STUDIA INFORMATICA 2009
Volume 30 Number 2B (84)

Marcin GORAWSKI, Michał FLASIŃSKI
Politechnika Śląska, Instytut Informatyki

CYFROWA TOŻSAMOŚĆ W INTERNECIE NA BAZIE OTWARTEGO PROTOKOŁU OPENID

Streszczenie. W artykule przedstawiono otwarty, zdecentralizowany protokół OpenID. Opisane zostały problemy, jakie protokół rozwiązuje, terminologię, specyfikacje, a także cechy OpenID. Przybliżono w dosyć szczegółowy sposób proces uwierzytelniania użytkowników. W dalszej części opisano implementację dostawcy OpenID, problem bezpieczeństwa, a zakończono rozważaniami na temat przyszłości aplikacji i samego protokołu.

Słowa kluczowe: OpenID, uwierzytelnianie, cyfrowa tożsamość, aplikacja webowa, bezpieczeństwo

DIGITAL IDENTITY IN INTERNET BASING ON OPENID PROTOCOL

Summary. The article presents open, decentralized OpenID protocol. It describes problems which are solved by the protocol, terminology, specifications as well as features of OpenID. The article introduces in details the user authentication process. Further OpenID provider, implementation and security problem are described. The summary contains considerations concerning the future of the application and the protocol.

Keywords: OpenID, authentication, digital identity, web application, security

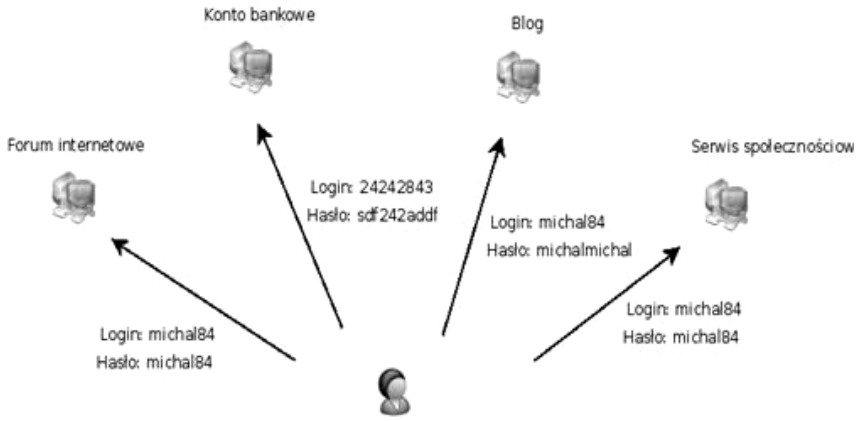
1. Wprowadzenie

Zarządzanie tożsamością użytkowników jest podstawowym mechanizmem kontroli dostępu do systemów informatycznych. Każdy użytkownik Internetu, chcący mieć dostęp do konta bankowego, skrzynki e-mail czy portalu społecznościowego, musi posiadać własną, cyfrową tożsamość i ją potwierdzić. Zarządzać tożsamością można na poziomie systemu operacyjnego, w przypadku systemów Unix/Linux są to LDAP, Kerberos, Radius, a w przypadku

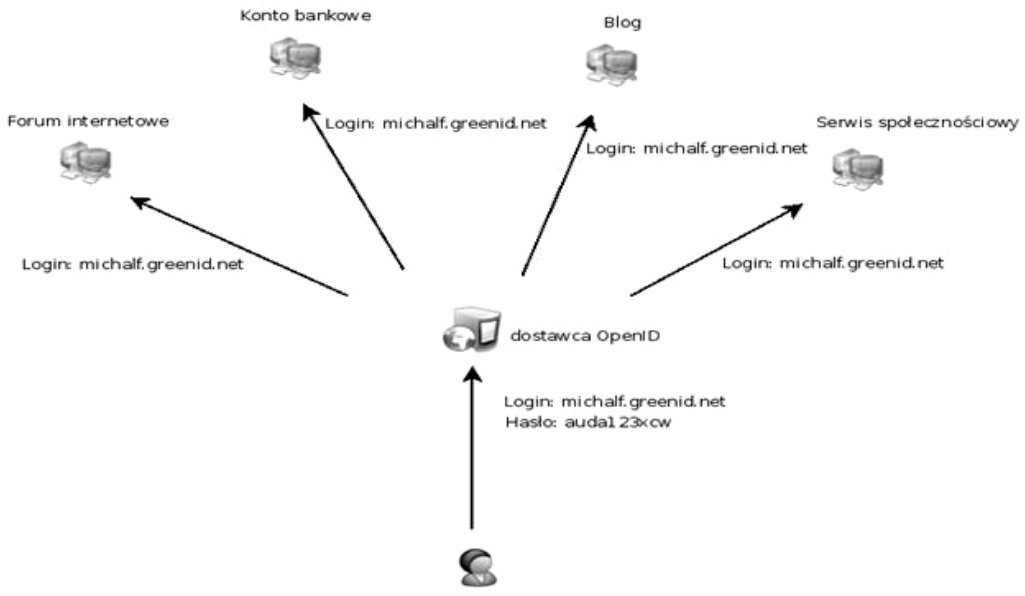
Microsoft Windows – Active Directory. Mechanizmy te sprawdzają się dobrze w zamkniętych systemach, w obrębie jednej organizacji czy korporacji.

Zupełnie inna jest sytuacja w przypadku aplikacji webowych, do których jest dostęp globalny. I w tym przypadku najbardziej typowym mechanizmem uwierzytelniania jest weryfikacja identyfikatora użytkownika i hasła. Użytkownik musi więc posiadać osobny identyfikator i hasło dla każdej aplikacji. Dostęp do aplikacji może być utrudniony bądź niemożliwy w przypadku zapomnienia przez użytkownika danych uwierzytelniających.

Zarządzanie i zapamiętywanie tych danych spada najczęściej na użytkownika – który zwykle korzysta z tego samego lub podobnego loginu i hasła, lub je jawnie zapisuje. Wiąże się to ze spadkiem bezpieczeństwa użytkownika, jego dane mogą zostać przejęte przez intruza i wykorzystane.



Rys. 1. Uwierzytelnianie w aplikacjach webowych - stan obecny
Fig. 1. Authentication in web applications – present state



Rys. 2. Uwierzytelnianie w aplikacjach webowych przy użyciu identyfikatora OpenID
Fig. 2. Authentication in web applications using OpenID identifier

Odpowiedzią na te problemy jest OpenID – otwarty protokół pozwalający na używanie URL-a jako własnego identyfikatora i wykorzystywanie go we wszystkich aplikacjach, które wspierają ten protokół.

2. Protokół OpenID

OpenID jest otwartym, zdecentralizowanym protokołem służącym do uwierzytelniania użytkowników. Głównym założeniem protokołu jest możliwość uwierzytelniania użytkowników we wszystkich witrynach za pomocą jednego loginu (będącego np.URL-em) i hasła. OpenID powstał w 2005 roku, a ostatnia wersja jego specyfikacji, oznaczona numerem 2.0, została sfinalizowana w grudniu 2007.

Do najważniejszych pojęć występujących w protokole OpenID należą [1, 2, 3]:

- Login, identyfikator (ang. Identifier) – jest nim URL, URI lub XRI należący do użytkownika, za pomocą którego uwierzytelnia się on w aplikacji zależnej.
- Aplikacja webowa, aplikacja zależna (ang. Relying Party, Consumer) – aplikacja, do której dostęp próbuje uzyskać użytkownik. Użytkownik uzyskuje dostęp do niej po zakończonym sukcesem uwierzytelnieniu za pomocą identyfikatora OpenID.
- Dostawca OpenID (ang. OpenID Provider) – serwer, przy pomocy którego użytkownik uwierzytelnia się i zarządza własną tożsamością.
- Profil (ang. persona) – zbiór informacji o użytkowniku. Użytkownik może mieć wiele profili i każdy profil składa się z wielu atrybutów.
- Atrybut (ang. attribute) – podstawowa jednostka informacji wymieniana między dostawcą OpenID a aplikacją zależną w przypadku rozszerzeń protokołu OpenID.

2.1. Cechy protokołu OpenID

Zaletami OpenID są:

- Otwartość – OpenID powstało dzięki społeczności Open Source, nie jest przez nikogo kontrolowane i całkowicie darmowe.
- Eliminowanie konieczności posiadania loginów i haseł dla każdej witryny – wystarczy jeden login OpenID i hasło, aby móc uwierzytelniać się we wszystkich aplikacjach wspierających OpenID.
- Logowanie do aplikacji webowej bez podawania loginu i hasła. Hasło zawsze podawane jest po stronie dostawcy OpenID. Możliwe jest podawanie aplikacji tylko adresu dostawcy OpenID, a nie dokładnego identyfikatora użytkownika.

- Możliwość wyboru przez użytkownika informacji jakie przekaże aplikacji webowej – użytkownik może się zgodzić na przekazanie wybranych danych lub odmówić ich udostępnienia.
- Mniejszy koszt zarządzania tożsamością po stronie aplikacji webowej – wszystkie informacje o użytkowniku przechowywane są po stronie dostawcy OpenID.
- Decentralizacja – nie istnieje żaden centralny rejestr użytkowników ani haseł. Każdy może postawić swój własny serwer OpenID, a adres serwera jest jednocześnie częścią loginu OpenID.
- Łatwość aktualizacji – dane przechowywane są w jednym miejscu.

OpenID nie jest pozbawione wad, tj.:

- Zbyt duża koncentracja danych – w przypadku włamania na serwer dostawcy OpenID, łupem złodziei mogą paść bardzo ważne dane (np. numery kart kredytowych).
- Kradzież tożsamości – oznacza przejęcie kontroli włamywacza nad tożsamością ofiary, uzyskując dostęp do wszystkich jej aplikacji.
- Phishing (spoofing) – wyłudzanie poufnych informacji osobistych przez podszywanie się pod godną zaufania osobę lub instytucję poprzez fakt, że logowanie odbywa się na innej stronie niż aplikacja, do której użytkownik chce uzyskać dostęp.

2.2. Specyfikacje OpenID

OpenID jest protokołem składającym się z kilku specyfikacji, które mogą być implementowane wszystkie razem lub każda osobno. Istnieje jednak specyfikacja bazowa – OpenID Authentication (aktualnie w wersji 2.0), która stanowi podstawę dla pozostałych. Ponieważ OpenID jest relatywnie nową technologią, nie wszystkie specyfikacje doczekały się wersji finalnych. Część z nich znajduje się w fazie dopracowywania.

W celu wymiany informacji między dostawcą OpenID a aplikacją zależną i vice versa wysyłane są wiadomości, które muszą spełniać następujące reguły:

- wiadomości wysyłane są jako zwykły tekst, z pełnym wsparciem dla Unicode,
- zawierają pary w postaci klucz-wartość,
- wszystkie klucze muszą rozpoczynać się sekwencją „openid.”,
- wiadomości mogą być wysyłane zarówno za pomocą żądań POST, jak i GET,
- wszystkie wiadomości OpenID muszą zawierać co najmniej 2 pola:
  - openid.ns – wartość to „http://specs.openid.net/auth/2.0”, jeżeli będzie to „http://openid.net/signon/1.1” lub „http://openid.net/signon/1.0” to oznacza, że została użyta wcześniejsza wersja protokołu,
  - openid.mode – wartość różna, w zależności od rodzaju wiadomości.

2.3. Przykładowy scenariusz uwierzytelniania

Inicjalizacja. Użytkownik inicjalizuje uwierzytelnianie przez wpisanie loginu OpenID w oknie aplikacji webowej korzystając z przeglądarki stron WWW i przyciska przycisk zatwierdzający formularz. Loginem jest URL, np: http://jankowalski.mojserwer.pl. Użytkownik musi być w posiadaniu tego identyfikatora, tzn. wcześniej musi założyć konto u dostawcy OpenID. Użytkownik ma pełne prawo do wyboru dostawcy OpenID i powinien wybrać tego, któremu najbardziej ufa. Celem protokołu jest zweryfikowanie, czy użytkownik jest tym, za kogo się podaje. W żadnym wypadku po stronie aplikacji zależnej użytkownik nie powinien podawać swojego hasła ani aplikacja tego wymagać. Po zatwierdzeniu formularza rozpoczyna się właściwa część uwierzytelniania.

Normalizacja identyfikatora wpisanego przez użytkownika. Użytkownik może dostarczyć login OpenID w formie URI bądź XRI (tab. 1). Proces normalizacji jest dosyć skomplikowany i rządzą nim następujące reguły:

- Jeżeli identyfikator zaczyna się od xri://, xri://$ip, lub xri://$dns* - wytnij je.
- Jeśli pierwszym znakiem jest któryś z symboli (=, @, +, $, !), wtedy identyfikator jest znormalizowanym identyfikatorem XRI. W innym przypadku identyfikator jest traktowany jako HTTP URL (jeśli prefiks http/https jest nieobecny, należy dodać http://). Aplikacja webowa musi podążać za tym URL-em i wszystkimi przekserowaniami, dopóki nie otrzyma dokumentu. Końcowy URL jest znormalizowanym identyfikatorem URL.

Tabela 1

Przykłady normalizacji identyfikatorów OpenID

Wpisany identyfikator	**Znormalizowany identyfikator**	**Typ identyfikatora**
=mojserwer	=mojserwer	XRI
mojserwer.pl	http://mojserwer.pl	URL
https://mojserwer.pl	https://mojserwer.pl	URL
xri://=mojserwer	=mojserwer	XRI
xri://$ip*192.168.17.20	http://192.168.17.20	URL

Wyszukiwanie adresu dostawcy OpenID. Znormalizowany identyfikator jest następnie poddawany procesowi wyszukania, który ma na celu poznanie informacji służących do wysłania żądania uwierzytelniania. Wyszukiwanie może zostać przeprowadzone na różne sposoby w zależności od tego, jaki identyfikator został dostarczony przez użytkownika:

- jeżeli typem identyfikatora jest XRI, przeprowadzone jest wyszukiwanie za pomocą tego identyfikatora, a otrzymanym dokumentem jest XRDS,

- jeżeli typem identyfikatora jest URL, może zostać wykorzystany protokół Yadis, wtedy otrzymany zostanie dokument XRDS,
- jeżeli typem identyfikatora jest URL, może zostać przeprowadzone wyszukiwanie za pomocą dokumentu HTML (taka sama sytuacja może nastąpić jeżeli protokół Yadis zawiódł). Wtedy otrzymany zostanie dokument HTML.

Dokument XRDS jest niczym innym jak dokumentem XML opisującym serwery OpenID mogące służyć do uwierzytelnienia użytkownika. Przykładowy dokument XRDS wygląda następująco:

```
<?xml version="1.0" encoding="UTF-8"
<xrds:XRDS xmlns:xrds="xri://$xrds" xmlns:openid="http://openid.net/xmlns/1.0"
xmlns="xri://$xrd*($v*2.0)">
<XRD>
    <Service priority="5">
        <Type>http://specs.openid.net/auth/2.0/server</Type>
        <Type>http://openid.net/srv/ax/1.0</Type>
        <Type>http://openid.net/sreg/1.0</Type>
        <URI>http://greenid.net/openid_login</URI>
    </Service>
    </XRD>
</xrds:XRDS>
```

Dokument ten wskazuje na jeden punkt końcowy dostawcy OpenID (wartość węzła <URI>), pod który aplikacja webowa powinna przekierować użytkownika. W dokumencie dostawca OpenID informuje aplikację webową, że wspiera specyfikację OpenID Authentication w wersji 2.0 a także rozszerzenia OpenID Simple Registration i OpenID Attribute Exchange (wartości węzłów <Type>). Dostarczony przez użytkownika login może być również identyfikatorem serwera dostawcy, wtedy wybranie właściwego loginu następuje dopiero po przekierowaniu do dostawcy OpenID.

Krok opcjonalny: ustanowienie asocjacji. Aplikacja webowa i dostawca tożsamości przeprowadzają asocjację, polegającą na wymianie kluczy generowanych algorytmem Diffie-Hellmana. Dostawca tożsamości używa klucza do podpisywania dalszych wiadomości, które weryfikuje aplikacja webowa. To eliminuje konieczność późniejszych bezpośrednich żądań w celu weryfikacji sygnatury każdego żądania lub odpowiedzi.

Proces ustanawiania asocjacji (opcjonalny, ale bardzo rekomendowany) zwiększa bezpieczeństwo wymiany danych między dostawcą OpenID a aplikacją zależną. Rezultatem procesu jest uchwyt asocjacji, który następnie jest przekazywany przy wymianie kolejnych wiadomości między dostawcą OpenID a aplikacją zależną.

Przekierowanie do dostawcy OpenID z żądaniem uwierzytelnienia. Aplikacja zależna prosi serwer dostawcy OpenID o potwierdzenie, że użytkownik jest w posiadaniu identyfikatora, a więc jest tym, za kogo się podaje. W żądaniu uwierzytelnienia przekazywane są następujące parametry (tabela 2):

Tabela 2

Parametry wysyłane w żądaniu uwierzytelnienia użytkownika

Parametr żądania	**Wartości**
openid.ns	http://specs.openid.net/auth/2.0
openid.mode	"checkid_setup" - pozwala użytkownikowi na interakcje po stronie dostawcy OpenID „checkid_immediate” - nie pozwala użytkownikowi na jakąkolwiek interakcję
openid.claimed_id	(opcjonalny) "openid.claimed_id" i "openid.identity" powinny być obecne razem albo żadne z nich. Jeśli nie ma ich, to znaczy, że żądanie nie dotyczy uwierzytelniania, ale jest żądaniem jednego z rozszerzeń specyfikacji.
openid.identity	(opcjonalny) lokalny identyfikator użytkownika po stronie dostawcy OpenID.
openid.assoc_handle	(opcjonalny) uchwyt jednoznacznie identyfikujący zawartą asocjację. Jeżeli nie jest obecny, wtedy proces przebiega w trybie bezstanowym.
openid.return_to	(opcjonalny) URL, pod który dostawca powinien przekierować użytkownika po wysłaniu odpowiedzi na to żądanie.
openid.realm	(opcjonalny) wzorzec URL-a, o który dostawca powinien zapytać użytkownika w celu potwierdzenia, że użytkownik rzeczywiście chce udzielić zgody na uwierzytelnienie.

Weryfikacja tożsamości użytkownika po stronie dostawcy OpenID. Specyfikacja nie definiuje samego procesu weryfikacji danych użytkownika ani rzeczy z tym związanych. Najczęściej weryfikacja polega na przeprowadzeniu interakcji z użytkownikiem: wpisaniu przez użytkownika loginu i hasła, a następnie odpowiedzenie na pytania o zgodę na uwierzytelnienie, o wybranie danych wymienianych z aplikacją, jeśli o to prosi i na końcu o potwierdzenie swojego wyboru.

Przekierowanie użytkownika do aplikacji zależnej. Dostawca OpenID decyduje (na podstawie informacji przekazanych przez użytkownika), czy proces uwierzytelnienia zakończył się pomyślnie i tę informację przekazuje aplikacji zależnej. Serwer dostawcy musi zbudować wiadomość składającą się z pól jak w tabeli 3.

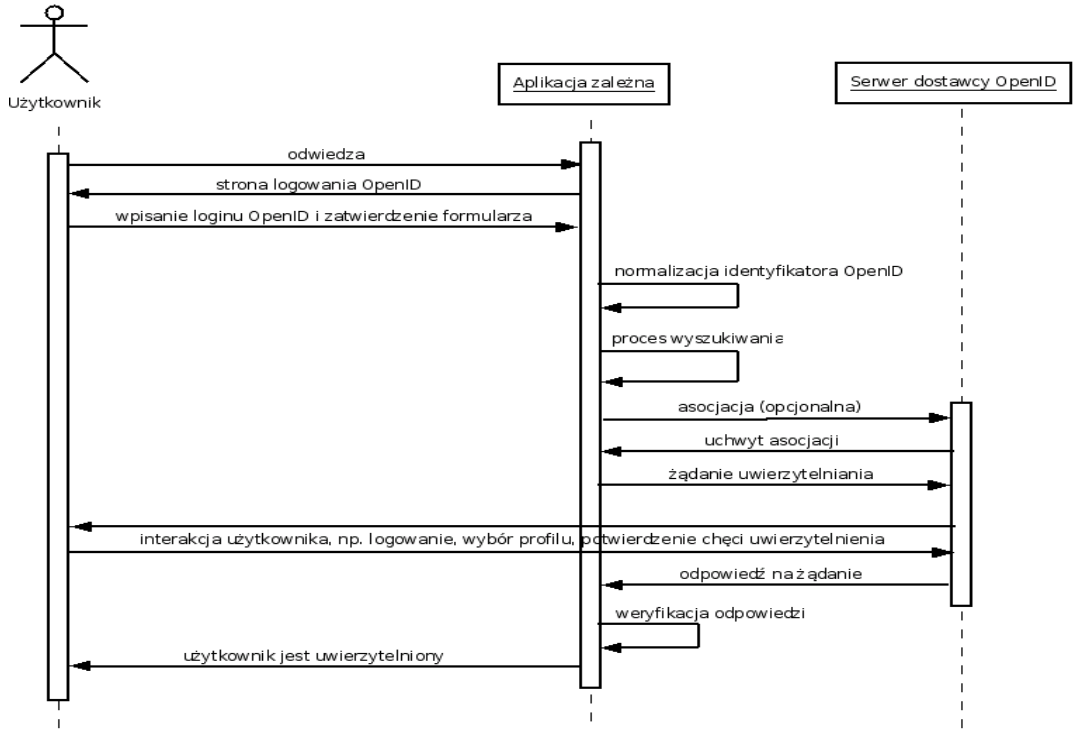
Wiadomość jest wysyłana jako przekierowanie do strony aplikacji zależnej (do URL-a podanego jako parametr „openid.return_to” w żądaniu uwierzytelniania).

Aplikacja zależna weryfikuje informacje otrzymane od dostawcy OpenID, została przedstawiona na rys. 3.

Tabela 3

Parametry przesyłane przez dostawcę OpenID do aplikacji zależnej

Nazwa pola	**Opis, możliwe wartości**
openid.ns	http://specs.openid.net/auth/2.0
openid.mode	„id_res” - odpowiedź na żądanie uwierzytelniania
openid.op_endpoint	końcowy URL dostawcy OpenID
openid.claimed_id	(opcjonalny) identyfikator użytkownika
openid.identity	(opcjonalny) lokalny identyfikator OpenID po stronie dostawcy
openid.return_to	kopia URL wysłanego w żądaniu
openid.response_nonce	unikalna wartość jednoznacznie identyfikująca to żądanie. Musi zaczynać się od aktualnego czasu na serwerze i może zawierać kolejne znaki ASCII.
openid.invalidate_handle	(opcjonalny) w przypadku niepoprawnego uchwytu asocjacji zawiera jego kopię
openid.assoc_handle	uchwyt asocjacji
openid.signed	oddzielona przecinkiem lista podpisanych pól
openid.sig	sygnatura wiadomości



Rys. 3. Przepływ zdarzeń przykładowego procesu uwierzytelniania OpenID
Fig. 3. Event flow of exemplary OpenID authentication process

Końcowym etapem uwierzytelniania jest weryfikacja-sprawdzenie informacji wymienianych z dostawcą OpenID:

- Czy wartość parametru „openid.return_to" pasuje do URL-a z aktualnego żądania. Aplikacja upewnia się, że użytkownik został przekierowany pod właściwy URL.
- Czy informacje uzyskane w procesie wyszukiwania pasują do tych z wiadomości wysłanej przez dostawcę.
- Czy wiadomość z tą samą wartością parametru „openid.response_nonce" nie została jeszcze zaakceptowana – to zabezpiecza przed atakami typu „reply attacks".
- Czy sygnatura jest prawidłowa i czy wszystkie pola zawarte w parametrze „openid. signed" zostały podpisane.

3. Bezpieczeństwo

OpenID może być podatny na różne metody ataku.

Podsłuchiwanie. W czasie procesu uwierzytelniania istnieje tylko jeden moment podatny na ten rodzaj ataku: jeżeli identyfikator wygenerowany przez dostawcę OpenID (ang. *nonce*) nie został zweryfikowany, podsłuchiwacz może przejąć wiadomość: openid.response_ nonce = 2008-07-01T13:53:17Z0. Identyfikator ten przesyłany jest jako jeden z parametrów odpowiedzi na żądanie uwierzytelniania. Ten rodzaj ataku jest eliminowany przez szyfrowanie warstwy transportowej (ang. Transport Layer Security). Innym rozwiązaniem jest weryfikacja identyfikatora w czasie weryfikacji wiadomości. W tym drugim przypadku wiadomość nie może być po raz drugi zostać użyta do uwierzytelnienia. Wysłanie po raz drugi odpowiedzi przez dostawcę OpenID, z tym samym identyfikatorem, musi spowodować jej odrzucenie i upadek procesu uwierzytelniania. Odrzucona powinna być również wiadomość, której parametr „openid.response_nonce" jest nieaktualny. Aplikacja webowa powinna odrzucać wiadomości wysłane wcześniej niż np. 180 sekund temu.

Atak typu „man in the middle". Sposobem na zapobieganie tego typu atakom jest użycie silnej metody podpisywania wiadomości. Podpisywanie zapobiega zmianie wartości podpisywanych pól – ich zmiana wymagałaby złamania klucza służącego do podpisywania. Poniżej przedstawiona jest przykładowa lista pól podpisanych i sygnatura wiadomości:

```
openid.signed=op_endpoint,claimed_id,identity,return_to,response_nonce,assoc_han
dle,sreg.email,sreg.gender,sreg.postcode,sreg.timezone,sreg.fullname,sreg.dob,
openid.sig = 0Ou6h7z9UIPxTj0LP3hy/NxivgGjpZ1VjX5HQL7Gr60=
```

Skuteczność zależy więc od losowości klucza i niemożliwości jego odgadnięcia. Atakujący może się wcielić w rolę dostawcy OpenID i ustanowić własną funkcję asocjacji bądź też działać w trybie bezstanowym. Jednym ze sposobów ochrony przed tego typu

atakiem jest cyfrowe podpisywanie dokumentów XRDS. Po upewnieniu się co do prawdziwości certyfikatu, oszustwo nie jest możliwe. Wcielenie się w rolę serwera dostawcy OpenID wymagałoby bowiem podrobienia certyfikatu, co jest praktycznie niewykonalne. Stosowanie certyfikatów SSL jest zalecane, ale nie konieczne. Dostawca powinien używać certyfikatu podpisanego przez uznany ośrodek certyfikujący.

Nieuczciwe przekierowania przez aplikację zależną (phihing). Może się tak zdarzyć, że „fałszywa" aplikacja zależna przekieruje użytkownika do podrobionej witryny dostawcy OpenID. Aplikacja zależna powinna przekierować użytkownika do strony dostawcy OpenID natychmiast po podaniu przez niego loginu i zatwierdzenia formularza na tej samej stronie, z widocznymi wszystkimi kontrolkami. Podanie przez użytkownika własnego identyfikatora i hasła na takiej podrobionej stronie może skończyć się przejęciem identyfikatora i wykradzeniem prywatnych danych użytkownika.

Specyfikacja nie definiuje metody zapobiegania tego typu atakom. Dostawcy OpenID stosują np. prywatne obrazki, które widoczne są przez użytkownika po przekierowaniu do ich strony. Jeżeli użytkownik widzi taki obrazek, to jest większa pewność, że przekierowanie było właściwe. Innym sposobem jest edukowanie użytkowników o tego typu procederze i oferowanie dodatkowego wsparcia, np. w postaci wtyczek do przeglądarki internetowej. Dobrym rozwiązaniem na przyszłość jest zaimplementowanie rozszerzenia OpenID Provider Authentication Policy, obecnie zapowiadana.

Przeglądarka internetowa. Może być zarażona programami typu spyware lub malwa re, które mogą przejąć identyfikator użytkownika i hasło. Tutaj niestety czujność musi zachować użytkownik. Podobny efekt można uzyskać za pomocą ataku „cross-site-scripting". Rozwiązaniem jest wyłączenie skryptów po stronie dostawcy OpenID.

Zamiana HTTP na HTTPS. Rekomendowaną metodą jest zamiana URL-a rozpoczynającego się od HTTP na HTTPS. W tym czasie następuje przekierowanie z jednego adresu na drugi i aplikacja zależna korzysta z URL-a rozpoczynającego się od HTTPS, jako identyfikatora użytkownika. Zamiast tego „fałszywa" aplikacja może przekierować użytkownika na podrobioną stronę. Zalecane jest, aby użytkownik podawał identyfikator rozpoczynający się od HTTPS dla uniknięcia przekierowania.

Atak typu „denial of service". „Fałszywa" aplikacja webowa może bombardować dostawcę OpenID serią wiadomości z żądaniem ustanowienia asocjacji, uwierzytelnienia bądź weryfikacji. To może doprowadzić do destabilizacji dostawcy OpenID. Zalecane jest, aby w tego typu sytuacjach dostawca odmawiał tego typu żądaniom, bazując na wartościach „openid.return_to" i „openid.realm". Adres atakującego trafia na "czarną listę" i żądania przez niego kierowane nie są obsługiwane.

4. Implementacja prototypowego dostawcy OpenID

W ramach prac własnych zbudowano prototypową aplikację pełniącą rolę serwera dostawcy OpenID, wspierającego najnowszą wersję protokołu, jak również rozszerzenia OpenID Simple Registration i OpenID Attribute Exchange [2, 3]. Prototypowa aplikacja dostawcy OpenID ma następujące funkcje:

- implementacja specyfikacji OpenID Authentication w wersji 2.0,
- rozszerzenie OpenID Attribute Exchange,
- rozszerzenie OpenID Simple Registration,
- tworzenie kont użytkowników,
- uwierzytelnianie i wylogowywanie użytkowników,
- tworzenie, usuwanie i modyfikacja profili użytkownika, możliwość wyboru profilu domyślnego,
- możliwość edycji ustawień konta,
- zarządzanie zaufanymi witrynami (usuwanie, zmiana ustawień),
- prowadzenie historii aktywności konta i możliwość jej przeglądania,
- katalog stron oferujących logowanie za pomocą OpenID,
- możliwość wysyłania wiadomości,
- administracja kontami użytkowników, witrynami i wiadomościami.

Do budowy aplikacji została użyta technologia Java Enterprise Edition w wersji 5, która zapewnia wielowarstwową architekturę komponentową, a aplikacje uruchamiane są na serwerze aplikacyjnym. Aplikacja działa na serwerze aplikacyjnym JBoss i wykorzystuje bazę danych MySQL. Warstwa prezentacji złożona jest z takich elementów, jak: XHTML, CSS, Java Server Faces, Facelets, biblioteka komponentów JSF - Richfaces, AJAX. Warstwa dostępu do bazy danych stworzona została przy wykorzystaniu Java Persistence API i Hibernate. Logikę biznesową tworzą komponenty biznesowe EJB i POJO, a całość spięta jest szkieletem aplikacyjnym JBoss Seam.

5. Podsumowanie

OpenID jest niewątpliwie ciekawym pomysłem na poprawę komfortu korzystania przez użytkowników z wszelkiego rodzaju aplikacji webowych. Hasło “jeden login, jedno hasło do wszystkiego” znalazło również zainteresowanie największych korporacji informatycznych, takich jak Google, Microsoft i IBM zrzeszonych w ramach OpenID Foundation. Pojawiają się nowi dostawcy OpenID, przybywa witryn obsługujących ten protokół. Do OpenID część użytkowników podchodzi nieufnie, to samo dotyczy twórców aplikacji, którzy dosyć rzadko

implementują obsługę OpenID w swoich serwisach ze względu na zagrożenia, tj. np. phishing. Wraz z nadejściem specyfikacji OpenID Authentication w wersji 2.0 mechanizmy zapewniające bezpieczeństwo zostały znacznie usprawnione w stosunku do wersji 1.1.

Bardzo ciekawym projektem dla społeczności skupionej wokół OpenID jest np. https://www.idselector.com/ oferujący możliwość łatwego dodania uwierzytelniania OpenID do własnej aplikacji webowej. Rozwijane są pluginy do przeglądarek (np.Verisign do Mozilli Firefox), również poprawiające komfort korzystania z OpenID. Użytkownicy mają możliwość zapoznania się ze stronami wspierającymi protokół dzięki katalogom stron, jak np. www.openiddirectory.com. Największym wyzwaniem dla społeczności skupionej wokół OpenID jest podniesienie świadomości użytkowników Internetu, pokazanie zalet korzystania z protokołu, a także przełamanie obaw z nim związanych. Dotychczas, w Polsce jest bardzo mała liczba witryn wspierających ten protokół. Przyszłość pokaże, czy OpenID odniesie sukces i będzie powszechnie używaną metodą uwierzytelniania czy też pozostanie niespełnioną ideą.

BIBLIOGRAFIA

1. Fitzpatrick B., Hardt D., Hoyt J., Recordon D.: Specyfikacja OpenID Authentication 2.0.
2. Hardt D., Bufu J., Hoyt J.: OpenID Attribute Exchange 1.0.
3. Hoyt J., Daugherty J., Recordon D.: OpenID Simple Registration Extension 1.0.
4. Rafeeq Ur Rehman: Get ready for OpenID, Conformix Technologies Inc., 2008.

Recenzent: Prof. dr hab. inż. Alicja Wakulicz-Deja

Wpłynęło do Redakcji 4 lutego 2009 r.

Abstract

The article is devoted to the topic of OpenID protocol - new solution which revolutionized the authentication in web applications. In today's world users need one username and password per web application to get access to user's account. It causes many problems which are described in the introduction. OpenID solves that problems with motto "one username and password for everything". Current situation is presented on Fig.1 and OpenID solution on Fig.2. Chapter 2 introduces in details OpenID protocol. Firstly, specific OpenID terminology is described. Secondly, the protocol features (as well as its defects),

OpenID specifications and messages format are characterized. Chapter 2.3 describes in details the authentication process which is summarized on Fig.3. Table 1 contains examples of OpenID identifier normalization, table 2 - parameters being sent in authentication request, table 3 - parameters being sent to OpenID consumer. Chapter 3 is devoted to the notion of GreenID.net - OpenID provider implementation and security problem related to OpenID protocol (possible attacks, dangers etc.). The summary comprises considerations of the author concerning OpenID provider implementation (new features, ideas etc.) and possible future of the protocol.

Adresy

Marcin GORAWSKI: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-100 Gliwice, Polska, m.gorawski@polsl.pl.
Michał FLASIŃSKI: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-100 Gliwice, Polska, mflasinski@gmail.com.