujący profil i dane osobowe).
- Serwer autoryzacyjny: serwer wystawiający klientowi token dostępu. Ten token zostanie użyty, aby uzyskać dostęp do żądanych zasobów.

Poniższy diagram przedstawia proces uwierzytelniania protokołem Oauth2:



1. Spotify chce uzyskać dostęp do listy znajomych z Twojego konta na Facebooku.
2. Spotify przekierowuje Cię na serwer autoryzacyjny (w tym przypadku Facebook).
3. Jeśli autoryzujemy dostęp, serwer autoryzacyjny wysyła kod autoryzacyjny do klienta (Spotify) w odpowiedzi zwrotnej.
4. Następnie ten kod jest wymieniany na token dostępu między Facebookiem a Spotify.
5. Teraz Spotify może używać tego tokena dostępu do wysyłania zapytań do serwera zasobów (Facebook) i pobierania listy znajomych.

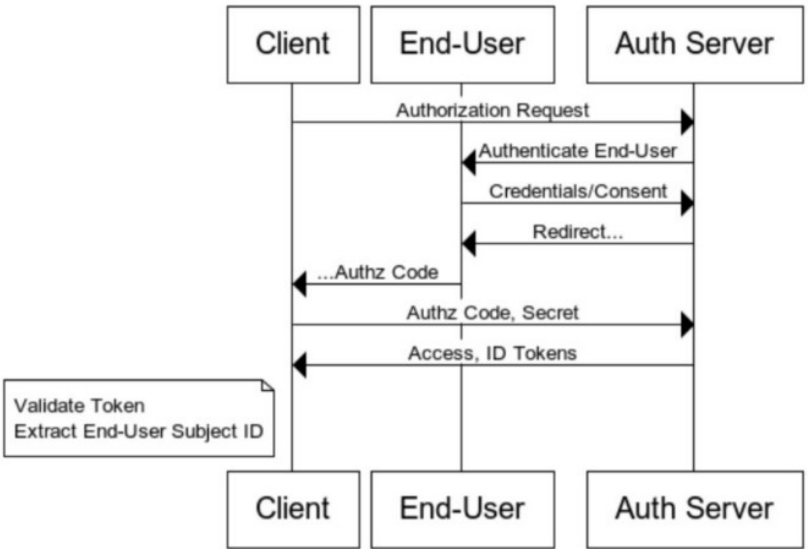
Należy zwrócić uwagę na to, że użytkownik nigdy nie zobaczy tokenu dostępu, będzie on przechowywany w sesji. Serwer autoryzacji wysyła również inne informacje, takie jak okres ważności tokenu i token odświeżania.

OpenID Connect

OpenID Connect to warstwa tożsamości uzupełniająca protokół OAuth 2.0, która rozszerza OAuth2 i umożliwia „uwierzytelnianie federacyjne” (Federated Authentication).

Przebieg procesu OpenID Connect jest podobny do przepływu autoryzacji OAuth2, a główną różnicą jest „token id”, który umożliwia uwierzytelnienie użytkownika.

Należy pamiętać, że uwierzytelnianie federacyjne to coś zupełnie innego niż autoryzacja delegowana. Przyjrzyjmy się ponownie przykładowi Facebooka i Spotify.



- Uwierzytelnianie federacyjne polega na logowaniu do Spotify przy użyciu danych logowania z Facebooka.
- Autoryzacja delegowana to możliwość uzyskania dostępu do zasobów przez aplikację zewnętrzną. W tym przypadku Spotify próbuje uzyskać dostęp do listy znajomych na Facebooku, aby zaimportować ją do Spotify.

Wszystkie powyższe informacje można podsumować w poniższej tabeli:

	SAML 2.0	OAuth2	OpenID Connect
Czym jest?	Otwarty standard do autoryzacji oraz uwierzytelniania	Otwarty standard do autoryzacji	Otwarty standard do uwierzytelniania
Twórca	Opracowany przez OASIS w 2001 roku	Opracowany przez Twitter oraz Google w 2006 roku	Opracowany przez OpenID Foundation w 2014 roku
Docelowy odbiorca	SSO dla aplikacji korporacyjnych	API autoryzacyjne	SSO dla aplikacji konsumenckich
Format komunikacji	XML	JSON	JSON

Przykład wykorzystania SSO

Aby rozpocząć prace deweloperskie polegające na zbudowaniu SSO w aplikacji webowej można skorzystać z wielu gotowych rozwiązań. Do zbudowania uwierzytelniania w oparciu o SAML warto użyć aplikacji napisanej w PHP o nazwie SimpleSAMLphp (https://simplesamlphp.org/). Jest to prosty skrypt dostarczający zarówno funkcjonalności Service Providera (SP) oraz Identity Providera (IdP). Najlepiej pierwsze kroki rozpocząć od tego narzędzia. Sam przykład implementacji jest mocno powiązany z wykorzystaną technologią po stronie klienta, tak więc nie da się tutaj przedstawić kompletnego przykładu wykonania tej funkcjonalności. Zatem należy przeprowadzić własne poszukiwania w celu przygotowania aplikacji na SSO z SAML. Dobrym zestawieniem możliwości zestawienia SAMLa w wielu różnych

technologiach jest artykuł dostępny pod tym adresem: https://medium.com/the-new-control-plane/i-need-a-saml-stack-now-63d9691e2d43

W przypadku standardu OAuth2 oraz Angulara warto spojrzeć na ten materiał: https://www.baeldung.com/rest-api-spring-oauth2-angular. Implementacja OAuth2 może nie być taka prosta. Więcej szczegółów na temat OAuth2 można odnaleźć w tym artykule: https://sekurak.pl/oauth-2-0-jak-dziala-jak-testowac-problemy-bezpieczenstwa/.

Implementacja SSO w projekcie grupowym

Kolejnym zadaniem do wykonania przez grupy projektowe jest implementacja dowolnego sposobu funkcjonalności Single Sign On. Można skorzystać z dowolnej technologii SSO (SAML, OAuth2). Grupie projektowej pozostawia się do wyboru, czy logowanie odbędzie się za pośrednictwem zewnętrznego serwisu ogólnie dostępnego jak Facebook, Google, etc., czy zaimplementowany zostanie własny serwer uwierzytelniania i autoryzacji.