

XIII Seminarium PLOUG
Warszawa
Kwiecień 2006

Problematyka bezpieczeństwa usług Web Services

Witold Andrzejewski, Maciej Zakrzewicz

*Wydział Informatyki i Zarządzania
Politechnika Poznańska
ul. Piotrowo 2, 60-965 Poznań
{wandrzejewski, mzakrzewicz}@cs.put.poznan.pl*

Abstrakt. Bezpieczeństwo komponentów usługowych Web Services publikowanych w sieci Internet lub wykorzystywanych przez aplikacje funkcjonujące w tej sieci podlega licznym zagrożeniom, m.in. podsłuchu komunikacji lub nieautoryzowanego dostępu. Metody ochrony komunikacji oferowane przez protokół HTTP, stanowiący podstawę dla SOAP, są dalece niewystarczające, przez co motywują rozwój specjalizowanych rozwiązań, np. WS-Security. Celem referatu jest omówienie dostępnych rozwiązań z zakresu podnoszenia bezpieczeństwa dostępu do rozproszonych komponentów usługowych Web Services.

Informacja o autorach. **Maciej Zakrzewicz** – pracownik Instytutu Informatyki Politechniki Poznańskiej oraz prezes Zarządu Stowarzyszenia Polskiej Grupy Użytkowników Systemu Oracle. Zainteresowania naukowe obejmują eksplorację danych (data mining), systemy baz danych/hurtowni danych oraz technologie internetowe. Dla krajowych i zagranicznych uniwersytetów oraz przedsiębiorstw (m.in. Niemcy, Wielka Brytania, USA, Słowacja, Słowenia) prowadzi wykłady i szkolenia z zakresu projektowania i implementacji systemów informatycznych. Kieruje i doradza w projektach informatycznych realizowanych w architekturach internetowych. Autor ponad 100 publikacji z zakresu odkrywania zbiorów częstych, przetwarzania zapytań eksploracyjnych, adaptatywnych serwerów WWW, technik indeksowania baz danych, strojenia systemów baz danych, standardów Java dla baz danych i dla aplikacji wielowarstwowych. Współtwórca serii konferencji i seminariów PLOUG, służących transferowi wiedzy i technologii wśród krajowych przedsiębiorstw informatycznych.

Witold Andrzejewski pracuje jako asystent na Wydziale Informatyki i Zarządzania Politechniki Poznańskiej. Jego zainteresowania naukowe obejmują eksplorację danych oraz indeksowanie złożonych typów danych. Jest twórcą wielu aplikacji do analizy i korekcji danych eksperymentalnych stosowanych na Wydziale Technologii Chemicznej Politechniki Poznańskiej oraz współtwórcą systemu do rejestracji przez Internet kandydatów na studia „Ksantypa” wykorzystywanego obecnie przez dziekanat Wydziału Informatyki i Zarządzania Politechniki Poznańskiej. Prowadzi działalność naukowo-dydaktyczną w zakresie programowania obiektowego, baz danych, multimedialnych i obiektowych baz danych oraz grafiki komputerowej.

1. Wstęp

Koncepcja usług Web Services pozwala na tworzenie luźno powiązanych ze sobą komponentów programowych, które można opublikować w sieci razem z formalnym opisem ich interfejsu wyrażonym w języku XML. Niezależny od platformy protokołów komunikacji z usługami Web Services (SOAP – Simple Object Access Protocol) działający nad protokołem HTTP oraz publicznie znany interfejs, pozwalają na łatwe ich wykorzystanie w dowolnej aplikacji, opartej na dowolnej platformie. Usługi Web Services mają wiele zastosowań. Mogą implementować logikę biznesową rozproszonych aplikacji, reprezentować aplikacje usługowe w sieci, albo, współpracując z innymi usługami, stanowić podstawę komunikacji pomiędzy wieloma systemami informatycznymi.

Obecnie prawie każde przedsiębiorstwo posiada jakiś system informatyczny wspomagający jego pracę. Zinformatyzowanie przedsiębiorstwa znacznie ułatwia wewnętrzny obieg danych np. pozwala na łatwe sprawdzenie ile towaru jest w magazynie, na łatwe obliczanie wyników finansowych na podstawie danych z kas itp. Komunikacja z innymi przedsiębiorstwami nie jest jednak równie łatwa, gdyż ich systemy informatyczne są bardzo często oparte o różne platformy programistyczne i posiadają niezgodne protokoły komunikacyjne. Opublikowanie części funkcjonalności takich systemów jako usługi Web Services pozwala na rozwiązanie również tego problemu. Niezależna od implementacji usługi i jej klienta komunikacja pozwala na integrację systemów informatycznych różnych przedsiębiorstw. Takie rozwiązanie ułatwia szereg czynności związanych z funkcjonowaniem przedsiębiorstw. Przykładowo, firma wycieczkowa może w łatwy sposób sprawdzić, czy w różnych hotelach są wolne pokoje i kiedy, ile oraz je ewentualnie zarezerwować, może również sprawdzić w firmie lotniczej rozkład lotów samolotów i zarezerwować bilety, bank może połączyć z miejscem pracy potencjalnego kredytobiorcy i sprawdzić jego dochody oraz warunki zatrudnienia, firma handlowa może w łatwy sposób złożyć zamówienie na nowy towar, sprawdzając uprzednio u różnych dostawców ceny hurtowe.

Podstawowym problemem komunikacji pomiędzy różnymi przedsiębiorstwami jest zapewnienie bezpieczeństwa przesyłanych danych. Wykorzystanie usług Web Services komunikujących się za pomocą Internetu otwiera wiele nowych „słabych punktów”, które mogą się stać przyczyną dostania się danych w niepowołane ręce. Błędnie zaprojektowane zabezpieczenia wymiany danych za pomocą usług Web Services mogą prowadzić do wielu poważnych skutków: wykradzenia danych handlowych przez konkurencję i utraty danych osobowych klientów, jak również tworzy okazje do wielu różnych oszustw i kradzieży. W konsekwencji przedsiębiorstwo może utracić zaufanie klientów i partnerów, odnotować spadek przychodów i straty, a dodatkowo popaść w konflikt z prawem w wyniku niedotrzymania prawnie zagwarantowanej ochrony danych osobowych klientów.

W niniejszej publikacji przedstawiono różne rozwiązania z zakresu podnoszenia bezpieczeństwa dostępu do rozproszonych komponentów usługowych Web Services. Przedstawiono rozwiązania w różnych stadium rozwoju: od rozwiązań dojrzałych, które zostały już przyjęte przez rynek informatyczny, poprzez rozwiązania młode, dopiero co opracowane i jeszcze niezbyt rozpowszechnione, aż do rozwiązań dopiero opracowywanych.

Struktura dalszej części niniejszej publikacji wygląda następująco. W sekcji 2 przedstawiono podstawowe definicje wykorzystywane w niniejszej pracy. Sekcja 3 przedstawia rozwiązania charakterystyczne dla zabezpieczania transmisji danych protokołem HTTP i pokazuje dlaczego rozwiązania te nie są wystarczające. W sekcji 4 opisano standardy bezpieczeństwa opracowane dla usług Web Services. Sekcja 5 podsumowuje publikację, a w sekcji 6 przedstawiono bibliografię.

2. Definicje

Poprzez *bezpieczeństwo* należy rozumieć następujące wymagania: *poufność*, która zapewnia, że przesyłane dane może przeczytać tylko adresat, *uwierzytelnianie*, które gwarantuje, że dostęp do usług mają jedynie jednostki, które posiadają odpowiedni dowód tożsamości, *integralność* oznaczającą pewność, że komunikat dotarł do adresata niezmieniony, *niezaprzeczalność*, dzięki której

nadawca nie może się wyprzeć nadania wiadomości oraz *autoryzacja*, która pozwala określić jaki podmiot ma dostęp do jakich zasobów [1].

Algorytmami *kryptografii symetrycznej* nazywamy zbiór algorytmów służących do szyfrowania danych, w których do zaszyfrowania i odszyfrowania wiadomości używa się tego samego *klucza*. W przeciwieństwie do kryptografii symetrycznej istnieje również *kryptografia asymetryczna*, w której używa się dwóch kluczy: *klucza publicznego* służącego do szyfrowania i *klucza prywatnego* służącego do odszyfrowywania wiadomości. Algorytmy kryptografii symetrycznej można zastosować do stworzenia tzw. *podpisu elektronicznego*, który zapewnia uwierzytelnienie i niezaprzeczalność oraz integralność podpisywanych wiadomości. Sprawdzenie przez kogo została podpisana wiadomość jest możliwe, jeżeli znany jest klucz publiczny podpisującego podmiotu. W celu powiązania tożsamości podmiotu z kluczem publicznym wystawiany jest elektroniczny dokument zawierający dane dotyczące tożsamości oraz klucz publiczny, który jest podpisany cyfrowo przez trzecią, zaufaną stronę. Dokument ten nazywany jest *certyfikatem*, a trzecia strona podpisująca ten dokument nazywana jest *administratorem certyfikatów* (*certificate authority*). Układ, w którym trzecia strona może potwierdzać, bądź zaprzeczać tożsamości podmiotów, jak również tworzyć certyfikaty nazywa się *infrastrukturą klucza publicznego* (PKI) [2, 3, 4]. *Funkcją skrótu* nazwano funkcję haszującą generującą dla każdej wiadomości jej *skrót* o stałej długości. Wymagane jest dodatkowo, aby było obliczeniowo trudne znalezienie drugiej wiadomości o takim samym skrótzie jak znana już wiadomość.

Żetonem bezpieczeństwa (w skrócie *żetonem*) nazwano zbiór twierdzeń o podmiocie, takich jak: tożsamość, nazwa, klucz, uprawnienia itp. *Podpisany żetonem bezpieczeństwa* nazwano żeton, który jest kryptograficznie potwierdzony przez administratora certyfikatów. Może być nim zatem dowolny certyfikat [5].

3. Dotychczasowe rozwiązania

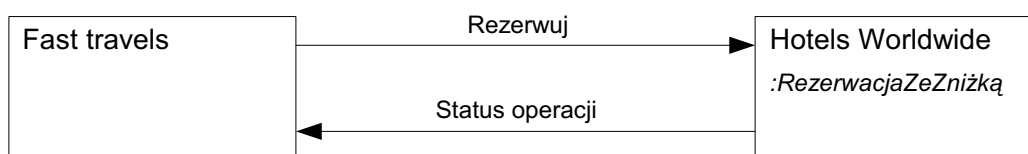
Najczęściej spotykanym mechanizmem uwierzytelniania podmiotu (np. użytkownika) w Internecie jest technika BASIC-AUTH, która jest zdefiniowana w RFC-2617 [6]. Dostęp do zasobu chronionego za pomocą BASIC-AUTH jest możliwy jedynie w sytuacji, gdy w nagłówku żądania HTTP zostanie przesłana nazwa użytkownika i jego hasło zakodowane metodą Base64. Serwer WWW, z którym następuje połączenie, posiada listę kontroli dostępu, która pozwala również na przeprowadzenie autoryzacji. Mechanizm ten, nie jest wystarczająco bezpieczny, gdyż bardzo łatwo jest podsłuchać nazwę użytkownika i hasło. W celu poradzenia sobie z tym problemem stosuje się protokół SSL/TLS (RFC-2246 [7]). W protokole tym podczas nawiązywania połączenia dochodzi do wymiany certyfikatów i kluczy publicznych serwera i użytkownika oraz do uzgodnienia symetrycznego klucza, który jest następnie wykorzystywany do szyfrowania przesyłanych danych. Protokół SSL/TLS zapewnia uwierzytelnianie zarówno klienta jak i użytkownika (poprzez weryfikację certyfikatów), poufność (dzięki szyfrowaniu) oraz integralność (razem z danymi jest przesyłany również ich skrót). Nie zapewnia on jednak niezaprzeczalności, ponieważ zarówno serwer jak i użytkownik stosują ten sam klucz. W protokole SSL nie jest obowiązkowe przesyłanie certyfikatu i klucza publicznego przez użytkownika przez co nie jest zapewnione również uwierzytelnianie klienta. Aby zapewnić taką możliwość można stosować technikę BASIC-AUTH.

Powyższe rozwiązania można zastosować również do zapewniania bezpieczeństwa przy komunikacji usług Web Services, których protokół komunikacji (SOAP) jest oparty na HTTP. Rozwiązania te nie są jednak wystarczające. Brak niezaprzeczalności powoduje, że np. klient może twierdzić, że nie wysłał zamówienia. Dodatkowo, specyfika protokołu SOAP pozwala na to, że w komunikacji pomiędzy dwoma podmiotami mogą pojawiać się pośrednicy, a u każdego z nich może zająć potrzeba zatajenia części wiadomości. Niestety, protokół SSL szyfruje cały komunikat, a dodatkowo, jako protokół warstwy transportowej, zapewnia komunikację jedynie pomiędzy dwoma węzłami, przez co umożliwia pośrednikom odczytanie całego komunikatu [1].

4. Standardy bezpieczeństwa dla usług Web Services

Wpływ na tworzenie standardów dla usług Web Services, jak również posiadanie praw własności takich standardów, które można wykorzystać do sprzedaży licencji, jest potencjalnym źródłem olbrzymich zysków. Nic dziwnego zatem, że na rynku powstały zwalczające się nawzajem grupy, które tworzą sprzeczne i częściowo pokrywające się standardy. Obecnie nie mniej niż cztery organizacje: Liberty Alliance, Organization for the Advancement of Structured Information Standards (OASIS), World Wide Web Consortium (W3C) oraz Web Services Interoperability Organization (WS-I) walczą o przewodnictwo nad procesem tworzenia standardów. Powstały dwie przeciwne grupy producentów: jedna oparta na przymierzu IBM i Microsoftu, i druga, w której znajdują się praktycznie wszyscy inni [8]. W niniejszej publikacji przedstawiono standardy opracowane przez każdą z tych grup, razem z zaznaczeniem części wspólnych pomiędzy niektórymi z tych standardów.

Wszystkie standardy, razem z problemami, do rozwiązywania których zostały stworzone, zostaną przedstawione w oparciu o następujący przykład. Firma turystyczna „Fast Travels” współpracuje z firmą reprezentującą sieć hoteli „Hotels Worldwide”. Obie firmy posiadają swoje systemy informatyczne, które wspomagają ich pracę. Firma „Hotels Worldwide” postanowiła rozbudować swój system informatyczny poprzez udostępnienie części swoich usług jako usługi Web Services. Firma „Fast Travels” postanowiła wykorzystać ten fakt do rozbudowy swojego systemu informatycznego o możliwość łatwego wykorzystywania usług sieci hoteli. Sytuację tą przedstawia rysunek 1. Przedstawia on prosty scenariusz, w którym system informatyczny firmy „Fast Travels” dokonuje rezerwacji pokoju w hotelu za pomocą usługi Web Service „RezerwacjaZeZniżką”. Niniejszy przykład, po niewielkich zmianach, został wzięty z serii artykułów [9, 10, 11, 12].



Rysunek 1. Przykładowa wymiana komunikatów SOAP.

Komunikacja pomiędzy firmą turystyczną a siecią hoteli jest podatna na wiele zagrożeń. Zależy zatem konieczność zapewnienia bezpieczeństwa przesyłanych danych. Rozważając przykład można zauważyć, że sieć hotelowa nie może ślepo wierzyć każdemu przychodzącemu wywołaniu usługi. Przykładowo, usługa „RezerwacjaZeZniżką” jest dostępna jedynie dla partnerów handlowych. Konieczny jest zatem dowód, że usługę wywołuje system informatyczny zaufanego partnera. Problem ten można rozwiązać poprzez zastosowanie podpisu elektronicznego do podpisywania komunikatu SOAP wywołującego usługę. Sposób podpisywania dokumentów XML (komunikaty SOAP są dokumentami XML) opisuje specyfikacja XML Digital Signature (XMLDS) [13]. Specyfikacja XMLDS została opracowana przez W3C i Internet Engineering Task Force (IETF) w celu zapewnienia integralności przesyłanych dokumentów XML oraz uwierzytelniania nadawcy. XMLDS jest od 12 lutego 2002 rekomendacją (standardem) W3C. Standard ten nie specyfikuje konkretnego algorytmu i typu podpisu elektronicznego, ale pozwala na stosowanie wielu różnych algorytmów kryptografii symetrycznej i asymetrycznej do wygenerowania podpisu, oraz później do uwierzytelnienia wiadomości oraz nadawcy (w tym między innymi za pomocą biletów Kerberos lub certyfikatów X.509v3). Ponieważ podpisywany jest zawsze skrót przesyłanego dokumentu, to zapewniona jest również integralność przesyłanych danych.

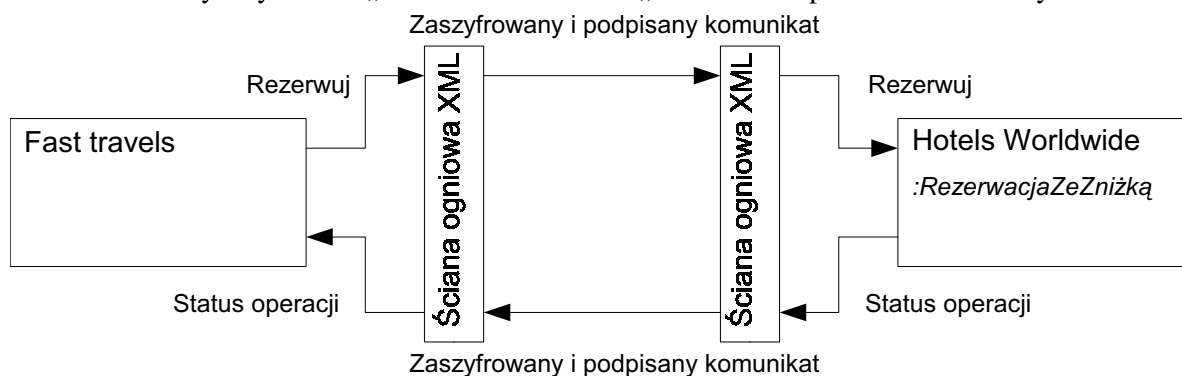
Pomimo zapewnienia uwierzytelniania i integralności wywołań SOAP dane przesyłane są jawnie, przez co są podatne na podsłuchiwanie. Konieczne jest zapewnienie poufności przesyłanych danych poprzez ich zaszyfrowanie. Sposób przesyłania zaszyfrowanych danych w postaci dokumentów XML został opracowany w specyfikacji XML Encryption (XMLENC) [14]. Specyfikacja XMLENC od 10 grudnia 2002 jest rekomendacją W3C. Dzięki XMLENC można zaszyfrować

cały dokument XML, pojedynczy element dokumentu XML, lub jego zawartość, dowolne dane, które nie są dokumentem XML, a nawet już raz zaszyfrowany dokument. Specyfikacja pozwala na zdefiniowanie plików XML zawierających dane zaszyfrowane dowolnym algorytmem kryptograficznym, zarówno symetrycznym jak i asymetrycznym. W jednym pliku XML można również umieścić fragmenty zaszyfrowane różnymi algorytmami. Dzięki standardowi XMLENC możliwe jest konstruowanie komunikatów SOAP, w których poszczególne fragmenty mają różny poziom poufności, przykładowo część danych może zostać odczytana jedynie przez pośredników, a część jedynie przez adresata.

Standardy XMLDS i XMLENC są standardami ogólnymi, nie związanymi w żaden sposób z technologią usług Web Services. Można je wykorzystać do podpisania lub zaszyfrowania czegośkolwiek. Sposób wykorzystania tych standardów do zabezpieczenia wiadomości SOAP jest zdefiniowany przez standard WS-Security (WSS) [5, 15]. Jest to standard organizacji OASIS, od marca 2004. 1 lutego 2006 OASIS zatwierdziło również jego nową wersję 1.1. Standard WSS nie opisuje jednego protokołu wymiany komunikatów SOAP, ale stanowi szkielet, na bazie którego można budować własne protokoły bezpieczeństwa. Umożliwia wykorzystanie dowolnych algorytmów podpisu elektronicznego i szyfrowania w celu zapewnienia uwierzytelniania, integralności, poufności i niezaprzeczalności przesyłanych komunikatów. Pozwala na definiowanie w nagłówku komunikatu SOAP całego łańcucha kolejnych transformacji i podpisów, którym podlegają fragmenty, bądź całość komunikatu, w celu zapewnienia różnych poziomów poufności, pozwalając pośrednikom na dostęp jedynie do fragmentów komunikatu przeznaczonych dla nich.

W niniejszym miejscu należy wprowadzić pojęcie *ściany ogniowej XML*. Normalne ściany ogniowe służą do blokowania przepływu pakietów TCP/IP i abstrahują od ich zawartości. Ściana ogniowa XML działa na wyższym poziomie, gdyż potrafi przeczytać zawartość komunikatu SOAP i sprawdzić integralność wiadomości oraz poprawność podpisu elektronicznego, jak również odszyfrować fragment wiadomości i przekazać ją dalej do usługi Web Service. Ściana ogniowa XML stanowi zatem warstwę odpowiedzialną za to, aby do systemu informatycznego przedsiębiorstwa dostawały się jedynie poprawne, uwierzytelnione i nie zmienione komunikaty. Może ją tworzyć zarówno dedykowane urządzenie, wzorem sprzętowych ścian ogniowych, może to być również osobny program (wzorem programowych ścian ogniowych) lub po prostu fragment platformy implementującej usługi Web Services.

Wykorzystanie XMLDS, XMLENC i WSS oraz ściany ogniowej XML do zabezpieczenia systemów informatycznych firm „Hotels Worldwide” i „Fast travels” przedstawiono na rysunku 2.

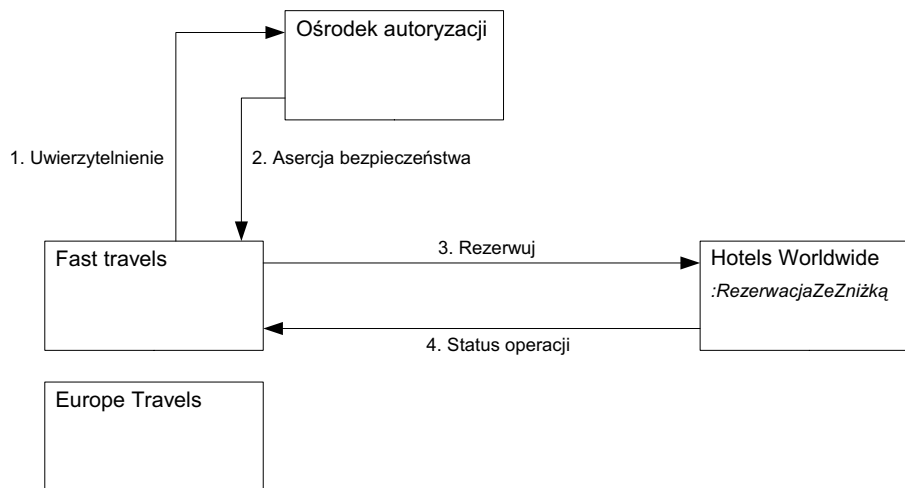


Rysunek 2. Zabezpieczona wymiana komunikatów SOAP.

Standardy XMLDS, XMLENC i WSS są dojrzałymi standardami wykorzystywanymi już obecnie na rynku. Zapewniają one podstawowe wymagania bezpieczeństwa, takie jak: uwierzytelnianie, integralność, poufność i niezaprzeczalność. Autoryzację można łatwo zapewnić na poziomie usługi, jeżeli jej wywołanie jest należycie uwierzytelnione. W celu promowania tych standardów organizacja WS-I opracowała Basic Security Profile, który jest zbiorem rekomendowanych standardów dotyczących bezpieczeństwa usług Web Services wraz z pewnymi uściśleniami ła-

twiającymi współpracę różnych systemów informatycznych. Wśród rekomendowanych standardów znajdują się: SSL/TLS, XMLDS, XMLENC i WSS.

Dotychczas omówiono jedynie komunikację pomiędzy dwoma podmiotami, jednak w rzeczywistym świecie wiele nieznanymi wcześniej podmiotów biznesowych może potrzebować dostępu do usług sieciowych udostępnionych przez system informatyczny przedsiębiorstwa. Zarządzanie danymi służącymi do uwierzytelniania i autoryzacji (np., certyfikatami, parami nazwa użytkownika – hasło) wszystkich potencjalnych klientów spowodowałoby nadmierną rozbudowę funkcjonalności usługi, która powinna się koncentrować jedynie na wykonywaniu swoich zadań. Należy zatem przenieść tę funkcjonalność do osobnego *ośrodka autoryzacji* wydającego tzw. *asercje bezpieczeństwa* [1]. Idea ta została przedstawiona na rysunku 3. Z usług sieci hoteli prócz firmy turystycznej „Fast travels” chce korzystać również inna firma turystyczna „Europe travels”. Aby uzyskać dostęp do usługi „RezerwacjaZeZniżką” obie firmy muszą najpierw skontaktować się z zaufanym ośrodkiem autoryzacji w celu uzyskania asercji bezpieczeństwa. Przy założeniu, że sieć hoteli „Hotels Worldwide” ufa ośrodkowi autoryzacji, asercja bezpieczeństwa, dołączona do żądania SOAP, stanowi przepustkę do jej usług. Dodatkowym problemem jaki można sobie wyobrazić jest fakt, że poszczególne firmy turystyczne będą wielokrotnie korzystać z usług sieci hotelowej w krótkim czasie, przykładowo, będą wielokrotnie próbować rezerwować miejsca w różnych hotelach. W takiej sytuacji przeprowadzanie pełnego uwierzytelniania i autoryzacji przy każdym wywołaniu usługi jest zbyt czasochłonne. Problem ten jest rozwiązywany przez koncepcję *Single Sign On*, według której proces autoryzacji i uwierzytelniania jest przeprowadzony jednokrotnie, a otrzymana przepustka może być wykorzystana wiele razy. Powyższą problematyką zajmuje się wiele specyfikacji, często sprzecznych i/lub częściowo pokrywających się. Wśród tych specyfikacji należy wymienić: *Security Assertion Markup Language (SAML)*, *eXtensible Access Control Markup Language (XACML)*, *Extensible Rights Markup Language (XrML)*, *WS-Trust*, *WS-Authorization* i *WS-SecureConversation*:



Rysunek 3. Wymiana komunikatów SOAP z uwierzytelnieniem przez ośrodek autoryzacji.

- Specyfikacja SAML jest standardem OASIS. Dotychczas opracowano i zatwierdzono kilka wersji: 1.0 5 listopada 2002 [16], 1.1 2 września 2003 [17] i 2.0 15 marca 2005 [18]. Standardy SAML 1.0 i 1.1 definiują mechanizmy żądania i przesyłania zwrotnego kilku rodzajów asercji: asercji dotyczących uwierzytelnienia, asercji dotyczących atrybutów i asercji dotyczących decyzji [1]. Asercja dotycząca uwierzytelnienia jest zaświadczeniem, że podmiot żądający asercji uwierzytelniał się w ośrodku autoryzacji w danym czasie przy użyciu danej metody autoryzacji. Asercja ta nie zawiera jednak żadnych danych dotyczących uwierzytelniającego się podmiotu. Innym typem asercji jest asercja dotycząca atrybutów, która jest dowodem, że podmiot, który się nią posługuje posiada własności zdefiniowane w asercji (mogą to być przykładowo e-mail, na-

zwa firmy, nazwa użytkownika itp.). Asercja dotycząca atrybutów jest wystarczająca do przeprowadzenia autoryzacji po stronie usługi. W celu przeprowadzenia autoryzacji przez ośrodek autoryzacji, musi on wydać asercję dotyczącą decyzji, która reprezentuje uprawnienie do skorzystania z konkretnego zasobu. SAML wspiera również koncepcję Single Sign On przez co umożliwia wydajną wymianę serii komunikatów. Specyfikacja SAML początkowo opracowana przez OASIS (wersje 1.0 i 1.1) została później rozbudowana przez Liberty Alliance i Internet2 Shibboleth Group, w celu umożliwienia komunikacji pomiędzy ośrodkami autoryzacji, wprowadzenia lepszej ochrony prywatności i lepszego opracowania metod uwierzytelniania. Te modyfikacje stały się podstawą standardu SAML w wersji 2.0 [19].

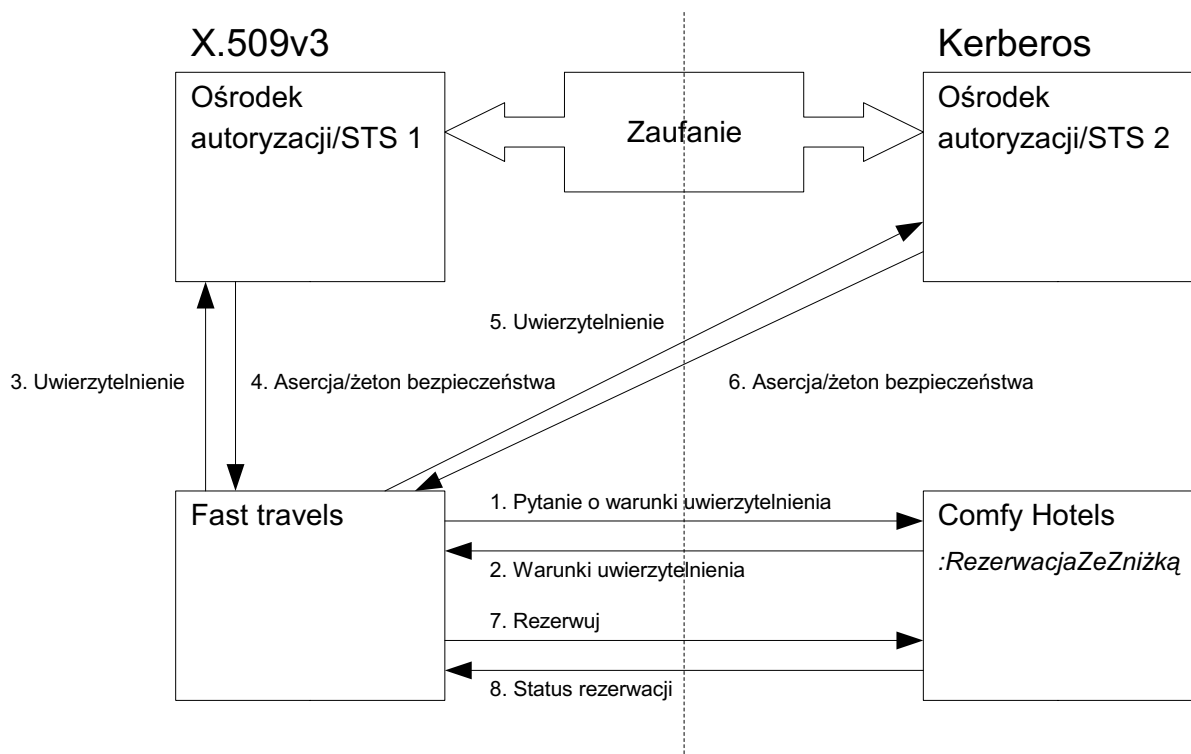
- Specyfikacja XACML jest również standardem OASIS. Wersję 1.0 tego standardu zatwierdzono 18 lutego 2003 [20], 1.1 7 lipca 2003 [21] i 2.0 1 lutego 2005 [22]. XACML definiuje sposoby opisu polityki dostępu do zasobów, a więc sposoby opisu metod i warunków autoryzacji. Wykorzystując XACML można zdefiniować warunki dostępu do zasobów w oparciu o praktycznie dowolne dostępne informacje dotyczące zasobu, między innymi jego zawartości (jeśli jest to dokument), oraz takich czynników jak aktualna data, godzina albo lokalizacja zasobu. Można również wziąć pod uwagę własności podmiotów żądających dostępu do zasobu, takich jak np. ich przynależność do określonych grup albo ról. XACML można używać niezależnie od pozostałych standardów użytych w polityce bezpieczeństwa, ale standard ten opisuje również funkcjonalność, którą można wykorzystać jedynie przy współpracy z SAML [19]. Niestety składnia XACML jest tak skomplikowana, że praktycznie nieczytelna dla człowieka i musi być generowana automatycznie przez odpowiednie aplikacje narzędziowe.

- Specyfikacja XrML [23] została ogłoszona 26 listopada 2001 przez kompanię ContentGuard, oraz zgłoszona do OASIS, ale póki co nie został opracowany przez tą organizację żaden standard oparty o nią oparty. Specyfikacja XrML została zdefiniowana przede wszystkim do zarządzania uprawnieniami dostępu do danych w kontekście ochrony praw autorskich (*Digital Rights Management* – DRM), ale może być wykorzystana do definiowania polityki autoryzacji dla dowolnych zasobów. Uprawnienia w XrML mogą być definiowane w oparciu o 4 koncepcje: zasobu, uprawnień, podmiotu, który żąda uprawnienia do zasobu oraz warunku, jaki musi być spełniony, by podmiot otrzymał uprawnienie. Jak łatwo zauważyć zakres problemów rozwiązywanych przez specyfikację XrML pokrywa się z XACML. Jedyną różnicą jest to, że XACML jest bardzo ogólny, podczas, gdy XrML skupia się przede wszystkim na problemach związanych z DRM [24]. Specyfikacje te nie są kompatybilne.

- Specyfikacja WS-Trust została opracowana przez IBM, Microsoft i wiele innych firm [25]. Nie została ona jednak jeszcze zatwierdzona jako standard ani przez W3C ani OASIS. Zakres tej specyfikacji pokrywa się częściowo ze specyfikacją SAML. Podobnie jak w SAML, w WS-Trust zdefiniowano sposoby uwierzytelniania dostępu do usług przy pomocy trzeciej strony pełniącej funkcję ośrodka autoryzacji. Model bezpieczeństwa definiowany przez WS-Trust opiera się na tym, że każda usługa Web Service może wymagać, aby przychodzące żądanie zawierało podpisany żeton bezpieczeństwa [26]. Warto zauważyć, że podpisany żeton bezpieczeństwa jest podobny do asercji bezpieczeństwa stosowanych w SAML. W rzeczywistości asercje bezpieczeństwa SAML są specyficznymi dla tego protokołu żetonami bezpieczeństwa. Jeżeli komunikat SOAP nie zawiera poprawnego żetonu bezpieczeństwa, to jest odrzucany. Jeżeli podmiot nie jest w stanie samodzielnie stworzyć żetonu bezpieczeństwa, może się zwrócić do strony trzeciej, tzw. *Usługi Żetonu Bezpieczeństwa* (*Security Token Service*, STS), która może mu go przesłać. STS może również wymagać jakiejś formy żetonu bezpieczeństwa w celu uwierzytelnienia żądania. Jeżeli podmiot uwierzytelnia się u STS, otrzymuje żeton umożliwiający uwierzytelnienie się u usługi Web Service. Jak łatwo zauważyć STS jest analogiem ośrodka autoryzacji SAML. Standard WS-Trust definiuje również protokół oparty na technice wyzwanie – odpowiedź, który zapewnia usłudze Web Service możliwość weryfikacji u STS poprawności żetonów przesłanych przez nadawcę komunikatu SOAP. WS-Trust posiada jedną przewagę nad SAML. Otóż dopuszcza różne typy żetonów bezpieczeństwa, podczas, gdy SAML wykorzystuje jedynie charakterystyczne dla siebie asercje.

- WS-Authorization jest niekompletną jeszcze specyfikacją opracowywaną przez grupę firm dookoła IBM i Microsoft. WS-Authorization będzie specyfikować sposoby autoryzacji podmiotów przez usługi Web Services. Jak łatwo zauważyć, specyfikacja ta pokrywa się częściowo z XACML. Niestety, obecnie nie jest jeszcze dostępna nawet treść wstępnej wersji tej specyfikacji. Zapowiedziano jedynie jej opracowanie w *white paper* opublikowanym przez IBM i Microsoft [27].
- WS-SecureConversation jest kolejną specyfikacją opracowaną przez grupę firm dookoła IBM i Microsoft [28]. Jej najnowsza dostępna wersja pochodzi z lutego 2005. Głównym zadaniem WS-SecureConversation jest zaimplementowanie filozofii Single Sign On w celu usprawnienia wymiany komunikatów SOAP. Dzięki WS-SecureConversation możliwe jest jednokrotne wzajemne uwierzytelnienie obu stron, oraz ustalenie wspólnego „sekreту” np. klucza do symetrycznego algorytmu szyfrowania komunikacji, który będzie wykorzystywany do szyfrowania wymiany wszystkich kolejnych komunikatów SOAP aż do wygaśnięcia sesji [26]. 17 października 2005 OASIS ogłosiła prośbę o udział (*Call for Participation*) w celu utworzenia komitetu *Web Services Secure Exchange* (WS-SX) [29], który miałby opracować standard mający na celu zapewnienie bezpiecznej i wydajnej wymiany wielu komunikatów SOAP oraz opracowanie sposobu komunikowania przez usługi Web Services wymagań co do bezpieczeństwa takich wymian. Standard ten ma powstać w oparciu o specyfikacje WS-SecureConversation, WS-Trust i WS-SecurityPolicy (ta ostatnia zostanie opisana w dalszej części niniejszej pracy).

Przykład przedstawiony na rysunku 3 nie oddaje całkowicie rzeczywistości. Może się zdarzyć bowiem, że firma „Fast Travels” będzie chciała skorzystać z usług innej sieci hoteli np. „Comfy Hotels”. Ze skorzystaniem z usług tej sieci jest jednak problem. Otóż jej polityka bezpieczeństwa oparta jest o inny, niż stosowany w „Fast Travels” sposób podpisywania komunikatów, a dodatkowo „Comfy Hotels” korzysta również z innego ośrodka autoryzacji. Przykładowo, sieć „Fast Travels” wykorzystuje do uwierzytelniania certyfikaty X.509v3 a sieć hoteli „Comfy Hotels” bilety Kerberosa. Nowy, rozbudowany przykład można zobaczyć na rysunku 4. Przedstawiony powyżej problem można rozwiązać jeżeli istnieje zaufanie pomiędzy ośrodkami autoryzacji. Jak wspomniano przy okazji omawiania standardu SAML, w wersji 2.0 tego standardu dodano możliwość nawiązania komunikacji pomiędzy ośrodkami autoryzacji. Komunikacja ta ma właśnie na celu ustalenie pomiędzy nimi zaufania [19]. Prócz standardu SAML, rozwiązywaniem tego typu problemów zajmują się: specyfikacja *WS-Federation* i standard *Identity Web Service Framework* (ID-WSF):



Rysunek 4. Wymiana komunikatów SOAP w sytuacji gdy stosowane są różne algorytmy uwierzytelniania.

- Specyfikacja WS-Federation została opracowana przez IBM, Microsoft i innych [30]. Jej najnowsza wersja ukazała się lipcu 2003. Umożliwia ona stworzenie mechanizmów tłumaczenia np. żetonów bezpieczeństwa stosowanych przez „Fast Travels”, na żetony stosowane przez „Comfy Hotels”. Mechanizm ten można wytłumaczyć na następującym przykładzie [26] (kolejne kroki przedstawiono na rysunku 4):

1. System informatyczny firmy „Fast Travels” chce skorzystać z usługi „Rezerwacja-ZeZniżką” w sieci hoteli „Comfy Hotels”. W tym celu wysyła zapytanie, jakie warunki musi spełnić, żeby móc skorzystać z tej usługi.
2. System otrzymuje informację, że musi skorzystać z STS numer 2.
3. Ponieważ system nie może skorzystać z STS numer 2 bezpośrednio, uwierzytelnia się u STS numer 1 i zgłasza prośbę o żeton.
4. System otrzymuje żeton od STS numer 1.
5. System wykorzystuje żeton do uwierzytelnienia się u STS numer 2 i otrzymania kolejnego żetonu.
6. System otrzymuje żeton od STS numer 2.
7. System wykorzystuje drugi żeton do uwierzytelnienia się u usługi w sieci hoteli „Comfy Hotels” i wywołania tejże usługi.
8. System otrzymuje wynik wykonania usługi.

- Standard ID-WSF [31, 32] został opracowany przez Liberty Alliance w celu rozwiązywania podobnych problemów co WS-Federation i częściowo się z nim pokrywa. W przeciwieństwie do WS-Federation, który opiera się na innych standardach IBMa i Microsoftu (np. WS-Trust), ID-WSF jest oparty o standardy SAML, WS-Security i SSL. Szczegółowy wykaz podobieństw i różnic można przeczytać w [33].

Prócz standardów związanych z uwierzytelnianiem, autoryzacją, poufnością, integralnością i niezaprzeczalnością komunikatów SOAP w mniej lub bardziej skomplikowanych środowiskach heterogenicznych, powstało wiele innych standardów wspomagających realizację zadań tych wcześniejszych. Wśród standardów wspomagających można wymienić: *Digital Signature Service*

(DSS), *XML Key Management Specification* (XMKS), *WS-Policy*, *WS-PolicyAttachment*, *WS-PolicyAssertion*, *WS-SecurityPolicy*, *Web Service Policy Language* (WSPL), *WS-Privacy* i *XML Common Biometric Format* (XCBF).

- Standard DSS jest nie ukończonym jeszcze standardem OASIS, który ma na celu ułatwienie przetwarzania podpisu elektronicznego. Wśród rzeczy, które mają się znaleźć w standardzie jest, między innymi, interfejs, który można wykorzystać do wysłania żądania, aby usługa Web Service utworzyła, bądź zweryfikowała podpis elektroniczny na przesłanych danych, z wykorzystaniem wskazanych algorytmów. Specyfikacja DSS opisuje usługi dotyczące: tworzenia podpisów elektronicznych, ich weryfikacji, tworzenia znaczników czasowych oraz dowolnej kombinacji tych operacji. Dzięki temu standardowi będzie można zbudować łatwo dostępne usługi Web Services, które można wykorzystać do przetwarzania podpisu elektronicznego, bez konieczności implementowania tych operacji we własnej aplikacji. [34]
- Specyfikacja XKMS [35] jest od 30 marca 2001 rekomendacją W3C. Na XKMS składają się dwie specyfikacje: *XML Key Information Service Specification* (X-KISS) oraz *XML Key Registration Service Specification* (X-KRSS). Celem XKMS jest zapewnienie łatwego zarządzania kluczami publicznymi przez aplikacje korzystające z usług Web Services. X-KISS jest protokołem pozwalającym na stworzenie usługi, do której można przekazać przetwarzanie informacji dotyczących klucza publicznego. Przykładowo, jeżeli aplikacja nie potrafi przetwarzać certyfikatów X.509v3, a otrzymała taki certyfikat w komunikacie SOAP, może przekazać przetwarzanie certyfikatu do usługi zgodnej z X-KISS, w celu ekstrakcji klucza. X-KISS pozwala również na otrzymanie, na podstawie certyfikatu, asercji dotyczących atrybutów związanych z właścicielem podpisu. X-KRSS jest protokołem pozwalającym na stworzenie usługi, której zadaniami są między innymi: rejestracja, usuwanie i ponowne przesłanie klucza publicznego w taki sposób, by możliwe było późniejsze wykorzystanie tego klucza za pomocą X-KISS, bądź dowolnej PKI takiej jak X.509 lub PKIX [24].
- WS-Policy [36] jest kolejną ze specyfikacji IBM i Microsoftu. Jej najnowsza wersja ukazała się w marcu 2006. Zadaniem tej specyfikacji jest zapewnienie szkieletu, który można wykorzystać do komunikowania przez usługę Web Services twierdzeń dotyczących jej cech i wymagań, takich jak na przykład: wymagane kodowanie przesyłanych komunikatów albo obsługiwane algorytmy podpisu elektronicznego itp. WS-Policy jest jedynie szkieletem, natomiast to jakie cechy usługi można komunikować definiują osobne specyfikacje zależne od domeny zastosowań. Jedną z takich specyfikacji jest WS-PolicyAssertion [37] (ostatnia wersja pochodzi grudnia 2002), która definiuje podstawowe, ogólne twierdzenia, takie jak np. wcześniej wspomniane kodowanie komunikatów. Z punktu widzenia bezpieczeństwa należy wymienić przede wszystkim WS-SecurityPolicy [38] (najnowsza wersja specyfikacji pochodzi z lipca 2005), która definiuje twierdzenia związane z wymaganiami dotyczącymi zabezpieczenia komunikacji z daną usługą Web Service. WS-Policy, wraz z WS-SecurityPolicy jest wykorzystywana przez takie rozwiązania jak: WS-Trust, WS-SecureConversation i WS-Federation do komunikowania wymagań jakie podmiot musi spełnić, aby zostać uwierzytelnionym. WS-Policy wykorzystuje również specyfikację WS-PolicyAttachment (najnowsza wersja pochodzi z marca 2006), która służy do definiowania powiązań pomiędzy twierdzeniami, a podmiotami, których te twierdzenia dotyczą. WS-PolicyAttachments definiuje również jak można powiązać twierdzenia WS-Policy z opisami WSDL i UDDI [39].
- Język WSPL został opracowany przez OASIS w oparciu o język XACML (jest jego podzbiorem). Obecnie nie jest jeszcze standardem, ale wstępną wersję można przeczytać w sieci [40, 41]. Dziedzina zastosowań języka pokrywa się z WS-Policy, ale pozwala on na dokładniejsze i wygodniejsze definiowanie twierdzeń o usługach Web Services.
- WS-Privacy jest niekompletną jeszcze specyfikacją opracowywaną przez grupę firm dookoła IBM i Microsoft. Ma być oparta o WS-Policy, WS-Trust i WS-Security. Celem WS-Privacy jest umożliwienie usługom Web Services komunikowanie twierdzeń o ich polityce dotyczącej danych osobowych. Dzięki temu, przedsiębiorstwa będą mogły uniknąć naruszenia ochrony danych osobowych swoich klientów poprzez przesłanie tych danych do usługi, która jest spreczna

z polityką firmy w tej sprawie. Niestety, obecnie nie jest jeszcze dostępna nawet treść wstępnej wersji tej specyfikacji. Jedynie zapowiedź jej opracowania została umieszczona w *white paper* opublikowanym przez IBM i Microsoft [27].

- XCBF jest formatem zapisu danych biometrycznych w postaci dokumentu XML. Można go wykorzystać jako żeton bezpieczeństwa przy uwierzytelnianiu komunikatów SOAP [42].

To co dotychczas opisano praktycznie wyczerpuje opracowane dotychczas standardy i specyfikacje dotyczące bezpieczeństwa w usługach Web Services. Należałoby zatem zadać sobie pytanie. Czy zastosowanie powyższych standardów gwarantuje bezpieczeństwo? Odpowiedź brzmi oczywiście „nie”. Bezpieczeństwo aplikacji zależy bowiem nie tylko od zastosowanych w niej standardów, ale również od projektu samej aplikacji. Nawet najlepiej implementujące standardy usługi Web Services mogą być podatne na ataki typu SQL Injection lub wiele innych sposobów włamań. Niebezpieczeństwo z tej strony można zminimalizować jedynie starannie projektując usługi i przeprowadzając długie testy. Istnieje również jeszcze jeden standard, który może się przydać przy minimalizacji tego problemu. Jest to specyfikacja opartego na XMLu języka AVDL [43] (*Application Vulnerability Description Language*) zatwierdzona jako standard przez OASIS w maju 2004. Język AVDL służy do opisywania dziur odkrytych w różnych aplikacjach. Sformalizowanie opisu tego typu problemów pozwala często na automatyczne zapobieganie naruszeniom bezpieczeństwa. Przykładowo, niech sieć hoteli „Hotels Worldwide” udostępnia wiele różnych usług Web Services. W pewnym momencie odkryto, że jedna z tych usług jest wrażliwa na ataki typu SQL Injection. Problem jest natychmiast opisywany w języku AVDL i publikowany. Firewall przedsiębiorstwa korzystając z tych informacji może automatycznie zablokować niebezpieczną usługę do czasu usunięcia dziury.

5. Podsumowanie

Proces budowy standardów bezpieczeństwa dla usług Web Services jest ciągle jeszcze nie zakończony. Świadczą o tym sprzeczne i częściowo nakładające się standardy i specyfikacje obecne na rynku. W takiej sytuacji trudno przewidzieć, które standardy się przyjmą w przyszłości. Wpływ mają na to przede wszystkim dwa czynniki. Pierwszym z nich jest liczba dostępnych implementacji poszczególnych specyfikacji na różnych platformach programistycznych. Obecnie standard WS-Security, wraz ze związanymi z nim XML-Signature i XML-Encryption są implementowane między innymi przez: Web Services Toolkit IBMa (WSTK 3.1), Java Web Services Developer Pack, czy Web Services Enhancements 3.0, które jest dodatkiem do Visual Studio 2005 i platformy .NET 2.0. Istnieją również ściany ogniowe XML, które implementują część standardów opisanych w niniejszej pracy. Wśród przykładowych programów tego typu można wymienić: XML Message Server firmy Westbridge Technology, SOAP Content Inspector firmy Quadraxis, jak również produkty firm Vordel, Reactivity, Forum Systems czy Flamenco Networks [44]. Drugim czynnikiem jest licencja na wykorzystywanie tychże standardów. Standardy W3C muszą być darmowe do wykorzystania, co zdecydowanie przemawia za ich wykorzystaniem. Polityka OASIS jest nieco mniej restrykcyjna, gdyż specyfikacje przesłane do tej organizacji mogą być opatentowane, co jest najprawdopodobniej jedną z przyczyn, dla której specyfikacje opracowane przez IBM i Microsoft są zgłaszane właśnie do OASIS [8]. Opatentowanie niektórych standardów może niestety poważnie ograniczyć rozwój rozwiązań opartych o usługi Web Services. Na razie pozostaje jednak jedynie czekać i obserwować dalszy rozwój standardów i specyfikacji przedstawionych w niniejszej publikacji.

Bibliografia

1. Graham S., Simeonov S., Boubez T., Davis D., Daniels G., et al. Java. Usługi WWW. Vademecum profesjonalisty. *Helion*, ISBN-83-7197-991-6, 2003.
2. Public Key Infrastructure, Wikipedia, http://en.wikipedia.org/wiki/Public_key_infrastructure
3. Public Key Certificate, Wikipedia, http://en.wikipedia.org/wiki/Public_key_certificate

4. Certificate Authority, Wikipedia, http://en.wikipedia.org/wiki/Certificate_authority
5. Nadalin A., Kaler C., Hallam-Baker P., Monzillo Ronald., eds., „OASIS Web Services Security: SOAP Message Security 1.0”, OASIS Standard Specification, <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf>, marzec 2004.
6. Franks J., Hallam-Baker P., Hostetler J., Lawrence S. et al., „RFC 2617: HTTP Authentication: Basic and Digest Access Authentication”, <http://www.ietf.org/rfc/rfc2617.txt>, czerwiec 1999.
7. Dierks T., Allen C., „RFC 2246: The TLS Protocol version 1.0”, <http://www.ietf.org/rfc/rfc2246.txt>, styczeń 1999.
8. Koch C., „The Battle for Web Services”, CIO Magazine, <http://www.cio.com/archive/100103/standards.html>, 1 październik 2003.
9. Siddiqui B., „Web Services Security, Part 1”, <http://webservices.xml.com/pub/a/ws/2003/03/04/security.html>, 4 marzec 2003.
10. Siddiqui B., „Web Services Security, Part 2”, <http://webservices.xml.com/pub/a/ws/2003/04/01/security.html>, 1 kwiecień 2003.
11. Siddiqui B., „Web Services Security, Part 3”, <http://webservices.xml.com/pub/a/ws/2003/05/13/security.html>, 13 maj 2003.
12. Siddiqui B., „Web Services Security, Part 4”, <http://webservices.xml.com/pub/a/ws/2003/07/22/security.html>, 22 lipiec 2003.
13. Eastlake D., Reagle J., Solo D., eds., „XML Signature Syntax and Processing”, W3C Recommendation, <http://www.w3.org/TR/xmlsig-core/>, 12 luty 2002.
14. Eastlake D., Reagle J., eds., „XML Encryption Syntax and Processing”. W3C Recommendation, <http://www.w3.org/TR/xmlenc-core/>, 10 grudzień 2002.
15. Nadalin A., Kaler C., Monzillo Ronald., Hallam-Baker P., eds., „OASIS Web Services Security: SOAP Message Security 1.1”, OASIS Standard Specification, <http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>, 1 luty 2006.
16. Hallam B., Maler E., eds., „Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML)”, OASIS Standard Specification, <http://www.oasis-open.org/committees/download.php/2290/oasis-sstc-saml-1.0.zip>, 5 listopad 2002.
17. Maler E., Mishra P., Philpot R., eds., „Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V1.1”, OASIS Standard Specification, <http://www.oasis-open.org/committees/download.php/3406/oasis-sstc-saml-core-1.1.pdf>, 2 wrzesień 2003.
18. Cantor S., Kemp J., Philpot R., Maler E., eds., „Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0”, OASIS Standard Specification, <http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf>, 15 marzec 2005.
19. Lockhart H., „Demystifying Security Standards”, http://dev2dev.bea.com/pub/a/2005/10/security_standards.html, 11 październik 2005.
20. Godik S., Moses T., eds., „OASIS eXtensible Access Control Markup Language (XACML) Version 1.0”, OASIS Standard Specification, <http://www.oasis-open.org/committees/download.php/2406/oasis-xacml-1.0.pdf>, 18 luty 2003.
21. Godik S., Moses T., eds., „OASIS eXtensible Access Control Markup Language (XACML) Version 1.1”, OASIS Committee Specification, <http://www.oasis-open.org/committees/xacml/repository/cs-xacml-specification-1.1.pdf>, 7 sierpień 2003.
22. Moses T., ed., „OASIS eXtensible Access Control Markup Language (XACML) Version 1.1”, OASIS Standard Specification, http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf, 1 luty 2005.
23. „XrML 2.0 Technical Overview”, <http://www.xrml.org/reference/XrMLTechnicalOverviewV1.pdf>, 8 marzec 2002.
24. Rosenberg R., „Trust, Access Control, and Rights for Web Services Part 2”, <http://www.devshed.com/c/a/Security/Trust-Access-Control-and-Rights-for-Web-Services-Part-2/>, 12 październik 2004.

25. Anderson S., Bohren J., et al., „Web Services Trust Language (WS-Trust)”, <ftp://www6.software.ibm.com/software/developer/library/ws-trust.pdf>, luty 2005.
26. Rosenberg R., „Trust, Access Control, and Rights for Web Services Part 1”, <http://www.devshed.com/c/a/Security/Trust-Access-Control-and-Rights-for-Web-Services-Part-1/>, 26 lipiec 2004.
27. „Security in a Web Services World: A Proposed Architecture and Roadmap”, <ftp://www6.software.ibm.com/software/developer/library/ws-secmap.pdf>, 7 kwiecień 2002.
28. Anderson S., Bohren J., et al., „Web Services Secure Conversation Language (WS-SecureConversation)”, <ftp://www6.software.ibm.com/software/developer/library/ws-secureconversation.pdf>, luty 2005.
29. OASIS Web Services Secure Exchange (WS-SX) Technical Committee, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ws-sx
30. Bajaj S., Della-Libera G., et al., „Web Services Federation Language (WS-Federation)”, <ftp://www6.software.ibm.com/software/developer/library/ws-fed.pdf>, 8 lipiec 2003.
31. Liberty Alliance ID-WSF 2.0 Specifications Overview, <http://xml.coverpages.org/ni2005-02-11-b.html#LA20-Specs>
32. Angal R., Narayaan S., Patterson P., Simhachalam M., „Building Identity-Enables Web Services”, <http://developers.sun.com/prodtech/identserver/reference/techart/id-enabled-ws.html>, 18 październik 2005.
33. „Liberty Alliance & WS-Federation: A Comparative Overview”, Liberty Alliance Project White Paper, <https://www.projectliberty.org/resources/whitepapers/wsfed-liberty-overview-10-13-03.pdf>, 14 październik 2003.
34. Drees S., „OASIS Digital Signature Service Core Protocols, Elements, and Bindings”, 3rd OASIS Committee Draft, <http://docs.oasis-open.org/dss/v1.0/dss-v1.0-spec-cd-Core-r03.pdf>, 29 listopad 2005.
35. Ford W., et al., „XML Key Management Specification (XKMS)”, W3C Note, <http://www.w3.org/TR/xkms/>, 20 marzec 2001.
36. Bajaj S., et al., „Web Services Policy Framework (WS-Policy)”, <http://download.boulder.ibm.com/ibmdl/pub/software/dw/specs/ws-polfram/ws-policy-2006-03-01.pdf>, marzec 2006.
37. Box D., et al., „Web Services Policy Assertions Language (WS-PolicyAssertions)”, <ftp://www6.software.ibm.com/software/developer/library/ws-polas.pdf>, 18 grudzień 2002.
38. Della-Libera G., et al., „Web Services Security Policy Language (WS-SecurityPolicy)”, <ftp://www6.software.ibm.com/software/developer/library/ws-secpol.pdf>, lipiec 2005.
39. Bajaj S., et al., „Web Services Policy Attachment (WS-PolicyAttachment)”, <http://download.boulder.ibm.com/ibmdl/pub/software/dw/specs/ws-polatt/ws-polatt-2006-03-01.pdf>, marzec 2006.
40. Moses T., ed., „OASIS XACML profile for Web-Services”, OASIS Working draft, <http://www.oasis-open.org/committees/download.php/3661/draft-xacml-wspl-04.pdf>, 29 wrzesień 2003.
41. Anderson A., „An Introduction to the Web Services Policy Language (WSPL)”, w materiałach konferencyjnych 5th IEEE International Workshop on Policies for Distributed Systems and Networks, Yorktown Heights, New York, <http://research.sun.com/projects/xacml/Policy2004.pdf>, 2004
42. Larmouth J., ed., „OASIS XML Common Biometric Format”, OASIS Standard Specification, <http://www.oasis-open.org/committees/download.php/3353/oasis-200305-xcbf-specification-1.1.doc>, sierpień 2003.
43. Bialkowski J., Heineman K., „OASIS Application Vulnerability Description Language v1.0”, Oasis Standard, <http://www.oasis-open.org/committees/download.php/7145/AVDL%20Specification%20V1.pdf>, maj 2004.
44. Gralla P., „THE WEB SERVICES ADVISOR: XML Firewalls”, http://searchwebservices.techtarget.com/tip/1,289483,sid26_gci855052,00.html, 8 październik 2002