Allgemeine Übersicht zu WS-Security

Seminararbeit

Betreuer:
Dipl.-Inform. Jochen Dinger
Universität Karlsruhe (TH)

vorgelegt am

Lehrstuhl für Praktische Informatik
Prof. Dr. W. Effelsberg
Universität Mannheim

im
August 2004

von
Alexander Grünke

Inhaltsverzeichnis

Seite

Abkürzungsverzeichnis

CA	Certificate Authority
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
PKI	Public Key Infrastruktur
POP	Post Office Protocol
RA	Registration Authority
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol
SSL	Secure Socket Layer
TCP	Transmission Control Protocol
TLS	Transport Layer Security
WS	Web Service
WSS	Web Service Security
XML	Extensible Markup Language
XMLEnc	XML Encryption
XMLSig	XML Signature

Abbildungsverzeichnis

1 Einleitung

Die Zahl der Sicherheitsvorfälle im IT-Bereich nimmt jährlich zu.[1] Dementsprechend wachsen auch die Sicherheitsanforderungen an Systeme, speziell im Internet. Dies gilt entsprechend auch für Web-Services, wobei diese sogar besonders gefährdet sind, da sie verschiedenste, für Angreifer unter Umständen interessante, Dienste anbieten und ihr großer Vorteil unter anderem darin liegt, dass sie leicht zugreifbar sind.[2] Daher ist es nun besonders wichtig bestehende Sicherheitskonzepte auch für Web-Services nutzbar zu machen und an den Punkten, wo die alten nicht alle Sicherheitsaspekte in zufrieden stellendem Maß abdecken, neue zu finden.

Ziel dieser Arbeit ist es, zunächst die Sicherheitsanforderungen an Web-Services aufzuzeigen und zu erläutern. Dabei werden sowohl allgemeine und spezifische Web-Service-Anforderungen, als auch Anforderungen an Firewalls betrachtet werden. An dieser Stelle werden auch einige grundlegende Konzepte vorgestellt, um den jeweiligen Sicherheitsanforderungen gerecht zu werden. Anschließend folgt eine kurze Betrachtung einiger bereits existierender Sicherheitstechnologien, welche zur Sicherung von Web Services relevant sind. Nachdem hier verdeutlicht wird, dass außer den bestehenden Verfahren weitere nötig sind, werden im letzten Teil das WS-Security Framework und seine Bestandteile genauer betrachtet. Hierbei soll insgesamt ein Überblick über die aktuellen Möglichkeiten gegeben werden, Sicherheit auch im Rahmen von Web-Services zu gewährleisten.

---

[1] Vgl. CERT/CC (2004)

[2] Vgl. O'Neill, Mark; et al. (2003): S. 22

2 Sicherheitsanforderungen an Web-Services

2.1 Allgemeine Anforderungen

Allgemein muss man sieben Sicherheitsaspekte beachten, um ein System sicher zu gestalten, wobei man absolute Sicherheit generell nicht erreichen kann. Diese Aspekte sind Vertraulichkeit (Confidentiality), Integrität (Integrity), Nicht-Abstreitbarkeit (Non-Repudiation), Authentifizierung (Authentication), Autorisierung (Authorization), Datenschutz (Privacy) und Verfügbarkeit (Availability). Des Weiteren unterscheidet man bezüglich Datensicherheit zwei Zustände, in denen Daten sich befinden können, nämlich im Speicher und auf dem Transportweg. Im Folgenden werden die sieben Punkte kurz erläutert und jeweils einige technische Umsetzungsmöglichkeiten kurz betrachtet.

2.1.1 Vertraulichkeit[3]

Um Vertraulichkeit zu gewährleisten, darf es keinem Dritten möglich sein die Kommunikation zwischen zwei Systemen unberechtigt mitzulesen. Ein Konzept, dies zu erreichen, ist, die ausgetauschten Nachrichten zu verschlüsseln. Aus Klartext wird mit Hilfe eines Algorithmus und eines Schlüssels ein sog. Ciphertext erzeugt. Dieser lässt sich im Idealfall von einem unbefugten Dritten nur mit einem dem Durchprobieren aller möglichen Schlüssel vergleichbaren Aufwand entschlüsseln. Dabei ist es heute üblich, dass der Algorithmus öffentlich bekannt und erläutert ist. Geheimgehaltene Algorithmen haben sich in der Vergangenheit allzu oft als fehlerbehaftet erwiesen. Wären sie vor ihrer Nutzung veröffentlicht und diskutiert worden, dann wären diese Fehler möglicherweise nicht erst aufgedeckt worden, nachdem sich das Verfahren bereits im Einsatz befand, und damit großer Schaden vermieden worden.
Generell unterscheidet man symmetrische und asymmetrische Verschlüsselungsalgorithmen bzw. Secret- und Public-Key-Verfahren.
Bei den symmetrischen Algorithmen wird der Ciphertext mit dem gleichen Schlüssel entschlüsselt, mit dem der Klartext verschlüsselt wurde. Diese Verfahren sind im Normalfall schneller als die asymmetrischen. Allerdings haben sie den großen Nachteil, dass man den geheimen Schlüssel vor Beginn der Verschlüsselung auf sicherem Weg zum Kommunikationspartner bringen

[3] Vgl. O'Neill, Mark; et al. (2003): S. 23-27

muss. Konkrete Secret-Key-Algorithmen sind beispielsweise DES, DES³, AES / Rijndael usw.
Bei den asymmetrischen Verfahren gibt es zwei Schlüssel, einen zum Verschlüsseln und einen zum Entschlüsseln. Ersterer ist meist öffentlich bekannt, so dass jeder eine Nachricht an den Besitzer des zweiten Schlüssels senden kann, welche aber nur dieser mit seinem geheim gehaltenen Schlüssel wieder entschlüsseln kann. Ein verbreiteter Algorithmus aus dieser Gruppe ist beispielsweise RSA.

2.1.2 Integrität[4]

Um die Integrität der Daten sicherzustellen muss man dafür sorgen, dass erkennbar ist, ob die Daten während des Transports manipuliert wurden oder verhindern, dass sie manipuliert werden können. Letzteres ist jedoch bei normalem Versand über das Internet generell unmöglich.
Für Ersteres bieten sich Hashing-Algorithmen an. Mit diesen wird eine mit dem menschlichen Fingerabdruck vergleichbare Kurzfassung der versendeten Daten, der Hash-Wert, erzeugt, wobei jede noch so kleine Änderung zu einem anderen Hash-Wert führt. Der Absender fügt diesen Hash-Wert an die versendeten Daten an. Der Empfänger kann nun den Hash-Wert der empfangenen Daten errechnen und diesen mit dem mitgesendeten vergleichen. Gängige Hash-Algorithmen sind beispielsweise MD2, MD4, MD5 und SHA.
Allerdings ist die Integrität der Daten immer noch nicht zuverlässig gewährleistet, da ein Dritter einfach die Daten ändern und dann den passenden neuen Hash-Wert anhängen könnte. Um dies zu verhindern, verwendet man digitale Signaturen. Hierbei nutzt man die Eigenschaft des asymmetrischen RSA-Verschlüsselungs-Algorithmus aus, auch ‚invers' zu funktionieren. Man kann also mit dem privaten Schlüssel Daten verschlüsseln, welche dann nur mit dem passenden öffentlichen Schlüssel wieder korrekt entschlüsselt werden können. Zum Erzeugen einer digitalen Signatur berechnet man nun zuerst den Hash-Wert der zu sendenden Daten und verschlüsselt diesen anschließend mit seinem privaten Schlüssel. Diese digitale Signatur fügt man nun an die Daten an, sie kann von jedermann mit dem passenden öffentlichen Schlüssel geprüft werden, womit das Ziel der prüfbaren Datenintegrität erreicht ist.

---

[4] Vgl. O'Neill, Mark; et al. (2003): S. 27-29

2.1.3 Nicht-Abstreitbarkeit[5]

Um diesem Aspekt gerecht zu werden, darf der tatsächliche Absender nicht abstreiten können, die Daten gesendet zu haben.
Dazu ist es nötig, dass die Identität des Absenders eindeutig feststellbar ist. Hierfür gibt es digitale Zertifikate, in welchen einige persönliche Daten des Inhabers, unter anderem Name, Adresse usw., zusammen mit seinem öffentlichen Schlüssel und einem festen Verfallsdatum gespeichert werden. Das gebräuchlichste Format für Digitale Zertifikate ist X.509.

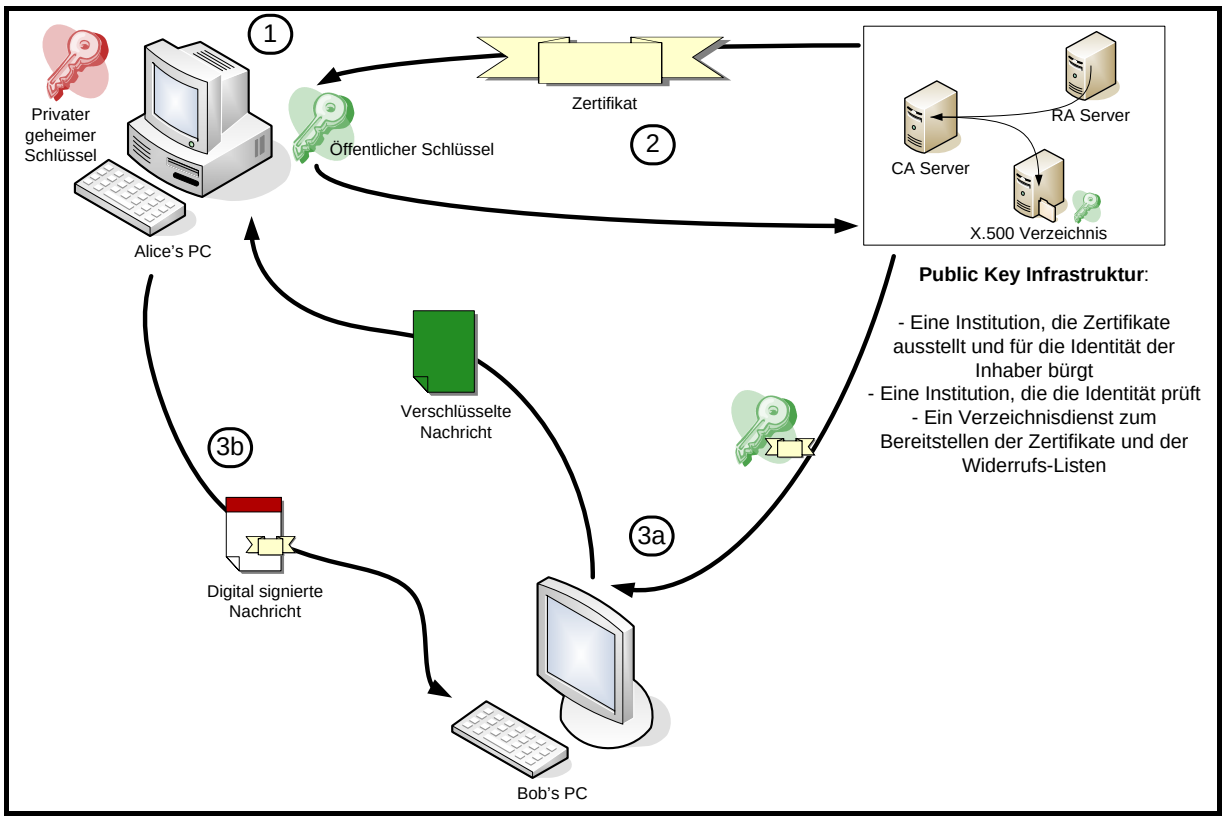

Abbildung 1: Veranschaulichung der Public Key Infrastruktur

Des Weiteren ist es nötig, dass die Korrektheit der Zertifikats-Informationen garantiert wird bzw. prüfbar ist. Daher werden Digitale Zertifikate nur nach Prüfung der persönlichen Daten durch eine Certificate Authority (CA) ausgestellt, welche danach für den Zeitraum der Gültigkeit dafür bürgt, dass das Zertifikat korrekt ist. Die Prüfung der Daten erfolgt im Auftrag der CA durch eine Registration Authority (RA). Die korrekten X.509 Zertifikate werden anschließend in öffentlichen Verzeichnissen abgelegt, welche normalerweise auf dem X.500 Standard basieren. Des Weiteren führt die CA eine öffentliche

[5] Vgl. O'Neill, Mark; et al. (2003): S. 29-31

Certificate Revocation List (CRL), in der Zertifikate aufgeführt werden, deren Gültigkeit widerrufen wurde, beispielsweise, weil der private Schlüssel des Inhabers nicht mehr mit hundertprozentiger Sicherheit geheim ist. Dieser Verbund aus CA, RA, X.500 Verzeichnis und CRL wird gewöhnlich als Public Key Infrastruktur bezeichnet, kurz PKI.
Mit Hilfe der PKI und digitaler Signaturen lässt sich nun auch Nicht-Abstreitbarkeit gewährleisten.
In Abbildung 1 wird der Aufbau der PKI veranschaulicht. Im Beispiel generiert Alice ihren privaten und ihren öffentlichen Schlüssel (1). Sie sendet den öffentlichen Schlüssel anschließend zwecks Zertifizierung an eine CA, welche ihre Daten von einer RA prüfen lässt und anschließend das Zertifikat ausstellt und sowohl an Alice zurücksendet als auch im X.500-Verzeichnis der CA veröffentlicht (2). Nun kann Bob das Zertifikat mit dem öffentlichen Schlüssel aus dem Verzeichnis laden und damit eine Nachricht an Alice verschlüsseln, welche nur sie wieder mit ihrem privaten Schlüssel entschlüsseln kann (3a). Außerdem hat Alice nun die Möglichkeit eine Nachricht an Bob digital zu signieren. Bob kann dann sowohl die Signatur prüfen, als auch kontrollieren, ob Alices Zertifikat noch gültig ist (3b).

2.1.4 Authentifizierung[6]

Bei Authentifizierung gilt es, die eigene Identität gegenüber dem Kommunikationspartners nachzuweisen. Die hierzu benötigten Informationen werden üblicherweise in die folgenden Kategorien eingeteilt (in je mehr Kategorien es passt, desto sicherer ist das Authentifizierungsverfahren):

- etwas, das man weiß (something you know), z.B. ein Passwort
- etwas, das man hat (something you have), z.B. eine Karte
- etwas, das man ist (something you are); hierunter fallen körperliche Merkmale, z.B. Fingerabdrücke

Konkrete Beispiele für Authentifizierungsmechanismen sind Biometrie oder auch Smartcards, welche nach Eingabe eines Passworts zufällige Einmal-Schüssel generieren. Für Web-Services sind aber im Prinzip nur Systeme geeignet, bei denen die nötigen Informationen digital gespeichert vorliegen, da sie ansonsten immer einen User bräuchten, der die Authentifizierung

[6] Vgl. O'Neill, Mark; et al. (2003): S. 32-34

durchführt. Zu diesen Verfahren gehört die Authentifizierung mittels Username und Passwort oder auch mittels einer bestehenden PKI, wobei die Identität mittels eines gültigen Zertifikats ausgewiesen werden kann.

2.1.5 Autorisierung[7]

Um diese Sicherheitsanforderung zu erfüllen, muss dafür gesorgt werden, dass ein Nutzer nur das machen darf, wofür er Rechte erhalten hat. Hierbei ist heute vor allem die Rollenbasierte Zugriffskontrolle von Bedeutung. Die Nutzer werden dabei entsprechend ihrer Aufgaben innerhalb einer Organisation bestimmten Rollen bzw. Gruppen zugeordnet, welche wiederum über bestimmte Zugriffsrechte verfügen. Dabei wird versucht, die reale Organisationsstruktur möglichst genau auf Rollen und Gruppen innerhalb des Systems abzubilden.

2.1.6 Datenschutz[8]

Der Datenschutzaspekt ist bei der Konzeption eines sicheren Systems ebenfalls von großer Bedeutung. Die entsprechenden rechtlichen und organisationseigenen Bestimmungen müssen bedacht werden, so dass eine Verletzung derselben nicht möglich ist. Allgemeine Verfahren hierfür gibt es nicht, im Normalfall lässt sich dieser Punkt durch umsichtige Gestaltung bei den vorgenannten Bausteinen - wie z.B. Autorisierung - ebenfalls absichern.

2.1.7 Verfügbarkeit[9]

Bei Verfügbarkeit ist vielleicht nicht auf Anhieb nachvollziehbar, warum dies ein bedeutender Sicherheitsbaustein ist. Allerdings gibt es eine spezielle Art von Angriffen, die genau auf diesen Punkt abzielen, so genannte Denial of Service Attacken. Bei diesen wird versucht, durch eine überwältigend hohe Zahl von Anfragen sämtliche Ressourcen eines Dienstes aufzubrauchen, so dass er für reguläre Nutzer nicht mehr erreichbar ist. Diese Attacke macht klar, dass auch dieser Aspekt bei der System-Konzeption berücksichtigt werden muss.

---

[7] Vgl. O'Neill, Mark; et al. (2003): S. 35

[8] Vgl. O'Neill, Mark; et al. (2003): S. 36

[9] Vgl. O'Neill, Mark; et al. (2003): S. 36

2.2 Spezielle Anforderungen

Neben den allgemeinen Sicherheitsanforderungen gibt es auch einige spezielle, die sich gezielt an Web-Services richten. Zum einen gilt es sog. Ende-zu-Ende Sicherheit zu garantieren, d.h. die Vertraulichkeit und Integrität der SOAP-Nachricht muss auch bei Übertragung über mehr als einen Web-Service sichergestellt sein, wie es beispielsweise bei reiner SSL-Verschlüsselung nicht der Fall wäre. Auch auf den Zwischenknoten müssen die Daten im Speicher weiterhin verschlüsselt und damit geschützt sein.[10]
Eine andere Anforderung ist die Einführung spezieller Autorisierungs- bzw. Authentifizierungsverfahren. Ein Web-Service benötigt unter Umständen zwecks Autorisierung Informationen über den End-Nutzer, ohne dass dieser direkt mit dem Web-Service kommuniziert, diese Daten müssen also in die SOAP-Nachrichten integriert werden. Auch soll sich der User nicht jedes Mal persönlich neu ausweisen müssen, wenn ein Vorgang beispielsweise über mehrere Web-Services abgewickelt wird. Hierfür werden Single-Sign-On-Methoden benötigt.[11]

2.3 Anforderungen an Firewalls[12]

Um die neuen Anforderungen an Firewalls zu klären, muss man zuerst einmal zwei Arten von Firewalls trennen: Paket-Filternde Firewalls und Application-Level Firewalls. Paket-Filternde Firewalls gleichen die einzelnen IP-Pakete anhand gewisser Kriterien mit festgesetzten Regeln ab. Je nach Ergebnis dieser Überprüfung werden die Pakete weitergeschickt oder verworfen. Zu den überprüften Kriterien können z.B. Zieladresse und -port, Absenderadresse und -port, das verwendete Protokoll oder auch das Format des Pakets gehören. Diese Art von Firewalls, welche mittlerweile bereits standardmäßig in Betriebssysteme und Router integriert ist, ist in der Praxis meistens so konfiguriert, dass sie nur Web- (HTTP, SSL) und E-Mail-Datenverkehr (POP, SMTP) durchlässt und die restlichen TCP/IP-Ports und andere Protokolle blockiert. Daher ist es mittlerweile üblich, auch andere Applikation über Web-Ports zu ‚tunneln'. Dies ist auch bei Web-Services der Fall.[13]

---

[10] Vgl. O'Neill, Mark; et al. (2003): S. 48-50

[11] Vgl. O'Neill, Mark; et al. (2003): S. 43-48

[12] Vgl. Albrecht, C. C. (2004)

[13] Vgl. O'Neill, Mark; et al. (2003): S. 57

Application-Level Firewalls dagegen wissen, welche Applikationen hinter der Wall welche Arten von Daten bekommen dürfen und prüfen die eingehenden Daten entsprechend dieser Regeln, wobei Port und Protokoll nun nur noch eine untergeordnete Rolle spielen. Diese Art von Firewall benötigt natürlich deutlich mehr Rechenzeit als die einfache Paket-Filternde Firewall.
Eine SOAP-Level Firewall, welche auch den neuen Anforderungen bzgl. Web-Services gerecht wird, ist also eine Application-Level Firewall, die in der Lage ist, SOAP-Nachrichten zu erkennen und sowohl zu prüfen, ob der Ziel-Web-Service überhaupt erreichbar sein soll, als auch zu verifizieren, ob die SOAP-Nachricht an sich syntaktisch korrekt ist, d.h. sie gegen entsprechende Schemata zu validieren. Außerdem können SOAP-Level Firewalls die im Rahmen der allgemeinen Anforderungen genannten Sicherheitsmerkmale prüfen bzw. einzelne nachträglich ergänzen, beispielsweise die Nachricht verschlüsseln, bevor diese das lokale Netz verlässt.[14]

[14] Vgl. O'Neill, Mark; et al. (2003): S. 58

3 Existierende Technologien

Nachdem jetzt die grundsätzlichen Sicherheitsanforderungen erläutert wurden sollen im Folgenden kurz einige konkrete Technologien und ihre Entwicklung vorgestellt werden, die speziell für den Einsatz im Zusammenhang mit Web Services Security interessant sind.

3.1 SSL / TLS[15]

Das SSL 1.0 Protokoll wurde im August 1994 von Netscape veröffentlicht, 5 Monate später folgte SSL 2.0, im Jahr darauf SSL 3.0. 1999 wurde dann der SSL-Nachfolger TLS 1.0 vom IETF als RFC 2246 standardisiert.

Die Protokolle gewährleisten innerhalb einer TCP/IP Punkt-zu-Punkt-Verbindung Vertraulichkeit, Integrität und einseitige bzw. beidseitige Authentifizierung. Die Daten werden also verschlüsselt und das bzw. die Zertifikate auf Gültigkeit geprüft. Hierbei sei angemerkt, dass man den Fall der beidseitigen Authentifizierung in der Praxis eher selten antrifft, zumeist weißt sich nur der Server aus, der Client hingegen nicht. Es ist auch möglich, auf den Authentifizierungs-Schritt ganz zu verzichten.

Während des Verbindungsaufbaus, des SSL bzw. TLS Handshakes, werden nach Abstimmung der Version und der Verschlüsselungsverfahren die X.509 Zertifikate ausgetauscht. Daraufhin wird ein geheimer Schlüssel für die symmetrische Verschlüsselung vom Client mittels des öffentlichen Schlüssels des Servers für die asymmetrische Verschlüsselung codiert und an diesen gesendet. Dieser geheime Schlüssel wird nun für die schnellere symmetrische Verschlüsselung der jetzt folgenden Datenübertragung genutzt. SSL bzw. TLS nutzen also eine Kombination aus asymmetrischer und symmetrischer Verschlüsselung, wobei sie den Geschwindigkeitsvorteil der letzteren für die eigentliche Datenübertragung nutzen und das Problem des Schlüsseltransports zu Beginn mittels asymmetrischer Verschlüsselung lösen.

Ein Nachteil von SSL/TLS-Verschlüsselung in Hinblick auf Web-Services ist, wie bereits erwähnt, dass die Daten zwischen den verschlüsselten Verbindungen als Plaintext vorliegen und damit ungeschützt sind.

---

[15] Vgl. Wikipedia (2004)

3.2 XML Encryption[16]

XML Encryption wurde im Dezember 2002 als Recommendation des W3C veröffentlicht. Der Standard definiert zum einen, wie XML-Dokumente verschlüsselt werden, zum anderen wie beliebige Dokumente in verschlüsselter Form im XML-Format dargestellt werden. Die Möglichkeit komplette XML-Dokumente zu verschlüsseln an sich ist dabei nichts neues, da es sich ja eigentlich nur um Text handelt. Neu dagegen ist die Möglichkeit, gezielt einzelne Abschnitte oder Elemente des Dokuments zu verschlüsseln, z.B. die Header-Informationen einer SOAP-Nachricht unverschlüsselt zu lassen und nur den Body-Inhalt zu codieren. Eine weitere Besonderheit bei XML Encryption ist die Canonicalization, eine Normalisierung des XML-Dokuments, welche gewährleistet, dass Namespace-Bindungen auch beim entschlüsseln innerhalb einer anderen Umgebung korrekt umgesetzt werden.
Durch den Standard werden keine neuen Algorithmen eingeführt. Viel mehr spezifiziert er, wie die Meta-Informationen über den genutzten Algorithmus und andere Parameter aussehen. Mittels XML Encryption lässt sich nun Ende-zu-Ende-Vertraulichkeit gewährleisten.

3.3 XML Signature[17]

XML Signature wurde im Februar 2002 als gemeinsame Spezifikation des W3C und der IETF veröffentlicht. Analog zu XML Encryption definiert XML Signature wie XML-Dokumente digital signiert werden und wie man digitale Signaturen beliebiger Daten als XML-Dokument darstellt. Ebenfalls ist es möglich, gezielt nur einzelne Teile eines XML-Dokuments zu signieren, beispielsweise einzelne Parameter einer SOAP-Nachricht. Auch bei XML Signature ist die Canonicalization von Bedeutung, welche auch dafür sorgt, dass verschieden formatierte XML-Dokumente, die aber inhaltlich identisch sind, letztlich die gleiche Signatur erhalten.
Mit XML Signature ist nun auch Ende-zu-Ende-Integrität möglich.

---

[16] Vgl. W3C (2002a)

[17] Vgl. W3C; IETF (2002)

4 Neue Spezifikationen: WS-Security

Die WS-Security Spezifikationen sind seit 2001 in Entwicklung und wurden erstmals im April 2002 von IBM und Microsoft veröffentlicht. Im Juni des gleichen Jahres wurden einige Teile zwecks Standardisierung an OASIS übergeben. Die Entwicklung wurde seit dem massiv durch die Industrie vorangetrieben, mittlerweile wird sie unterstützt von BEA Systems, Computer Associates, HP, IBM, Microsoft, Novell, SAP, Sun und einigen anderen.

Es handelt sich bei dem WS-Security Framework um eine Reihe von Spezifikationen, welche die Einbindung von Sicherheits-Elementen in SOAP-Nachrichten definieren. Dabei greifen sie auf bestehende Technologien wie z.B. XML Encryption, XML Signature oder auch den X.509 Standard zurück und integrieren diese. Die Definitionen sind dabei alle sehr modular gehalten. Es ist genau festgelegt, wie Sicherheitsmerkmale eingebunden bzw. Sicherheitsverfahren angewandt werden, aber welche Merkmale und Verfahren dies sein sollen bleibt weitestgehend freigestellt, so dass sich die Spezifikationen alle auch für die Integration in bereits bestehende Umgebungen eignen bzw. auch offen für zukünftige Entwicklungen sind.

Abbildung 2 zeigt den ursprünglich geplanten Aufbau des WS-Security Frameworks und was davon bis heute umgesetzt wurde.

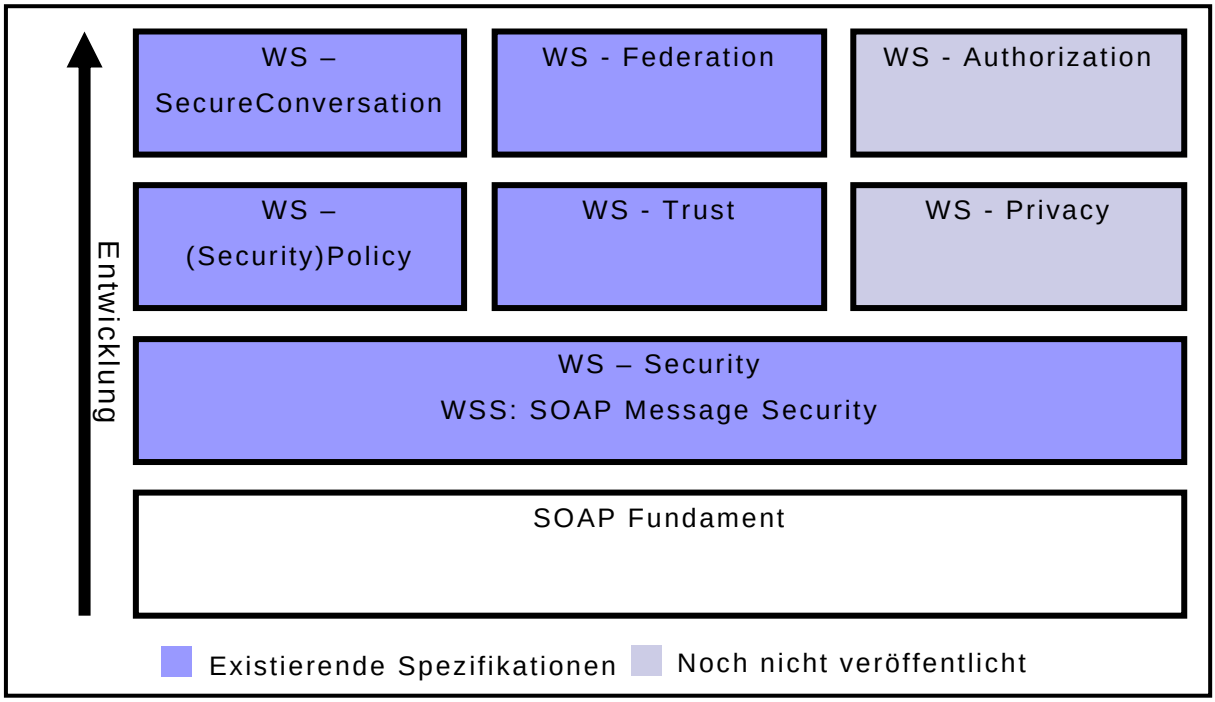

Abbildung 2: Ursprüngliches WS Security Framework[18]

[18] Vgl. Microsoft Corp.; IBM Corp. (2002)

4.1 WSS: SOAP Message Security[19]

WSS: SOAP Message Security, aus der grundlegenden WS-Security-Spezifikation hervorgegangen, wurde im März 2004 von OASIS standardisiert. Der Standard beschreibt eine Erweiterung des SOAP-Protokolls zur Gewährleistung von Vertraulichkeit und Integrität beim Austausch einzelner SOAP-Nachrichten. Dabei unterstützt er eine Vielzahl von Sicherheitstechnologien, wobei speziell XML Encryption und XML Signature genutzt werden. Des Weiteren beschreibt er generell, wie Sicherheits-Merkmale, sog. Security-Tokens, in SOAP-Nachrichten eingebunden und codiert werden. Tokens sind beispielsweise persönliche Nutzer-Informationen, Schlüssel oder auch Zertifikate. Innerhalb einer SOAP-Nachricht stehen die WSS: SOAP Message Security Ergänzungen dabei im Header, wie Abbildung 3 anhand einer schematischen Darstellung verdeutlicht.

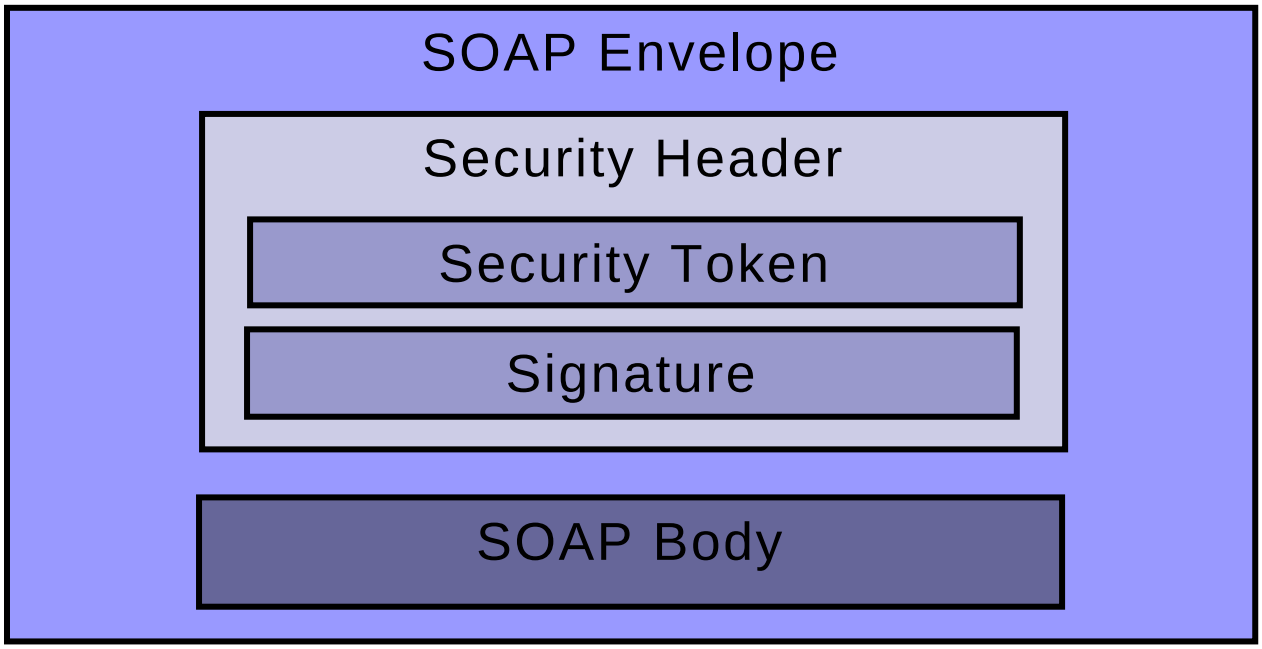

Abbildung 3: Schema einer SOAP Nachricht mit Security-Header

Ebenfalls bereits 2004 standardisiert wurden zwei Sub-Standards zur Definition von Sicherheits-Tokens innerhalb von WSS: SOAP Message Security: WSS: Username Token Profile 1.0 und WSS: X.509 Certificate Token Profile 1.0.

---

[19] Vgl. OASIS (2004)

4.2 WS-Policy / WS-SecurityPolicy[20]

WS-Policy 1.1 wurde im Mai 2003 von IBM, Microsoft, BEA und SAP veröffentlicht. WS-SecurityPolicy 1.0 wurde bereits im Dezember 2002 von IBM, Microsoft, VeriSign, und RSA publiziert. WS-Policy ist eine SOAP-Erweiterung zur Beschreibung von Policies, zu deutsch Regeln oder Verfahrensweisen. Sie existiert unabhängig von WS-Security und beschränkt sich entsprechend nicht nur auf Sicherheitsmerkmale sondern ist eher auf Geschäftsregeln ausgerichtet. WS-SecurityPolicy hingegen versteht sich als reine Ergänzung zu WS-Security.
Die Spezifikationen definieren, wie man Anforderungen, Präferenzen und vorhandene Fähigkeiten innerhalb einer SOAP-Nachricht beschreibt. Beispielsweise kann ein Web Service damit eindeutig mitteilen, welche Authentifizierungsverfahren er unterstützt, welches er bevorzugt oder auch welche Verschlüsselungsart er innerhalb der Kommunikation mindestens erwartet.

4.3 WS-Trust[21]

Im Mai 2004 veröffentlichten IBM, Microsoft, VeriSign, BEA, RSA u. a. WS-Trust 1.1. Die Spezifikation definiert SOAP Methoden für die Anforderung und Ausstellung von Sicherheits-Tokens und für die Vermittlung von Vertrauensverhältnissen von einer Vertrauens-Zone zur nächsten. In diesem Zusammenhang ist auch der Security-Token-Service zu nennen, der nach Erhalt eines von ihm akzeptierten Tokens einer anderen Vertrauenszone, beispielsweise eines X.509 Zertifikats, eine Token für seine eigene Vertrauenszone ausstellt, z.B. ein Kerberos-Ticket. Dieses Token wird benötigt um die Dienste innerhalb dieser Vertrauenszone zu nutzen.
Dabei ist die Spezifikation, wie alle im Rahmen von WS-Security, wieder sehr flexibel und erweiterbar und lässt sich mit verschiedenen gängigen Methoden kombinieren.

4.4 WS-Federation[22]

Im Juli 2003 wurde durch IBM, Microsoft, VeriSign, BEA und RSA die WS-Federation-Spezifikation 1.0 herausgegeben. Hierin werden aufbauend auf

---

[20] Vgl. Microsoft Corp.; IBM Corp.; et al. (2002 & 2003b)

[21] Vgll. Haupt, A.; Tamm, G. (2003), Microsoft Corp.; IBM Corp.; et al. (2004b)

[22] Vgl. Haupt, A.; Tamm, G. (2003), Microsoft Corp.; IBM Corp.; et al. (2003a)

WSS: SOAP Message Security, WS-Trust, WS-Policy und WS-SecureConversation Mechanismen definiert, um Zusammenschlüsse mehrerer Vertrauenszonen in einem Verbund zu ermöglichen.
Hierbei sind der Identity-Provider und der Pseudonym-Service erwähnenswert. Der Identity-Provider, meist eine Erweiterung zum Security-Token-Service, stellt, nachdem der Nutzer sich ihm gegenüber ausgewiesen hat, dem Nutzer eine temporäre ID für eine andere Vertrauenszone aus, zu der ein Trust-Verhältnis besteht. Dieser kann nun mit dieser ID dort im Rahmen der festgelegten Möglichkeiten agieren. Ein Pseudonym-Service verwaltet innerhalb eines Verbundes die verschiedenen Aliase eines Nutzers, ausgehend von der Annahme, dass er für die unterschiedlichen Ressourcen im Verbund jeweils ein anderes Alias verwendet. Diese Konzepte sind speziell im Hinblick auf Single-Sign-On und eben Identity-Mapping von Interesse.

4.5 WS-Privacy[23]

WS-Privacy sollte ursprünglich das Zusammenspiel von WSS: SOAP Message Security, WS-Policy und WS-Trust definieren, um automatische Bekanntgabe bzw. Einhaltung von Datenschutz-Richtlinien innerhalb einer SOAP-Kommunikation zu ermöglichen. Allerdings wurde dazu bis heute keine Spezifikation veröffentlicht.

4.6 WS-SecureConversation[24]

WS-SecureConversation 1.1 wurde im Mai 2004 von IBM, Microsoft, VeriSign, BEA, RSA u. a. publiziert. Die Spezifikation beschreibt auf WSS: SOAP Message Security und WS-Trust basierend Mechanismen um eine sichere Kommunikation auch über mehrere Nachrichten hinweg zu ermöglichen. Dabei geht es speziell um den Aufbau und die gemeinsame Nutzung eines Sicherheits-Kontexts sowie das Wiedererlangen eines Schlüssels aus einem bestehenden Kontext.

---

[23] Vgl. Microsoft Corp.; IBM Corp. (2002)

[24] Vgl. Microsoft Corp.; IBM Corp.; et al. (2004a)

4.7 WS-Authorization[25]

WS-Authorization sollte der ursprünglichen WS-Security-Roadmap von 2002 nach definieren, wie Zugriffsrechte für Web-Services spezifiziert und verwaltet werden. Auch zu diesem geplanten Standard liegt bis heute keine Publikation vor.

[25] Vgl. Microsoft Corp.; IBM Corp. (2002)

5 Zusammenfassung

Web-Services werden mit zunehmender Verbreitung immer häufiger das Ziel von Angriffen werden. Daher ist es zwingend notwendig beim Entwurf eines Web-Service immer auch die Sicherheitsanforderungen an denselben zu bedenken. Dabei sollte man die allgemeinen Anforderungen Vertraulichkeit, Integrität, Nicht-Abstreitbarkeit, Authentifizierung, Autorisierung, Datenschutz und Verfügbarkeit bedenken. Außerdem ist es wichtig, sich darüber im Klaren zu sein, dass bestehende Technologien wie SSL alleine nicht ausreichen, da sich für Web-Services neue Herausforderungen, wie z.B. Ende-zu-Ende-Sicherheit ergeben.

Die neuen Spezifikationen und Standards im Rahmen des WS-Security Frameworks bieten für viele dieser Anforderungen diverse Arten der Einbindung in SOAP-Nachrichten und Web-Services. Dabei sind alle Spezifikationen sehr modular und flexibel gestaltet, so dass sie sich einfach in bestehende Umgebungen integrieren und mit vorhandenen Technologien wie z.B. X.509 oder Kerberos kombinieren lassen. Auch zukünftige Neuerungen beispielsweise im Bereich der Kryptographie lassen sich somit aller Voraussicht nach einbinden, ohne dass Änderungen des Standards oder auch nur größere Änderungen an Web-Service-Grundstrukturen nötig werden.

Zusätzlich zur Nutzung der sich durch WS-Security bietenden Möglichkeiten, Web-Services und SOAP-Kommunikation sicher zu machen sollte man außerdem Application-Level Firewalls, speziell SOAP-Level Firewalls als Ergänzung der Sicherheit in Erwägung ziehen. Diese sind zwar Prozessor-lastiger, bringen aber viele Vorteile mit sich, die wiederum den Web-Service selbst entlasten.

Literaturverzeichnis

Albrecht, C. C. (2004): How Clean is the Future of SOAP?, Communications of the ACM Vol. 47 No. 2: 66-68.

CERT/CC (2004): CERT/CC Statistics 1988-2003, http://www.cert.org/stats/cert_stats.html#incidents, Stand: 24.07.2004.

Geer, D. (2003): Taking Steps to Secure Web Services, IEEE Computer Vol. 36 No. 10: 14-16.

Haupt, A.; Tamm, G. (2003): Trust and Security for Web-Services.

Microsoft Corp.; IBM Corp. (2002): Security in a Web Services World: A Proposed Architecture and Roadmap, http://msdn.microsoft.com/ws-security/, Stand: 24.07.2004.

Microsoft Corp.; IBM Corp.; et al. (2002): Web Services Security Policy Language, http://msdn.microsoft.com/ws/2002/12/ws-security-policy/, Stand: 24.07.2004.

Microsoft Corp.; IBM Corp.; et al. (2003a): Web Services Federation Language (WS-Federation), http://msdn.microsoft.com/ws/2003/07/ws-federation/, Stand: 24.07.2004.

Microsoft Corp.; IBM Corp.; et al. (2003b): Web Services Policy Framework (WS-Policy), http://msdn.microsoft.com/ws/2002/12/Policy/, Stand: 24.07.2004.

Microsoft Corp.; IBM Corp.; et al. (2004a): Web Services Secure Conversation Language (WS-SecureConversation), http://msdn.microsoft.com/ws/2004/04/ws-secure-conversation/, Stand: 24.07.2004.

Microsoft Corp.; IBM Corp.; et al. (2004b): Web Services Trust Language (WS-Trust), http://msdn.microsoft.com/ws/2004/04/ws-trust/, Stand: 24.07.2004.

OASIS (2004): Web Services Security: SOAP Message Security 1.0 (WS-Security 2004), http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf, Stand: 24.07.2004.

O'Neill, Mark; et al. (2003): Web Services Security, McGraw-Hill Osborne Media.

W3C (2002a): XML Encryption Syntax and Processing, http://www.w3.org/TR/xmlenc-core/, Stand: 24.07.2004.

W3C (2002b): XML Exclusive Canonicalization 1.0, http://www.w3.org/TR/xml-exc-c14n/, Stand: 24.07.2004.

W3C; IETF (2002): XML Signature Syntax and Processing, http://www.w3.org/TR/xmldsig-core/, Stand: 24.07.2004.

Wikipedia (2004): Secure Sockets Layer, http://de.wikipedia.org/wiki/SSL, Stand: 24.07.2004