Universität Stuttgart

Institut für Parallele und Verteilte Systeme
**Anwendersoftware**

Institut für Architektur von Anwendungssystemen

# Authentifizierung und Autorisierung bei Web Services

**Seminararbeit**

Studienprojekt SIMPL (SS 2009)

Betreuer: Michael Reiter

**Michael Schneidt**

Stuttgart, 26. Juni 2009

# Inhaltsverzeichnis

# Authentifizierung und Autorisierung von Web Services

Michael Schneidt

**Zusammenfassung.** In BPEL, einer auf XML basierenden Sprache zur Beschreibung von Geschäftsprozessen, werden Aktivitäten als Web Services implementiert. Web Services bieten neben der losen Kopplung in einer dienst-orientierten Architektur den Vorteil, dass bereits bestehende Sicherheitsstandards genutzt werden können. Diese Ausarbeitung spezialisiert sich auf die Authentifizierung und Autorisierung von Zugriffen auf Web Services und bietet einen Überblick über nötige und mögliche Sicherheitsmaßnahmen und eine Einführung in die wichtigsten Sicherheitsstandards. Zuletzt wird skizziert, wie diese im Kontext von BPEL Engines eingesetzt werden können.

## 1 Einleitung

Mit dem Studienprojekt SIMPL (Simtech: Information Management, Processes and Languages) soll ein Framework für den Datenzugriff aus BPEL geschaffen werden. Der Zugriff auf die Daten und andere Aktivitäten werden dabei über Web Services realisiert, die vor fremden Zugriffen geschützt werden müssen. Im Folgenden wird zunächst ein Überblick über die allgemeinen Sicherheitsanforderungen und die möglichen Sicherheitsmaßnahmen gegeben. Anschließend wird im Detail auf die Authentifizierung und Autorisierung und die jeweiligen XML Standards eingegangen. Des Weiteren folgt eine Vorstellung der für das Thema relevanten Standards speziell für Web Services. Ein grober Ausblick, wie diese Sicherheitsstandards im Kontext von BPEL Engines genutzt werden können, schließt diese Ausarbeitung ab.

Es werden die Themen „SOA und Web Services“ und „Der WS-Standard-Stack und BPEL“ vorausgesetzt, die ebenfalls im Rahmen des Seminars behandelt werden.

## 2 Grundlagen

Bei der Kommunikation mit einem Web Service gibt es verschiedene Sicherheitsanforderungen und Ziele, die im Folgenden erläutert werden.

## 2.1 Kommunikation mit Web Services

Die Kommunikation mit Web Services erfolgt in der Regel über das SOAP Protokoll, welches auf XML basiert und ein standardisiertes Nachrichtenformat bietet. Für die eigentliche Nachrichtenübertragung bedient sich SOAP aber anderer Standard-Protokolle wie zum Beispiel HTTP oder FTP. In der Praxis kommt meist HTTP zum Einsatz, da andere Protokolle oftmals von Netzwerk-Firewalls gefiltert werden. Eine SOAP Nachricht bietet neben den eigentlichen Nutzdaten auch Platz für beliebige Metadaten, in denen unter anderem auch sicherheitsrelevante Informationen mitgesendet werden können.

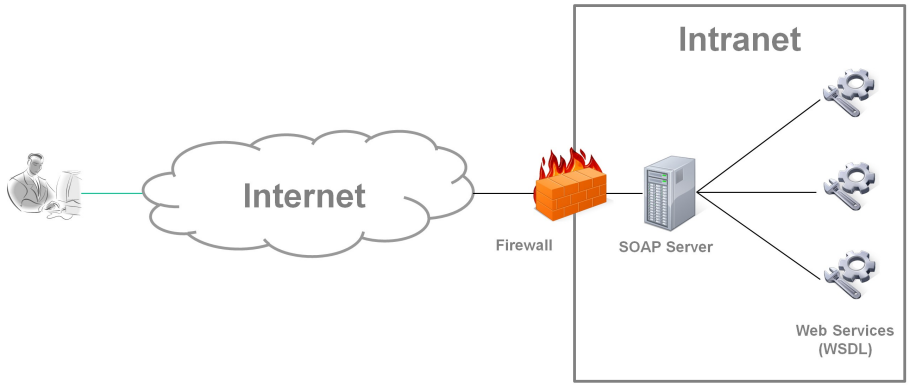


**Abbildung 1.** Kommunikation mit Web Services

## 2.2 Sicherheitsanforderungen

Wenn auf einen Web Service zugegriffen wird, ist es für das System entscheidend zu wissen von welchem Benutzer der Zugriff kommt und ob derjenige überhaupt berechtigt ist die gewünschte Funktion in Anspruch zu nehmen. Genauso ist es aber auch für den Benutzer entscheidend zu wissen, dass er tatsächlich mit dem gewünschten Service kommuniziert. Außerdem ist es bei sicherheitskritischen Anwendungen wichtig, dass Zugriffe eindeutig nachgewiesen werden können.

Zusätzlich muss sichergestellt werden, dass die übertragenen Daten nicht von Dritten eingesehen werden können, und Nachrichten unterwegs nicht manipuliert worden sind. Zu diesem Zweck gibt es verschiedene Sicherheitsdienste, die als Schutzvorrichtungen diese Sicherheitsanforderungen erfüllen sollen.

**Authentifizierung.** Die Authentifizierung stellt vor einer Datenübertragung sicher, dass sich beide Seiten gegenseitig ausgewiesen haben und stellt somit ein Vertrauensverhältnis her. Basierend auf diesem Vertrauensverhältnis können anschließend Daten ausgetauscht werden. Da mit einer Authentifizierung erheblicher Aufwand verbunden ist, und in einer service-orientierten Architektur (SOA) mit vielen Web Services oftmals viele Zugriffe nacheinander stattfinden, möchte man zusätzlich erreichen, dass eine Authentifizierung nicht bei jedem einzelnen Zugriff notwendig ist. Im optimalen Fall muss sich ein Benutzer nur einmal authentifizieren und hat sich damit bei allen Web Services ausgewiesen. Dieses Prinzip nennt man auch Single Sign On (SSO).

**Autorisierung.** Die Autorisierung findet gewöhnlich nach einer erfolgreichen Authentifizierung statt und kontrolliert den eigentlichen Zugriff des Benutzers. Wenn beispielsweise der Benutzer ein Administrator ist, soll dieser zusätzliche Zugriffsrechte haben, die einem normalen Benutzer auf jeden Fall verwehrt bleiben sollten. Ein wichtiges Ziel bei der Autorisierung ist die Organisation der Zugriffsrechte, so möchte man die Benutzer beispielsweise in Gruppen einteilen oder eine Hierarchie an Zugriffsrechten bilden. Denkbar sind auch Sicherheitsstufen, wie im militärischen Bereich, wo sichergestellt werden muss, dass keine Zugriffe auf höhere Stufen stattfinden können.

**Integrität und Vertraulichkeit.** Die Integrität und Vertraulichkeit beschäftigen sich mit der Absicherung der eigentlichen Datenübertragung. Dabei geht es um Maßnahmen die eine Manipulation der Daten bei der Übertragung verhindern bzw. auf jeden Fall erkennen lassen und die Einsicht der Daten von Dritten verhindern.

### 2.3 Sicherheitsgrundlagen

Sicherheitsmaßnahmen können bei Web Services auf verschiedenen Ebenen ansetzen und müssen den gesamten Kommunikationsweg berücksichtigen. Über das Internet ist der Kommunikationsweg heute keine Direktverbindung mehr von einem Punkt zu einem anderen, sondern es gibt viele Zwischenstationen die bei dem Thema Sicherheit als potentielle Angriffspunkte berücksichtigt werden müssen.

**Punkt-zu-Punkt-Sicherheit.** Eine Punkt-zu-Punkt-Sicherheit ist nur zwischen zwei Punkten die direkt miteinander verbunden sind gewährleistet. Kommen verschlüsselte Daten bei der Übertragung an Zwischenstellen (Itermediaries) vorbei, sind diese Daten dort im Klartext einsehbar und nicht geschützt, sie werden dort entschlüsselt und für die weitere Übertragung erneut verschlüsselt.

**Ende-zu-Ende-Sicherheit.** Eine Ende-zu-Ende-Sicherheit gewährleistet eine Absicherung der Daten bei der gesamten Datenübertragung von einem End-Benutzer zu einem anderen End-Benutzer Zwischenstellen haben dabei keine Einsicht.

**Sicherheitsebenen.** Bei Web Services gibt es verschiedene Ebenen auf denen diese Schutzvorrichtungen ansetzen können.

*Transportebene.* Die unterste Ebene ist die Transport- bzw. Netzwerkebene, bei der die einzelnen übertragenen Pakete kontrolliert werden können. Dies wird in der Praxis mit Netzwerk-Firewalls erreicht, die den Datenverkehr abfangen und die Zugriffe regeln. Üblicherweise erfolgt die Datenübertragung auf dieser Ebene mit TCP/IP und die Authentifizierung und Autorisierung kann nur über IP-Adressen und Ports erfolgen. Allerdings ist dies sehr unflexibel und es ist keine eindeutige Authentifizierung möglich, da hinter einer IP-Adresse beispielsweise in ganzes LAN stehen kann. Um die Vertraulichkeit und Authentizität auf dieser Ebene zu gewährleisten wird zum Beispiel SSL als Verschlüsselungsverfahren eingesetzt. SSL gewährleistet aber nur eine Punkt-zu-Punkt-Sicherheit und schreibt keine beidseitige Authentifizierung vor.

*Meldungsebene.* Die nächst-höhere Ebene ist die Meldungs- bzw. Nachrichtenebene, in unserem Fall z.B. das bereits erwähnte SOAP-Protokoll. Da SOAP von sich aus keinerlei Sicherheitsmechanismen anbietet, wurde der WS-Security Standard entwickelt, der beschreibt, wie über SOAP eine sichere Datenübertragung möglich ist und wie Sicherheitsinformationen übertragen werden können.

*Anwendungsebene.* Die höchste Ebene ist die Anwendungsebene, bei der XML-Dokumente als Nachrichten zwischen Anwendungen verschickt und ausgewertet werden. Hier können beispielsweise die Nutzdaten direkt beim Endpunkt verschlüsselt und entschlüsselt werden. So kann eine Ende-zu-Ende-Sicherheit erreicht werden.

**Sicherheitstaxonomie.** Als Übersicht über die verschiedenen Sicherheitsdienste, -mechanismen und -standards soll folgende Taxonomie in Abbildung 2 dienen. Auf die rot hervorgehobenen Sicherheitsmechanismen und -standards wird im weiteren Verlauf dieser Ausarbeitung im Detail eingegangen.

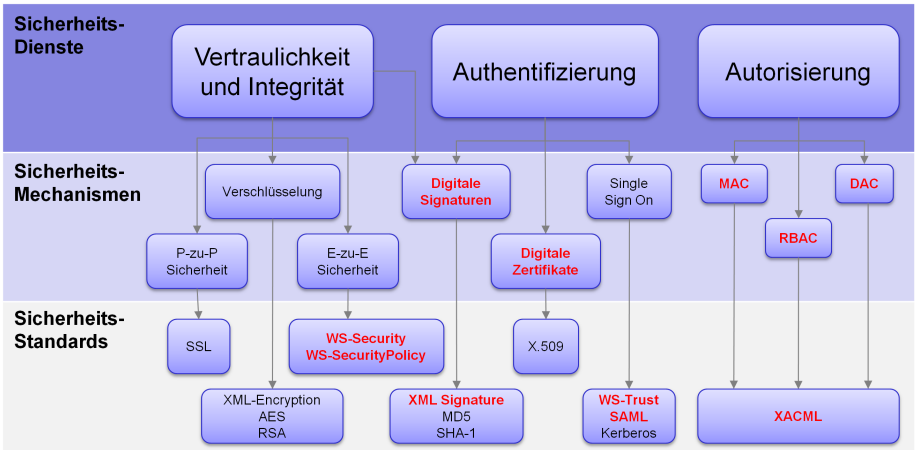


**Abbildung 2.** Sicherheitstaxonomie (vgl. [4] Abb. 2.3)

# 3 Authentifizierung

## 3.1 Signaturen

Eine Signatur ist ein kryptographisches Verfahren, das die Integrität sowie Authentizität einer Nachricht sicherstellt. Es basiert in der Regel auf einem asymmetrischen Kryptosystem bei dem beide Seiten ein Schlüsselpaar aus einem geheimen privaten und einem öffentlichen Schlüssel besitzen.

Die Nachricht eines Senders wird mit dessen geheimen privaten Schlüssel signiert und kann vom dem Empfänger über den bekannten öffentlichen Schlüssel authentifiziert werden. Die Anwendung des öffentlichen Schlüssels erfolgt dabei meist auf eine Prüfsumme (Digest) fester Länge, die über die Nachricht gebildet wird. Da die Nachricht selbst sehr groß sein kann und bei der Anwendung entsprechend viel Ressourcen benötigt würden. In Abbildung 3 wird der Ablauf bei einem solchen Signaturverfahren dargestellt.

Zunächst wird beim Sender über eine festgelegte Hashfunktion (z.B. SHA-1) eine Prüfsumme berechnet, auf die anschließend der private Schlüssel angewendet wird wodurch eine Signatur erzeugt wird. Anschließend wird die Signatur an die Nachricht angehängt und beides versendet. Der Empfänger berechnet nun seinerseits die Prüfsumme über den Klartext, zusätzlich stellt er mit dem öffentlichen Schlüssel des Empfängers die Prüfsumme mit der die Signatur erstellt wurde wieder her. Nur wenn beide Prüfsummen identisch sind, handelt es sich um die unveränderte ursprüngliche Nachricht. Dadurch, dass es zu dem öffentlichen Schlüssel nur einen privaten Schlüssel gibt der dem Sender gehört, authentifiziert sich der Sender gegenüber dem Empfänger.

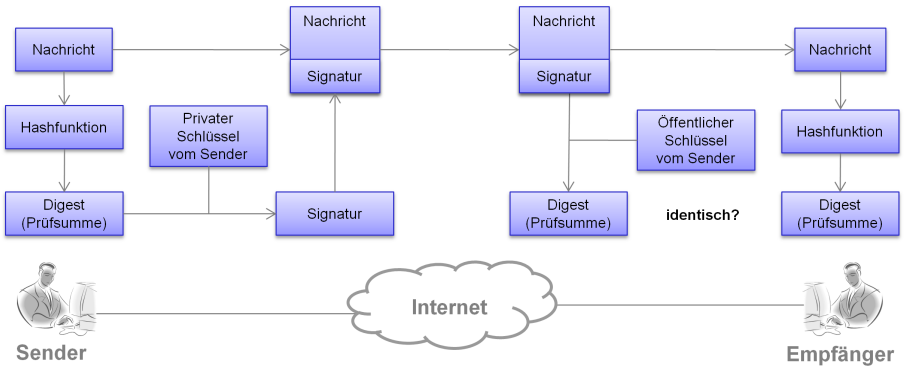


**Abbildung 3.** Signaturverfahren (vgl. [7] S.15)

### 3.2 XML Signature

Für die Signierung von XML-Dokumenten gibt es den XML Signature-Standard, der es ermöglicht beliebige Daten eines XML-Dokuments zu signieren. XML Signature bietet verschiedene Arten der Signierung und ist daher nicht nur auf XML-Dokumente beschränkt.

**Arten der Signierung.** XML Signature bietet drei verschiedene Arten der Signierung, wie in Abbildung 4 dargestellt.

*Enveloped.* Ein XML Dokument enthält die Signatur als Element, die Signatur bezieht sich dabei auf das gesamte Dokument.

*Enveloping.* Die Signatur umschließt bzw. beinhaltet das gesamte XML-Dokument, sie enthält jedoch einen Verweis auf das XML-Dokument, mit dem auch nur auf ein bestimmtes Element des Dokuments verwiesen werden kann, für das die Signatur gültig ist. Die Daten müssen dabei nicht zwangsweise XML-Daten sein, es können auch beliebige andere Dateiformate umschlossen werden.

*Detached.* Hier werden XML-Dokument und Signatur getrennt geführt, die Signatur enthält lediglich eine Referenz (URI) auf das XML-Dokument. Wenn die Referenz nicht mehr gültig ist, also das XML Dokument nicht mehr erreichbar ist, verliert die Signatur auch ihre Gültigkeit.

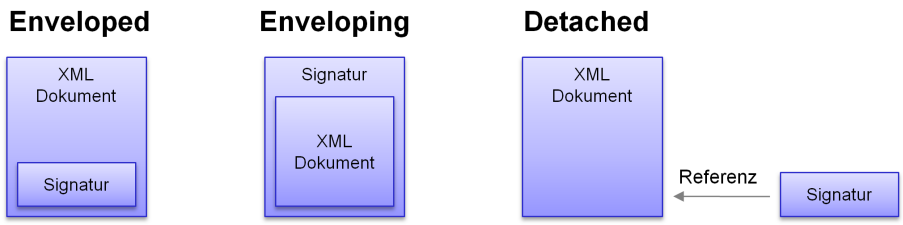


**Abbildung 4.** XML Signature

**Kanonisierung.** Damit die Signierung richtig funktioniert, muss die Berechnung der Prüfsumme auf beiden Seiten, bei gleichen Daten, das gleiche Ergebnis liefern. Aber in der Natur von XML-Dokumenten liegt ihre Interoperabilität, so dass diese bei unterschiedlichen Anwendungen auf unterschiedlichen IT-Systemen Station machen und dabei physischen Änderungen unterliegen können, die den eigentlichen Inhalt nicht verändern. Probleme sind beispielsweise ein unterschiedlicher Zeichensatz, die Reihenfolge von Elementen, oder zusätzlicher Whitespace, so dass die Integritätsprüfung fehlschlägt obwohl sich an den Nutzdaten nichts geändert hat. Aus diesem Grund werden XML Dokumente vor der Signierung zuerst in eine kanonisierte, einheitliche Form gebracht, die sicherstellt, dass solche Probleme nicht auftreten können.

### 3.3 Zertifikate

Mit Signaturen ist es also möglich den Sender einer Nachricht eindeutig zu identifizieren. Aber dies geht nur dann, wenn sichergestellt ist, dass der verwendete öffentliche Schlüssel auch tatsächlich dem Sender gehört und nicht einer dritten Person.

Dies wird erst mit Zertifikaten erreicht, die von vertrauenswürdigen Zertifizierungsstellen (z.B. VeriSign, GlobalSign, ...) ausgegeben werden. Ein Zertifikat beinhaltet den öffentlichen Schlüssel und Informationen über den Eigentümer. Um ein solches Zertifikat zu erlangen, muss man sich mit mehr oder weniger hohen Auflagen bei der Zertifizierungsstelle ausweisen, beispielsweise durch das Vorzeigen eines Personalausweises. Anschließend bekommt man ein Schlüsselpaar zugeschickt und ein Zertifikat mit dem die Nachrichten signiert werden können..

Zur Sicherheit wird den Zertifikaten eine Gültigkeitsdauer mitgegeben. Abgelaufene oder gesperrte Zertifikate werden durch Zertifikatsperrlisten (CRL - Certificate Revocation Lists) identifiziert.

Die Zertifizierungsstellen gehören zu einer sogenannten Public-Key-Infrastructure (PKI), die es ermöglicht Zertifikate auszustellen, zu verteilen und zu prüfen. Der bekannteste Standard für eine solche PKI ist X.509.

### 3.4 SAML

Ein wichtiger Standard bei der Authentifizierung ist die Security Assertion Markup Language (SAML). Sie ermöglicht es sicherheitsrelevante Informationen in XML zu formulieren und bietet Vorschriften für die Übertragung dieser Informationen. Diese Informationen können Angaben über eine erfolgte Authentifizierung sein oder auch Benutzerinformationen, wie Rollen, die später bei der Autorisierung ausgewertet werden können.

Mit SAML wird die Authentifizierung vom eigentlichen Service Anbieter, dem Service Provider, getrennt. Die Authentifizierung läuft dabei über einen Identity Provider, mit dem der Service Provider über in SAML definierte Protokolle kommunizieren kann. Dadurch kann der Service Provider Aufwand und Kosten sparen und sich auf seine eigentliche Aufgabe konzentrieren.

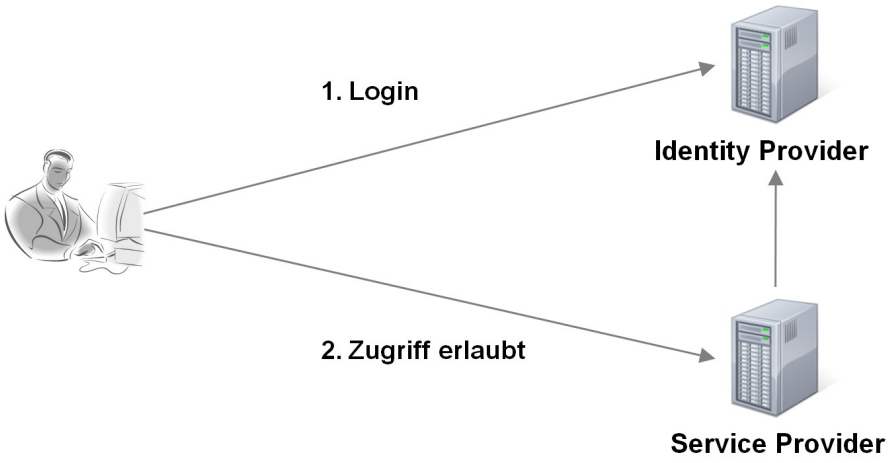


**Abbildung 5.** Identity Provider und Service Provider (vgl. [13] S.3)

Da die Sicherheitsinformationen nicht mehr an Organisationsgrenzen gebunden sind und zwischen verschiedenen Systemen ausgetauscht werden können, ergeben sich unter anderem folgende Möglichkeiten.

**Single Sign On.** Mit SAML wird ein Single Sign On möglich, da nach einem ersten Login die folgenden Anfragen mit entsprechenden Informationen über eine zuvor erfolgte erfolgreiche Authentifizierung ausgestattet werden können.

**Identity Federation.** Durch die Möglichkeit der Übertragung von Identitäts-Informationen können sich verschiedene Systeme gegenseitig unterstützen. So kann beispielsweise ein System einem Partnersystem Zugriff gewähren, bei dem sich ein Benutzer zuvor authentifiziert hat. Das Partnersystem erkennt anhand einer (signierten) SAML Nachricht die erfolgte Authentifizierung des Benutzers und kann dessen Zugriffe über eigene Zugriffsregeln (Policies) steuern. Dieses Prinzip der verteilten Identitätsinformationen zwischen verschiedenen Systemen, bei dem die gesamten Sicherheitsinformationen nicht zentral lokalisiert sind, sondern sich ergänzen, bezeichnet man auch als föderierte (zusammengefasste) Identität oder Identity Federation.

### 3.5 SAML Komponenten

SAML bietet folgende aufeinander aufbauende Komponenten.

**Assertions.** Assertions sind sicherheitsrelevante Aussagen, die sich in drei verschiedene Typen aufteilen. Es können damit Informationen über eine erfolgte Authentifizierung (Authentication Assertion), eine erfolgte Autorisierung (Authorization Decision Assertion) und benutzerspezifische Informationen (Attribute Assertion) ausgedrückt werden.

**Protocols.** Protocols sind einfache Frage-Antwort-Protokolle, die die Kommunikation zwischen Identity Provider und Service Provider beschreiben. Wie bei den Assertions wird hier zwischen Fragen zur Authentifizierung, zur Autorisierung und zu benutzerspezifischen Daten unterschieden.

**Bindings.** Bindings beschreiben das Mapping der verwendeten Protocols auf Standard-Nachrichten-Protokolle. Bei Web Services ist dies üblicherweise das SAML SOAP Binding.

**Profiles.** Profile haben in SAML zwei Bedeutungen. Zum einen beschreiben sie die Kombination von Assertions, Protocols und Bindings für bestimmte Anwendungsfälle (z.B. Web SSO), zum anderen werden damit Vorschriften für die Abbildung von Attributen aus anderen Systemen, wie beispielsweise XACML Nachrichten, auf SAML Assertions beschrieben.

### 3.6 SAML Anwendungsfall

Abbildung 6 stellt einen Anwendungsfall für den Zugriff auf einen Web Service dar.

Der Benutzer möchte auf einen Web Service zugreifen und wendet sich für das Login an einen Identity Provider. Nach Prüfung der Identität stellt dieser eine Authentication Assertion aus und fügt ggf. weitere Benutzerinformationen wie Rollen als Attribute Assertions hinzu. Diese Informationen bilden zusammen ein SAML Token, das nun für den Zugriff auf die Ressource verwendet werden kann.

Der Zugriff auf den Service wird von einem Policy Enforcement Point (PEP) abgefangen, der sich mit dem Token an einen Policy Decision Point (PDP) wendet, um eine Entscheidung über den Zugriff zu treffen. Der PDP kann anhand der ihm bekannten Zugriffsberechtigungen (Policies) eine Entscheidung treffen ob der Zugriff genehmigt oder gesperrt wird. Diese Entscheidung fügt er anschließend in Form einer Authorization Decision Assertion dem Token hinzu, so dass beim erneuten Versuch mit dem Token auf den Service zuzugreifen, der PEP anhand der vom PDP angehängten Information unmittelbar selbst die Entscheidung übernehmen kann.

Durch die Wiederverwendung des Tokens bei einem weiteren Zugriff ist eine erneute Authentifizierung und Autorisierung nicht mehr notwendig.

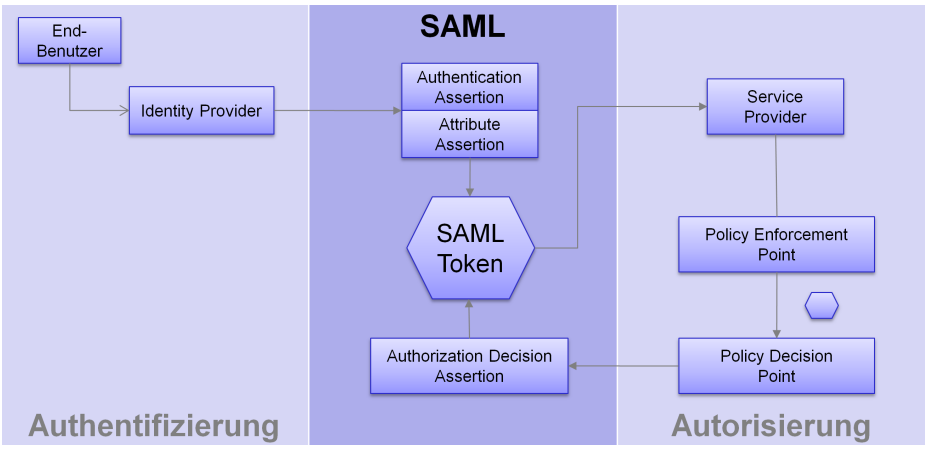


**Abbildung 6.** SAML Anwendungsfall (vgl. [14] S.5)

# 4 Autorisierung

## 4.1 Zugriffskontrolle

Bei der Kontrolle von Zugriffen gibt es verschiedene Modelle, die jeweils kurz vorgestellt werden.

**Discretionary Access Control (DAC).** DAC ist eine rein Identitätsbasierte Zugriffskontrolle, bei dem die Zugriffsrechte einer Relation von Subjekt, Objekt und Recht entsprechen. Ein Sicherheitsproblem bei dieser Art der Zugriffskontrolle, ist die Erweiterung von Benutzerrechten für eingeschränkte Operationen, da dazu kurzfristig andere Rollen eingenommen werden müssen, die unter Umständen ausgenutzt werden können.

**Mandatory Access Control (MAC).** MAC benutzt neben der Identität zusätzlich Regeln und Eigenschaften als Bedingungen. Die Durchsetzung der Zugriffsregeln geschieht im Unterschied zu DAC und auch RBAC direkt vom IT-System oder der Anwendung. Damit können Multi-Level-Sicherheitssysteme mit vertikalen Schutzstufen und Multi-Laterale-Sicherheitssysteme realisiert werden, die zusätzlich eine horizontale Erweiterung ermöglichen.

**Role Based Access Control (RBAC).** Bei der rollenbasierten Zugriffsregelung werden den Benutzern bestimmte Rollen und Gruppen zugeteilt. Ein Benutzer kann mehrere Rollen haben und verschiedenen Gruppen angehören. Für die Zugriffskontrolle werden in der Regel Systeme zum Identitätsmanagement verwendet, die auch über mehrere Systeme hinweg für eine Konsistenz der Zugriffsregelung sorgen können.

### 4.2 XACML

Die eXtensible Access Control Markup Language ist ein XML-Standard zur Formulierung und Auswertung von Zugriffsberechtigungen (Policies). XACML wurde im Zusammenhang mit SAML entwickelt und ergänzt SAML um die Definition und Auswertung von Zugriffsregeln. Mit XACML lassen sich alle Zugriffsarten realisieren, dazu stehen folgende Elemente zur Verfügung.

Die elementaren Elemente von XACML sind Rule, Policy und PolicySet. Eine Rule ist das Basiselement einer Policy und kann nicht isoliert als Nachricht verschickt werden sondern nur innerhalb einer Policy. Während eine Policy eine Ansammlung von Regeln vereint, die es Auszuwerten gilt, ist es mit einer PolicySet möglich mehrere Policies zu vereinen. Da außerdem vom Aufbau her kein Unterschied besteht, wird im Folgenden nur auf die Policy im Detail eingegangen.

**Policy.** Eine XACML Policy besteht aus einem Ziel (Target), mehreren Regeln (Rules) und einer Angabe zu einem Auswertungs-Algorithmus, dazu kommen weitere Verpflichtungen (Obligations), die mit einer positiven Auswertung ausgeführt werden müssen.

*Target.* Das Target gibt das Ziel der Policy an, also auf welche Zugriffe sich die Policy bezieht bzw. für welche Zugriffe sie zuständig ist. Dies ist eine Kombination von Subject, Attribute und Action. Das Subject ist die Einheit von der ein Zugriff ausgeht bei dem eine Action auf einer bestimmten Resource ausgeführt werden soll.

*Rules.* Regeln bestehen aus einem Target, mehreren Conditions und einem Effect. Damit lassen sich die Bedingungen zu Attributen formulieren. Das Target entspricht dem Ziel einer Policy und bestimmt die Relevanz der Regeln für Zugriffe. Conditions sind Bedingungen zu Attributen, die zu True, False und Indeterminate (Unbestimmt) ausgewertet werden. Der Effect definiert schließlich die Auswirkung der positiven Auswertung einer Regel und liefert entweder Permit oder Deny.

*Rule Combining Algorithm.* Der Auswertungsalgorithmus bestimmt, wie eine finale Entscheidung anhand der Auswertung der Regeln erfolgen soll. So kann beispielsweise mit dem Deny-overrides-Algorithmus bestimmt werden, dass eine Policy zu Deny ausgewertet wird, sobald nur eine der Regeln zu Deny ausgewertet wurde.

*Obligations*. Obligations sind zusätzliche Verpflichtungen, die von einem PEP erfüllt werden müssen, wenn eine Policy zu Permit ausgewertet wird. Dies ermöglicht eine noch feinere Zugriffskontrolle.

### 4.3 XACML Anwendungsfall

Im folgenden Anwendungsfall in Abbildung 7 wird das Konzept von XACML vorgestellt. Die hervorgehobenen Elemente zeigen die Gemeinsamkeit mit SAML, zusätzlich gibt es bei XACML Elemente wie den Policy Administration Point (PAP) und den Policy Information Point (PIP), die für die Erstellung von Policies und die Verwaltung der Attribute zuständig sind.

Wie bei SAML erfolgt ein Zugriff über einen Policy Enforcement Point (PEP), der die Entscheidung an einen Policy Decision Point (PDP) übergibt und auf eine Antwort wartet. Damit der PDP eine Entscheidung fällen kann, muss überprüft werden welche Policies für den Zugriff relevant sind. Dazu wendet sich der PDP an einen PAP, der für die Erstellung und Verwaltung von Policies zuständig ist. Für die Auswertung der Regeln der Policies, kann der PDP auf einen PIP zurückgreifen, der die nötigen Informationen zu den Attributen bereitstellt.

Abschließend wird die Entscheidung zusammen mit den Verpflichtungen (Obligations) der Policies an den PEP übergeben.

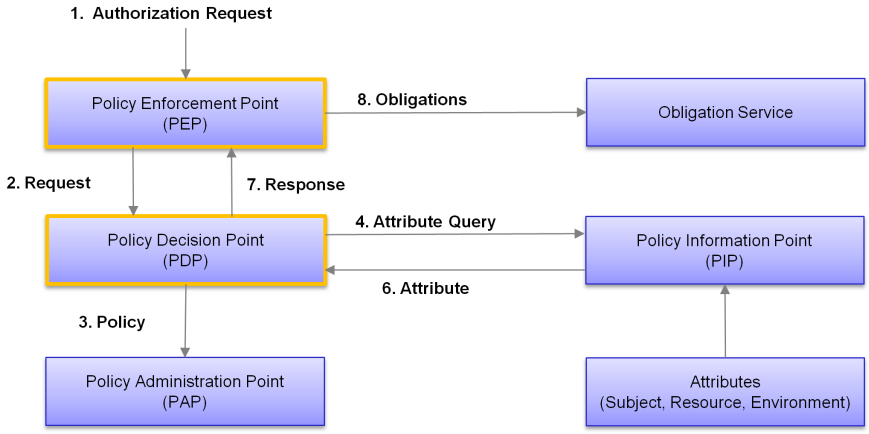


**Abbildung 7.** XACML Anwendungsfall (vgl. [9])

# 5 Web Service Standards

Nachdem nun die wichtigsten allgemeinen Sicherheitsstandards vorgestellt wurden, beschäftigt sich dieses Kapitel speziell mit Standards die im Rahmen von Web Services entstanden sind und deren Anwendungsgebiet nicht durch SOAP und der Web Service Decription Language (WSDL) abgedeckt werden. Dabei wird nur auf die Standards eingegangen, die für die Sicherheit bzw. für die Authentifizierung und Autorisierung relevant sind.

## 5.1 WS-Security

Mit WS-Security wird Vertraulichkeit, Integrität und Authentizität einer SOAP Nachricht gewährleistet. WS-Security beschreibt wie Nachrichten entsprechend abgesichert über SOAP verschickt werden können, dazu werden XML-Signature (siehe Kapitel 3.2) und XML-Encryption (siehe Abbildung 1) verwendet.

Des Weiteren beschreibt WS-Security die Möglichkeit Security Tokens, wie zum Beispiel ein X.509 Zertifikat oder ein SAML Token, in eine SOAP Nachricht einzubetten und zu übertragen.

## 5.2 WS-SecurityPolicy

WS-Policy liefert einen Standard, mit dem es allgemein möglich ist Anforderungen und Einschränkungen von Web Services zu beschreiben und nach außen bekannt zu machen. WS-SecurityPolicy liefert dazu als Ergänzung die Möglichkeit spezielle Sicherheitsanforderungen zu formulieren.

## 5.3 WS-Trust

WS-Trust ist ein Standard für das Identitätsmanagement und definiert das Konzept eines Security Token Service, der zur Herausgabe, Erneuerung und Validierung von Security Tokens zuständig ist. Ferner wird die Anfrage und Weiterleitung von Security Tokens beschrieben, wodurch Vertrauensdomänen überbrückt werden können. Dabei werden verschiedene Systeme berücksichtigt und eine Konvertierung zwischen Security Tokens beschrieben.

# 6 Sicherheit in BPEL Engines

Die Kommunikation mit BPEL Prozessen wird über sogenannte Partnerlinks realisiert. Dem Partner und Prozess werden Rollen zugewiesen, die jeweils durch eine WSDL Schnittstelle definiert werden. Rollen die dem Prozess zugeordnet werden, müssen dabei vom BPEL Prozess implementiert werden, was durch BPEL Aktivitäten realisiert wird. Die Bereitstellung und Ausführung von solchen Prozess-Endpunkten wird in der

Regel nicht von der BPEL Engine erledigt, sondern von einer Middleware, die die Web Services bereitstellen und ausführen kann.

Eingehende Nachrichten müssen authentifiziert und autorisiert werden, ausgehende Nachrichten müssen entsprechend signiert werden.. Im Verarbeitungsprozess in Abbildung 8, werden die verschiedenen Ansatzpunkte für eine Authentifizierung und Autorisierung deutlich.

Die Authentifizierung und Autorisierung ist bereits vor der BPEL Engine über den Server bzw. die Middleware möglich, oder innerhalb der BPEL Engine. Die Möglichkeit des Servers ist sehr eingeschränkt, darum wird im Folgenden lediglich die Vor- und Nachteile der Implementierung in Middleware und BPEL Engine eingegangen. (vgl. [4] S. 27-58)

## 6.1 Verarbeitungsprozess

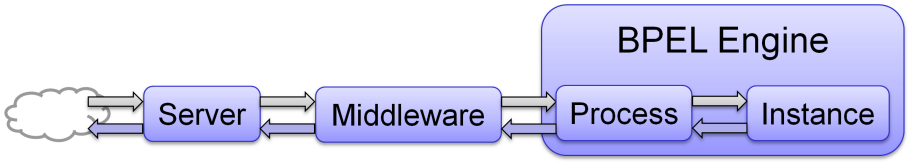


**Abbildung 8.** BPEL Verarbeitungsprozess (vgl. [4] Abb. 3.2)

Alle Anfragen landen zunächst beim Server. Wenn der Server für diese Art von Anfrage zuständig ist, leitet er die Nachricht an die Middleware weiter. Die Middleware kann den Endpunkt der Nachricht im Prozess ermitteln und die Nachricht an die BPEL Engine übergeben. In der BPEL Engine wird dann über die für die Nachricht zuständige Aktivität eine Prozess-Instanz ausgeführt.

Beispielsweise gibt ein Mitarbeiter eine Bestellung auf (eingehende Nachricht), die nach interner Abwicklung an den Lieferanten geschickt wird (ausgehende Nachricht).

## 6.2 Authentifizierung in der Instanz

**Eingehend.** Eingehende Nachrichten werden zum Server über ein sicheres Netzwerkprotokoll wie z.B. HTTPS übertragen, das in der Regel aber nur eine einseitige Authentifizierung bietet. Damit der Sender auch vom Server authentifiziert werden kann, müssen zusätzlich zur eingehenden Nachricht entsprechende Credentials des Senders hinzugefügt werden.

**Ausgehend.** Bei ausgehenden Nachrichten ist nur eine Authentifizierung des Senders gegenüber dem Empfänger nötig. Dies kann zum Beispiel durch die Signierung der ausgehenden Nachricht mit einem vertrauenswürdigen Zertifikat geschehen, aufgrund dessen der Empfänger dann den Sender authentifizieren kann. Um die Vertraulichkeit zu gewährleisten, kann die Nachricht zusätzlich mit dem öffentlichen Schlüssel des Empfängers verschlüsselt werden.

**Bewertung.** Die Identität eines Benutzers kann vom Prozess in einer Variable festgehalten und benutzt werden, zudem kann auf Fehlschläge bei der Authentifizierung direkt reagiert werden. Allerdings wird hierbei die Prozesslogik mit der Sicherheitslogik vermischt und es entsteht redundanter Code bei weiteren Authentifizierungen. Durch das Hinzufügen von Credentials werden zudem die Nutzdaten mit Sicherheitsinformationen vermischt. Durch die Nachrichtenübertragung über Sicherheitsprotokolle wie HTTPS, kann hier lediglich eine Punkt-zu-Punkt Sicherheit (siehe 2.3) erreicht werden.

### 6.3 Authentifizierung in der Middleware

**Eingehend.** Bei der Verwendung des SOAP Protokolls kann die Nachrichtenübertragung mit WS-Security (siehe 5.1) abgesichert und die Endpunkte, die hier in der Middleware liegen, mit WS-SecurityPolicy entsprechend konfiguriert werden. Die Überprüfung bei der Authentifizierung kann dabei über mehrere prozess-spezifische Authentication Services oder durch einen zentralen Authentication Service realisiert werden. Die dabei verwendeten Schlüssel und Zertifikate müssen bereitgestellt werden, was durch separate Konfigurationsdateien, oder auch innerhalb einer Security Policy möglich ist.

**Ausgehend.** Ausgehende Endpunkte können ebenfalls über Security Policies abgesichert werden. Die Credentials zur Authentifizierung können hier aber dynamisch sein, da diese nicht nur von einem Endpunkt abhängen können, sondern auch von einem Benutzer, der variieren kann. Für die dynamischen Credentials, kann von der BPEL Engine ein Callback Handler bereitgestellt werden, über den die Middleware die erforderlichen Credentials abrufen kann. Die Credentials können innerhalb des Callback Handlers in der BPEL Engine über einen Identity Service in Erfahrung gebracht werden. Dadurch benötigt die Middleware keinerlei Kenntnisse über Benutzerinformationen.

**Bewertung.** Eine Authentifizierung über die Middleware bietet eine saubere Trennung zwischen Prozesslogik und Sicherheitslogik und ermöglicht die Anwendung verschiedener Authentifizierungsverfahren. Zudem kann eine Ende-zu-Ende Sicherheit (siehe 2.3) erreicht werden. Für eingehende Nachrichten entstehen dabei keinerlei unnötige Instanzen, da unerlaubte Zugriffe bereits von der Middleware abgefangen werden. Da aber keine Benutzerinformationen innerhalb des Prozesses vorhanden sind, muss dafür eine Erweiterung geschaffen werden, die entweder dem Prozess Zugriff auf die Benutzerinformationen in der Middleware ermöglicht, oder die Benutzerinformationen in den Prozess mitliefert.

### 6.4 Autorisierung in der Instanz

Eine Autorisierung in der Instanz kann deklarativ, vor einer Aktivität oder imperativ bei einer Aktivität erfolgen.

**Imperativ.** Die imperative Autorisierung erfolgt durch das Ausführen von BPEL-Code, der die Rollenüberprüfung übernimmt. Dieser Code kann nach den Aktivitäten bei eingehenden Nachrichten eingefügt werden.

**Deklarativ.** Bei der deklarativen Autorisierung werden die Sicherheitsanforderungen über Erweiterungsattribute im BPEL Code deklariert.

**Bewertung.** In beiden Fällen Vermischen sich, wie schon bei der Authentifizierung in der Instanz, die Prozesslogik und die Sicherheitslogik. Die deklarative Variante bietet dabei den Vorteil, dass keine unnötigen Prozessinstanzen entstehen, dafür müssen unter Umständen viele Erweiterungsattribute definiert werden. Dagegen bietet die imperative Variante größtmögliche Flexibilität.

### 6.5 Autorisierung in der Middleware

Die Autorisierung in der Middleware kann mit XACML (siehe 4.2) realisiert werden. Dazu muss für die Endpunkte ein Autorisierungs-Dienst (Authorization Service) eingerichtet werden, der die Entscheidung über eine Autorisierung treffen kann und als PDP eingesetzt wird. Die Middleware stellt dann den PEP dar, der entsprechende Anfragen (Decision Requests) an den PDP stellen kann.

**Bewertung.** Wie schon bei der Authentifizierung wird über die Middleware eine Trennung der Prozesslogik und der Sicherheitslogik erreicht. Durch das Autorisieren außerhalb der BPEL Engine entstehen auch hier keine unnötigen Instanzen. Ein Problem ist aber, dass die Middleware hier über keinerlei Informationen zu der Prozess-Instanz verfügt, für die die Nachricht bestimmt ist. Dies ist beispielsweise dann notwendig, wenn der Sender der Nachricht mit dem Benutzer der Prozess-Instanz verglichen werden muss. Policies die solche Prozessinformationen benötigen, können nicht auf Middleware-Ebene ausgewertet werden, da erst in der BPEL Engine die Prozessinstanz zu einer Nachricht ermittelt wird. Um dieses Problem zu umgehen gibt es noch die Möglichkeit die Autorisierung im Prozess zu realisieren.

### 6.6 Autorisierung im Prozess

Die Autorisierung im Prozess kann über ein Event-Modell realisiert werden, das von vielen BPEL Engines wie z.B. Apache ODE unterstützt wird. Damit können Aktivitäten überwacht werden und vor der Ausführung eine Autorisierung durchgeführt werden, die im negativen Fall eine Ausführung verhindern kann.

Damit sich dabei die Prozesslogik nicht mit der Sicherheitslogik vermischt, kann dazu eine Erweiterungskomponente für die BPEL-Engine realisiert werden. Des weiteren müssen für eine einfache Konfiguration einer solchen Erweiterungskomponente zusätzliche Adapter zwischen der Erweiterungskomponente und dem Autorisierungs-Dienst geschaffen werden.

**Bewertung.** Hiermit stehen bei der Autorisierung die Prozessinformationen zur Verfügung und eine Autorisierung kann flexibel an jeder Stelle im Prozess erfolgen. Dafür ist allerdings ein hoher Implementierungs- und Wartungsaufwand erforderlich.

## 7 Zusammenfassung

Authentifizierung und Autorisierung sind bei dem Einsatz von Web Services elementar wichtige Themen für die Sicherheit. Dazu stehen bereits eine ganze Reihe von Standards zur Verfügung, die erweiterbar sind, sehr flexibel eingesetzt werden können und den heutigen Sicherheitsanforderungen genügen. Die Implementierung für die BPEL Engine ist dabei nur ein Beispiel von vielen Einsatzmöglichkeiten.

Für die Zukunft zeichnen sich aber jetzt schon Schwachstellen beim Thema Zertifikate ab. Es ist bereits gelungen bei Hash-Algorithmen wie MD5 und mittlerweile auch SHA-1, durch Schwächen im Algorithmus Kollisionen zu erzeugen. Damit lassen sich gefälschte Zertifikate erstellen, die die gleiche Prüfsumme wie vertrauenswürdige Zertifikate aufweisen und bei einer Authentizitätsprüfung nicht von diesen zu unterscheiden sind. Noch ist dafür ein hoher Aufwand an Rechenleistung nötig, die aber mit der Zeit immer günstiger zu erwerben sein wird. Deshalb ist es wichtig in absehbarer Zeit auf sicherere Algorithmen wie SHA-2 umzusteigen.

# 8 Glossar

**BPEL (Business Process Execution Language)**
Sprache zur Modellierung von Geschäftsprozessen in XML.

**FTP (File Tranfer Protocol)**
Standardprotokoll für die Übertragung von Dateien über TCP/IP Netzwerke.

**HTTP (Hypertext Transfer Protocol)**
Standardprotokoll für die Übertragung von Daten über TCP/IP Netzwerke.

**HTTPS (Hypertext Transfer Protocol Secure)**
Standardprotokoll für die abhörsichere Übertragung von Daten über ein Netzwerk.

**LAN (Local Area Network)**
Lokales Netzwerk das in der Regel über mehrere Räume verteilt ist und vom Internet abgeschirmt wird.

**Middleware**
Eine Software, die zwischen zwei Anwendungen vermittelt und die Komplexität der Zugriffe reduziert.

**SSL (Secure Sockets Layer)**
Netzwerkprotokoll zur sicheren Datenübertragung im Internet.

**TCP/IP (Transmission Control Protocol / Internet Protocol)**
Standardprotokollsammlung für die Datenübertragung im Internet.

**URI (Uniform Resource Identifier)**
Ein eindeutiger Bezeichner der eine Ressource identifiziert.

**Whitespace**
Zeichen in einem Text zur Abgrenzung von Wörtern, wie zum Beispiel Leerzeichen, die im Texteditor nicht zu sehen sind.

**XML (eXtensible Markup Language)**
Standard für den Aufbau von Markup-basierten Sprachen.

# 9 Literatur


1. Papazoglou, M. P.: Web Services: Principles and Technology, PrenticeHall, 2007
2. OASIS: eXtensible Access Control Markup Language (XACML), http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml
3. OASIS: Web Services Security (WSS), http://www.oasis-open.org/committees/tc_home.-php?wg_abbrev=wss
4. Heiko Görig: Kontextbezogene Zugriffskontrolle für BPEL (Engines). Diplomarbeit Nr. 2866, Universität Stuttgart (2009)
5. Marc Chanliau: Web Services-Sicherheit und die SAML. http://entwickler.com/itr/online_artikel/psecom,id,468,nodeid,69.html (5.6.2009)
6. Ed Simon, Paul Madsen, Carlisle Adams: An Introduction to XML Digital Signatures. http://www.xml.com/pub/a/2001/08/08/xmldsig.html (5.6.2009)
7. Prof. Dr. Walter Lösel: Kapitel 2.1 Verschlüsselung und digitale Signatur. http://www.bw.fh-deggendorf.de/eb/f2/gliederung.htm (31.5.2009)
8. wikipedia.org: Public-Key-Infrastruktur. http://de.wikipedia.org/wiki/Public-Key-Infrastruk-tur (2.6.2009)
9. Manish Verma: XML Security: Control information access with XACML. http://www.ibm.com/developerworks/xml/library/x-xacml/ (3.6.2009)
10. OASIS: eXtensible Access Control Markup Language (XACML) Version 2.0. http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf (26.5.2009)
11. wikipedia.org: Zugriffskontrolle. http://de.wikipedia.org/wiki/Zugriffskontrolle (3.6.2009)
12. OASIS: Security Assertion Markup Language (SAML) V2.0 Technical Overview. http://www.oasis-open.org/committees/download.php/27819/sstc-saml-tech-overview-2.0-cd-02.pdf (26.5.2009)
13. OASIS: SAML V2.0 Executive Overview. http://www.oasis-open.org/committees/download.php/13525/sstc-saml-exec-overview-2.0-cd-01-2col.pdf (26.5.2009)
14. Netegrity Security Assertions Markup Language (SAML). http://xml.coverpages.org/Netegrity-SAMLWP.pdf (3.6.2009)
15. Michael Kain, Guido Keller: SAML 2.0, ein Tutorium - Teil 1: Theorie. http://www.acando.de/Global/GER/fachartikel_ger/kain_keller_JS_05_07.pdf (6.6.2009)
16. Bilal Siddiqui: Web Services Security, Part 1. http://webservices.xml.com/pub/a/ws/2003/03/04/security.html (26.5.2009)
17. OASIS: WS-SecurityPolicy 1.2. http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200512/ws-securitypolicy-1.2-spec-cd-01.pdf (8.6.2009)
18. OASIS: WS-Trust 1.3. http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.pdf (8.6.2009)
19. OASIS: Web Services Security: SOAP Message Security 1.1. http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMes-sageSecurity.pdf (8.6.2009)
20. Michel Alessandrini, Willi Nüßer, Michael Pollmeier: WS-Trust: Ein Einstieg in die Praxis. http://www.s-und-n.de/fileadmin/news/veroeffentlichungen/alessandrini_nuesser_2_js_04_07.pdf (26.5.2009)
21. Mario Jeckle, Barbara Zengler: SOAP aber sicher. http://www.jeckle.de/secureSOAP.html (28.5.2009)