hochschule für angewandte wissenschaften
FACHBEREICH ELEKTROTECHNIK UND INFORMATIK hamburg
university of applied sciences

Hochschule für Angewandte Wissenschaften Hamburg
Fachbereich Elektrotechnik und Informatik SS 2005
Masterstudiengang Anwendungen I
Kai von Luck

Web Service Security

Thies Rubarth
rubart_t@informatik.haw-hamburg.de

Dieses Dokument ist die Ausarbeitung eines Vortrages, der am 1. Juni 2005
an der HAW Hamburg gehalten wurde.

Inhaltsverzeichnis

1 Einleitung

An jede Anwendung müssen unter Berücksichtigung der Daten, die sie verarbeitet, bestimmte Sicherheitsanforderungen gestellt werden. Zum einen muss das System vor unberechtigtem Zugriff geschützt werden. Dazu werden Authentifikation und Autorisierung genutzt. Die Authentifikation stellt sicher, dass ein Benutzer, der sich am System anmeldet, wirklich derjenige ist, für den er sich ausgibt. Die Autorisierung stellt sicher, dass die Benutzer nur Aktionen ausführen dürfen, zu denen sie berechtigt sind. Desweiteren muss sichergestellt werden, dass die Daten nicht von Dritten gelesen oder verändert werden können, dass heißt Vertraulichkeit und Integrität müssen gewährleistet werden.

Web Services sind Anwendungen, die über das Internet von anderen Anwendungen aufgerufen werden können. Da das Internet eine sehr „offene“ Infrastruktur bietet können diese Dienste leicht angegriffen werden. Viele Sicherheitsrisiken im Internet werden heute durch Firewalls bekämpft. Im Fall der Web Services ist dies jedoch nicht möglich, da die SOAP-Nachrichten, die in der Regel für die Kommunikation mit Web Services eingesetzt werden, extra darauf ausgelegt sind leicht durch Firewalls hindurch zu kommen. In dieser Ausarbeitung soll deshalb untersucht werden, wie eine auf Web Services basierende Anwendung sicher gemacht werden kann.

2 Secure Socket Layer

Secure Socket Layer (SSL) ist ein Protokoll, das sichere Kommunikation im Internet über HTTPS ermöglicht. SSL unterstützt dabei die Authentifikation des Servers und des Clients und die verschlüsselte Datenübertragung zwischen den beiden Kommunikationspartnern.

2.1 SSL Handshake

Damit die Kommunikation zwischen Client und Server verschlüsselt werden kann, muss zu beginn der Kommunikation ein Handshake (siehe Abbildung 1) stattfinden, bei dem die zur Verschlüsselung verwendeten Algorithmen festgelegt und die benötigten Schlüssel ausgetauscht werden. Optional können Client und Server dabei authentifiziert werden.

Zu beginn des Handshake sendet der Client eine Hello-Nachricht an den Server. In dieser Nachricht wird auch mitgeteilt, welche Algorithmen vom Client unterstützt werden.

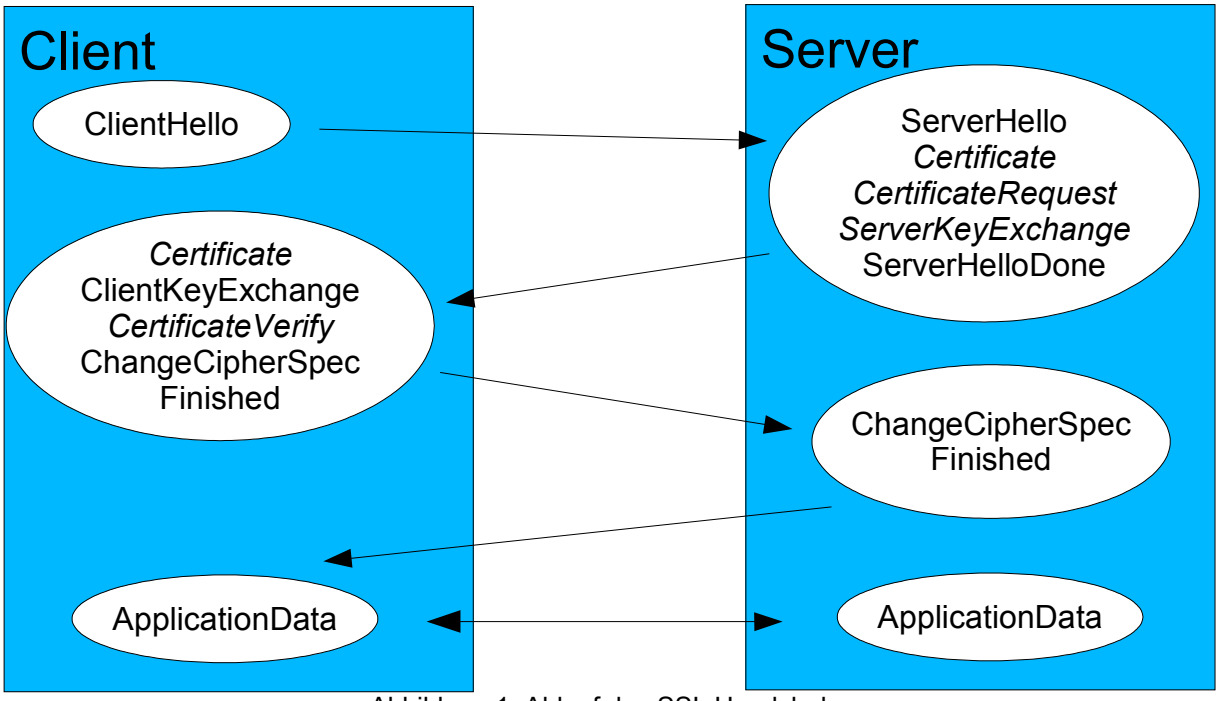

Abbildung 1: Ablauf des SSL Handshake

Der Server antwortet dann mit seiner Hello-Nachricht. Um sich zu authentifizieren kann der Server sein Zertifikat mitschicken und gegebenenfalls ein Zertifikat vom Client anfordern. Wenn der Server kein Zertifikat mitschickt, muss er für den Schlüsselaustausch ein temporäres Zertifikat erstellen und als „ServerKeyExchange“ an den Client senden. Desweiteren legt der Server fest, welche Algorithmen zum Verschlüsseln verwendet werden.

Der Client antwortet gegebenenfalls auf der Zertifikatanforderung und sendet dann die „ChangeCipherSpec“-Nachricht, die besagt, dass von nun an die Daten verschlüsselt übertragen werden sollen. Der Server bestätigt diese Nachricht von seiner Seite und beide Kommunikationspartner können mit dem austauschen von verschlüsselten Daten beginnen.

2.2 Grenzen von SSL

SSL bietet keine Möglichkeit der Autorisierung. Das bedeutet, dass die Autorisierung unabhängig von SSL gemacht werden muss, so dass auch die Authentizität des Benutzers neu sichergestellt werden muss. SSL bietet also nur die Möglichkeit Daten sicher zu übertragen.

Der zweite Nachteil von SSL besteht darin, dass der Sicherheitskontext nur zwischen zwei direkten Kommunikationspartnern besteht. Ruft ein Client z.B. einen Web-Service auf, der einen weiteren Web-Service aufruft, ist es nicht möglich, dass der Client Teile der Daten so verschlüsselt, dass nur der zweite Web-Service, nicht aber der zwischengeschaltete Web-Service die Daten lesen kann (siehe Abbildung 2).

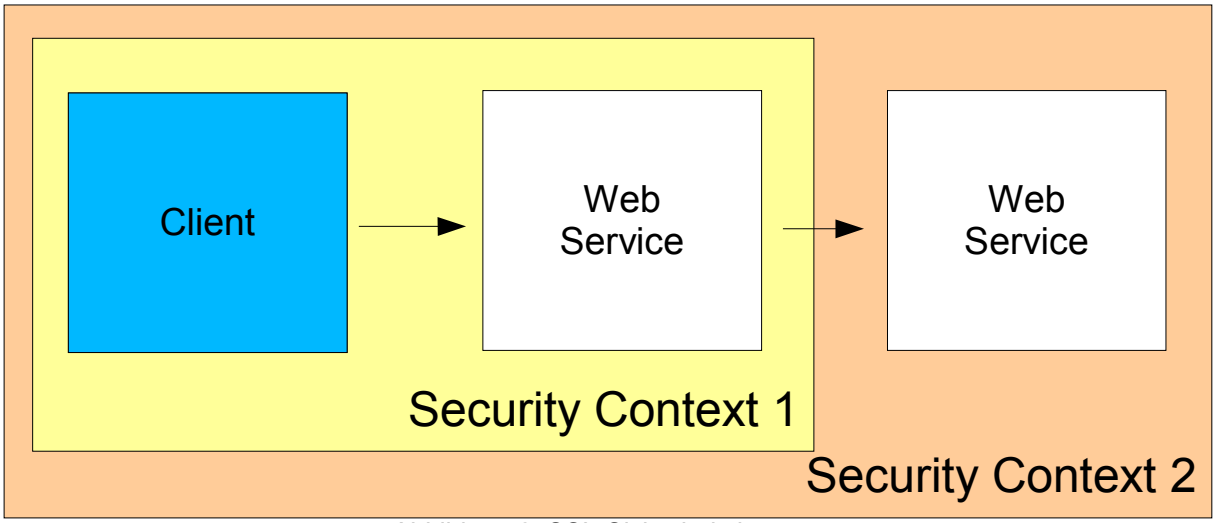

Abbildung 2: SSL-Sicherheitskontext

SSL ist deshalb nicht geeignet um eine Anwendung sicher zu machen, da es nur sicherstellt, dass HTTP-Nachrichten sicher übertragen werden. Um eine Anwendung sicher zu machen, werden Mechanismen benötigt, mit denen die Sicherheitsaspekte direkt in den Nachrichten enthalten sind.

3 Verschlüsseln und Signieren von XML-Dokumenten

Um SOAP-Nachrichten sicher zu machen, muss es möglich sein Teile der Nachrichten zu verschlüsseln, um die Vertraulichkeit und die Integrität der Daten sicher zu stellen. Damit eine Authentifikation und Autorisierung möglich ist, muss es möglich sein Teile der Nachrichten zu signieren.

Da SOAP-Nachrichten XML-Dokumente sind, bieten sich hier XML-Encryption und XML-Signature an.

3.1 XML-Encryption

XML-Encryption ist eine Spezifikation, die es ermöglicht komplette XML-Dokumente oder Teile von XML-Dokumenten zu verschlüsseln. Dazu wird spezifiziert, wie die Verschlüsselten Daten in das XML-Dokument eingebunden werden, und welche Meta-Informationen (z.B. verwendete Algorithmen, Schlüssel usw.) benötigt werden.

Für Details wird auf die Folien des Vortrags bzw. die Quelle [6] verwiesen.

3.2 XML-Signature

XML-Signature ist eine Spezifikation, die es ermöglicht komplette XML-Dokumente oder Teile davon zu signieren. Wie XML-Encryption spezifiziert XML-Signature, wie die Signaturen in das XML-Dokument eingebunden werden und welche Meta-Informationen beötigt werden.

Für Details wird auf die Folien des Vortrags bzw. die Quelle [7] verwiesen.

4 WS -* Spezifikationen

Die WS-* Spezifikationen sind eine Reihe von Spezifikationen, die von den großen Software-Unternehmen wie z.B. IBM und Microsoft entwickelt wurden und die Qualität von Web Services zu verbessern (siehe Abbildung 3).

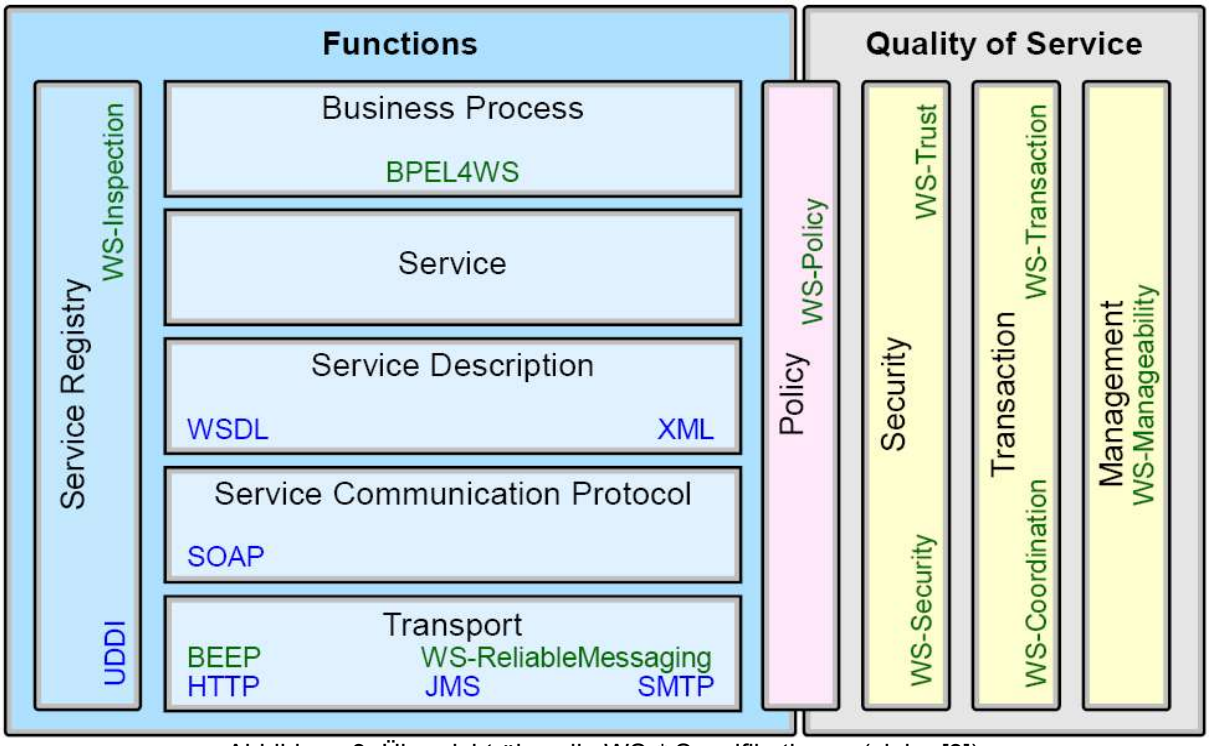

Abbildung 3: Übersicht über die WS-* Spezifikationen (siehe [8])

Für die Sicherheit von Web Services gibt es folgende Spezifikationen:

- WS-Security
- WS-Policy
- WS-Trust

Diese Spezifikationen sind noch nicht freigegeben und daher noch nicht in Anwendungsservern oder Rahmenwerken für den Aufruf von Web-Services implementiert. Die Verfügbarkeit im April 2004 ist in der Abbildung 4 zu sehen. Bis heute scheint sich der Stand nicht wesentlich verändert zu haben.

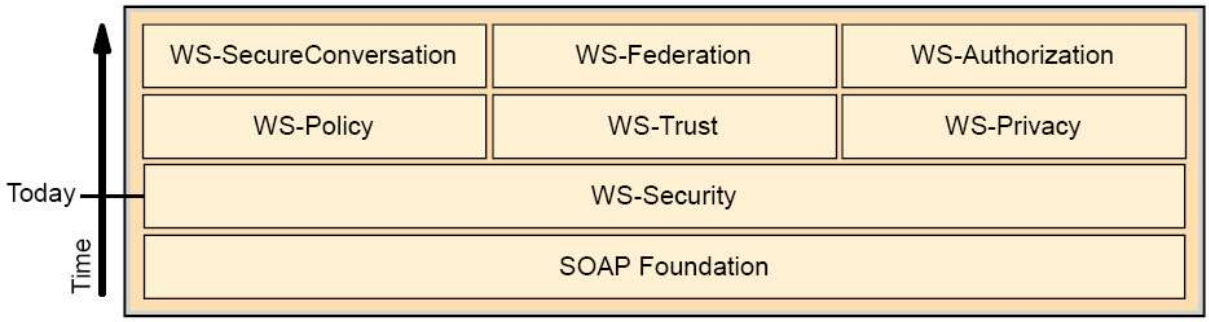

Abbildung 4: Verfügbarkeit der Sicherheits-Spezifikationen (siehe [8])

Da es notwendig ist die Kommunikation über Web Services sicher zu machen, bietet es sich trotz der nicht gegeben Verfügbarkeit von Implementierungen der Sicherheitsspezifikationen an, bei einer eigenen Implementierung der Sicherheit auf diesen Spezifikationen aufzusetzen.

4.1 WS-Security

WS-Security ist eine Spezifikation, die es ermöglicht SOAP-Nachrichten durch angeben bestimmter „Tokens“ sicher zu machen. WS-Security ist dabei unabhängig von den Verfahren, mit dem Daten verschlüsselt oder signiert werden.

WS-Security spezifiziert nicht, wie ein Sicherheitskontext über mehrere Nachrichten hinweg aufgebaut werden kann und legt auch nicht fest, wann eine Nachricht vertrauenswürdig ist. Auch der Austausch von Schlüsseln wird nicht von WS-Security spezifiziert.

WS-Security ermöglicht es auf einheitliche Art und Weise die Daten einer SOAP-Nachricht mittels XML-Encryption zu verschlüsseln bzw. mittels XML-Signature zu signieren.

Für Details wird auf die Folien des Vortrages und die Quelle [2] verwiesen.

4.2 WS-Policy

WS-Policy ist eine Spezifikation, die es ermöglicht Anforderungen für einen Web Service anzugeben. Dabei ist es möglich verschiedene Anforderungen zu kombinieren. Die „Policy“ eines Web Service kann deshalb aus mehreren „Policy Alternatives“ bestehen, die mehrere „Policy Assertions“ enthalten können. Eine „Policy Assertion“ ist eine konkrete Anforderung, wie z.B. „es wird ein Kerberos-Ticket benötigt“.

Für Details wird auf die Folien des Vortrages und die Quelle [3] verwiesen.

4.3 WS-Trust

WS-Trust ist eine Spezifikation, die sich um die Erstellung vertrauenswürdiger SOAP-Nachrichten kümmert. Um eine vertrauenswürdige SOAP-Nachricht zu erstellen muss festgelegt werden, was Vertrauen ist, und wie dieses sichergestellt werden kann. WS-Trust legt deshalb fest wie und welche „Security Tokens“ ausgetauscht werden. Dabei geht es insbesondere um die Erstellung, Erneuerung und Validierung der Tokens.

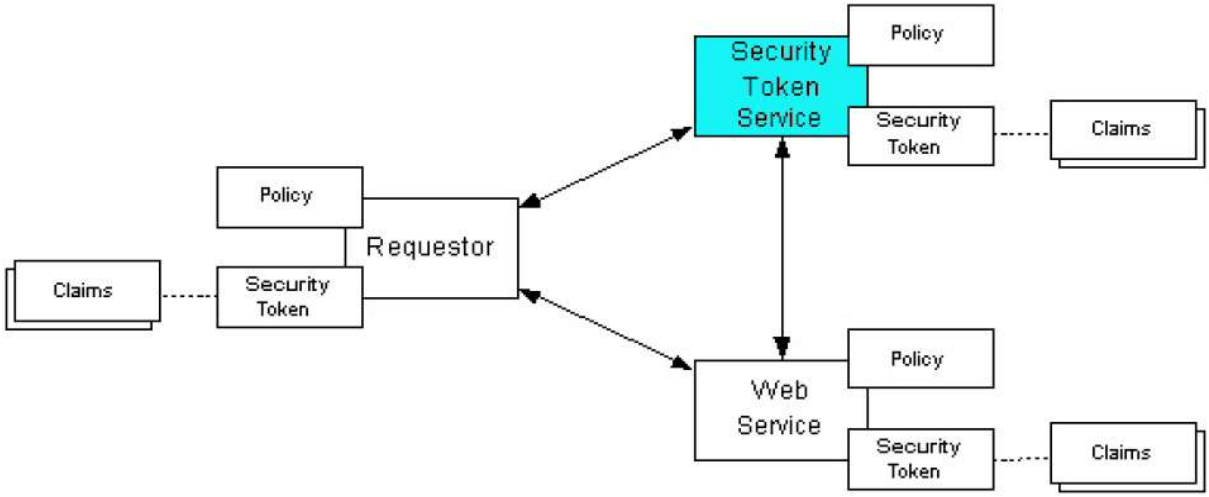

Abbildung 5: WS-Trust Model (siehe [4])

Zentraler Punkt von WS-Trust ist der „Security Token Service“ (siehe Abbildung 5). Dieser Dienst stellt die Gültigkeit von Tokens sicher. Dabei gibt es verschiedene Ansätze. Bei dem Ansatz „Fixed Trust Roots“ gibt es feste Dienste, denen vertraut wird und nur Tokens, die von diesen Diensten erstellt wurden sind gültig. Bei dem Konzept der „Trust Hierarchies“ gibt es eine Hierarchie von Diensten, die Tokens erstellen dürfen. Der Client (bzw. der Requestor) muss dann entweder die komplette Hierarchie seines Tokens mitliefern, oder der Web Service hat die Möglichkeit die Hierarchie selbst zu ermitteln. Bei dem dritten Ansatz gibt es einen zentralen „Authentication Service“, der die Gültigkeit eines Tokens überprüfen kann.

Für Details wird auf die Folien des Vortrags und auf die Quelle [4] verwiesen.

5 Zusammenfassung und Angebot

Mit den Spezifikationen WS-Security, WS-Policy und WS-Trust ist es möglich eine Sicherheitsarchitektur zu erstellen, die auf einer Public Key Infrastructure (PKI) basiert. Das Hauptproblem bei einer PKI ist das vergeben von Zertifikaten, so dass sichergestellt ist, dass die Person, die ein Zertifikat bekommt auch wirklich diese Person ist. Kann dies nicht sichergestellt werden, kann das Zertifikat nicht für die Authentifikation verwendet werden.

In einem Ferienclub-Projekt ist dieses Problem leicht zu lösen, da der Gast persönlich bei der Rezeption erscheinen muss, so dass bei Vergabe des Zertifikats sichergestellt ist, dass kein Hacker sich dieses Zertifikat über das Internet „ergaunert“ hat.

Das Angebot für das Projekt ist also der Entwurf eines Trust-Konzeptes und einer Sicherheitsarchitektur, die auf den WS-*-Spezifikationen aufsetzt, sowie die Erstellung der dafür notwendigen Infrastruktur.

Literaturverzeichnis

[1] Prof. Dr. Stefan Fischer - VL-Skript: Enterprise Applications (SS 2004)
[2] Bob Atkinson et. al. - WS-Security Version 1.0 (5. April 2002)
[3] Siddharth Bajaj et. al. - WS-Policy (September 2004)
[4] Steve Anderson et. al. - WS-Trust (Februar 2005)
[5] Alan O. Freier et. al. - The SSL Protocol Version 3.0 (März 1996)
[6] Takeshi Imamura et. al. - XML Encryption Syntax and Processing (Dezember 2002)
[7] Mark Bartel et. al. - XML-Signature Syntax and Processing (Februar 2002)
[8] Mark Endrai et. al. - Patterns: Service- Oriented Architecture and Web Services
[9] Mark O'Neill - Web Services Security