Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Ausarbeitung Seminar Ringvorlesung
Master Thesis Outline

Eike Falkenberg

Performance Untersuchung von WS-Security
Implementierungen in interoperablen Umgebungen

Fakultät Technik und Informatik
Studiendepartment Informatik

Eike Falkenberg

Thema der Ausarbeitung Seminar Ringvorlesung
Master Thesis Outline
Performance Untersuchung von WS-Security Implementierungen in interoperablen Umgebungen
Stichworte
Performance WS-* Interoperabilität
Kurzzusammenfassung
Lose gekoppelte Architekturen gewinnen mehr und mehr an Bedeutung. Meist wird bei der Realisierung die offene und plattformunabhängige Web Services Technologie verwendet. Wesentliche Sicherheits Standards wurden erst vor kurzem verabschiedet und die ersten Implementierungen der Hersteller sind verfügbar. Im Rahmen der Master Thesis soll das Performance Verhalten, insbesondere im heterogenen Umfeld und bei interoperablen Zugriffen, in einem praxisnahen Anwendungsszenario untersucht werden.

Inhaltsverzeichnis

1 Einleitung

Web Services sind heute die erste Wahl, um lose verteilte Systeme zu konstruieren. Die schlanke, offene und interoperable Struktur hat der Web Service Technologie zu ihrem großen Erfolg verholfen. Wie so oft, wurde erst im nachhinein der Aspekt der Sicherheit berücksichtigt und dann auf die bereits bestehende Technik aufgesetzt.
Web Services bieten eine Reihe von Angriffspunkten. Die Kommunikation findet über klartext Nachrichten im XML Format statt, was zu der Gefahr des Abhörens führt, wie es beispielsweise in (Eckert, 2004) oder (Cyrus Peikari, 2004) beschrieben ist. Bewährte Techniken wie Firewalls helfen im Normalfall wenig, da die Web Services offene Ports benötigen und auch Ad-Hoc erreichbar sein müssen.
SSL (Secure Socket Layer) ist für dieses Problem keine Lösung, da SSL lediglich eine Punkt-zu-Punkt Verschlüselung ermöglicht. Web Services bieten die Möglichkeit, Nachrichten weiterzuleiten, quasi als Router zu fungieren. In diesem Fall werden folglich Ende-zu-Ende Verschlüsselungen benötigt, da die dazwischenliegenden Web Services, die so genannten Intermediaries, die Nachrichten im Klartext sehen würden.
Weiterhin werden in großen Architekturen Instanzen zur Authorisierung gebraucht. Kerberos als ein oft verwendeter, verteilter Authentifizierungsdienst, ebenso wie Public-Key-Infrastruktur sind typische Vertreter, die in die Web Services integriert werden müssen. Auch hier hilft SSL nicht, denn es ist zu unflexibel und die Sicherheitssemantik wird erst auf der Anwendungsschicht abgebildet.
Damit die Interoperabilität gewährleistet werden kann, werden einheitliche Standards benötigt. Diese sind von der Organization for the Advancement of Structured Information Standards (OASIS) geschaffen worden und bilden die Grundlage der hier vorgestellten Technologien. Fachleute sind sich einig, dass diese Standards gut sind und der Web Services Technologie zu weiterem Erfolg verhelfen wird [s. (Sosnoski, 2006)].

1.1 Motivation

Inzwischen sind die Standards soweit gereift, dass die ersten Implementierungen verfügbar werden. Sowohl in Microsoft .NET als auch in Java implementierte Produkte sind bereits verfügbar und mit der Auslieferung von Windows Communication Foundation in Windows Vista wird die Verfügbarkeit noch ansteigen.
Eine der Grundansprüche an Web Servives ist die Interoperabilität. Auch die Einführung der neuen Sicherheits Standards in die Systeme, dürfen diesen Anspruch nicht zunichte machen.
Der Einsatz von zusätzlicher Sicherheits Technologien bringt auch immer die Frage mit, ob diese sich negativ auf das Perfomance Verhalten des Systems auswirken. Diese Frage möchte ich in einem typischen und möglichst realitätsnahen Szenario untersuchen.

Bei heutigen Hardware Kosten sind gewisse Performance Engpässe durch Aufstockung der Hardware zu verkraften. Jedoch muss dies planbar sein und auch in einem gewissen Rahmen bleiben. Auch wenn es theoretisch möglich ist, wird kein global agierendes Unternehmen bereit sein, die Hardware Leistung in ihrer gesamten, weltweiten IT-Landschaft erheblich zu erhöhen. Bei Performance Einbußen finden oft proprietäre Entwicklungen statt, jedoch spricht viele für den Einsatz, standardisierter Sicherheitstechnologien.
Im besonderen möchte ich das interoperable Verhalten untersuchen. Unterschiedliche Betriebssysteme und Programmiersprachen versprechen die einheitliche Bereitstellung von Sicherheitsstandards; die Vergangenheit hat jedoch gezeigt, dass dieses Versprechen nicht immer wirklich erfüllt wird, da die Implementierungen von den Standards abweichen.

1.2 Abgrenzung

Gegenüber bisherigen Untersuchungen bezüglich der Performance von Web Services Security, wie beispielsweise [Hongbin Liu (2005)] und [Kezhe Tang (2006)] möchte ich die komplexeren Standards in realistischeren Szenarien untersuchen. Bestehende Arbeiten haben sich auf einfache Lasttests mit den eher simplen Spezifikationen beschäftigt. Diese Ergebnisse berücksichtigen jedoch nicht die Komplexität von heterogenen Unternehmensanwendungen. Sie sind daher nach meiner Ansicht nur bedingt geeignet, als Entscheidungshilfe für oder gegen die Einführung der neuen Techniken zu dienen.

2 Grundlagen

Ich werde hier nur kurz auf die Grundlagen der Web Servive Sicherheitsstandards eingehen. Ein ausführlicher Überblick über die Standards und Konzepte rund um Web Service Sicherheit, beschreibe ich in meiner Ausarbeitung zu Anwendungen 2.

2.1 Web Services Security (WSS)

Als die ersten Web Services Anwendungen entstanden, wurde schnell klar, dass eine Authentfizierung auf der Applikationsschicht benötigt wird. Da Web Services per Definition zustandslos sind, wurden Username und Passwort in die Methodenaufrufe integriert. Die Integration von Kerberos und PKI in eine Web Services Landschaft war eine frühe Anforderung, die bei dem Einsatz von Web Services offensichtlich wurde [s. O'Neill (2003)]. So entstanden die ersten Web Services Security Standards, die diese Information durch einheitliche SOAP Header definierten. Dies sollte sicherstellen, dass Services verschiedener Hersteller miteinander kompatibel sind.

2.1.1 UsernameToken

Das UsernameToken ist die einfachste Form der Authentifizierung. Der Client sendet mit jeder SOAP Nachricht einen Benutzernamen und das Passwort als Feld im Header mit. Das Passwort kann und sollte als Hashwert übertragen werden, so wird verhindert, dass dritte an das Passwort gelangen können. Diese Technik ist nur im einfachsten Fall geeignet, für komplexere Strukturen eher ungeeignet.

2.1.2 BinarySecurityToken

Dieses bietet die Möglichkeit, eine Authentifizierung mittels Kerberos Tickets oder in einer PKI mit X.509 Zertifikaten durchzuführen. Auch hierfür werden SOAP-Header definiert, in denen die entsprechenden Informationen mit jedem Aufruf vom Client an den Server übertragen werden. Gerade für größere Infrastrukturen ist diese Technik wesentlich geeigneter als ein einfaches UsernameToken.

2.1.3 Signaturen und Verschlüsselungen

WSS beschreibt weiterhin die Integration von XML-Signature und XML-Encryption in die Web Services Technologie. So können Nachrichten, oder auch nur Teile davon, verschlüsselt bzw. signiert werden.

2.2 WS-*

Als WS-* wird eine Reihe von, auf WSS aufbauenden, Sicherheits Spezifikationen bezeichnet:

- WS-SecurityPolicy
- WS-Trust
- WS-SecureConversation
- WS-Federation
- WS-Authorisation
- WS-Privacy

Die für den Ausblick relevanten Standards werde ich kurz vorstellen..

2.3 WS-SecurityPolicy

WS-SecurityPolicy, oder auch kurz WS-Policy beschreibt ein Werkzeug, welches Web Services die Möglichkeit bietet, Ihre Bedingungen und Anforderungen deklarativ zu beschreiben. Diese Beschreibungen werden Policy Assertions genannt. Lose gekoppelte Systeme bekommen so die Möglichkeit, die Sicherheitsanforderungen oder Algorithmen zu wechseln, ohne die Clients explizit darüber informieren zu müssen.

WS-Policy deklariert:

- verwendete Algorithmen
- erforderliche Signaturen und Verschlüsselungen
- akzeptierte SecurityTokens

2.4 Web Services Trust Language

Die Web Services Trust Language, auch WS-Trust genannt, will Probleme lösen, die durch den Einsatz von WSS entstehen können:

- Kompatibilitätsprobleme beim Security Token Format
- Glaubwürdigkeit des Security Tokens
- Unterschiedliche Namespaces

WS Trust definiert ein Request-Response Protokoll, bestehend aus einem RequestSecurityToken (RST) und einem RequestSecurityTokenResponse (RSTR) und führt einen Security Token Service (STS) als zentrale Stelle zum erzeugen, prüfen und erneuern von Security Tokens ein. Dies birgt den Vorteil, dass sich nicht der einzelne Aufrufer um die erforderlich Sicherheits Credentials des Services kümmern muss. Der Aufrufer authentifiziert sich bei dem STS und bekommt mit dem RSTR einen Security Assertion Markup Language (SAML) Token, mit dem er den Service direkt aufrufen kann. So ist es auch möglich, dass ein Aufrufer auf einen Service zugreift, dessen Credentials er nicht erfüllen kann.
Auch auf der Seite der Services bietet dies eine wesentlich Besserung. Der Service muss keine Authorisations Prüfung mehr durchführen, sondern nur sicherstellen, dass das SAML-Token wirklich vom STS stammt. Eine grafische Übersicht zu dem Konzept von WS-Trust bietet die Abbildung 1.

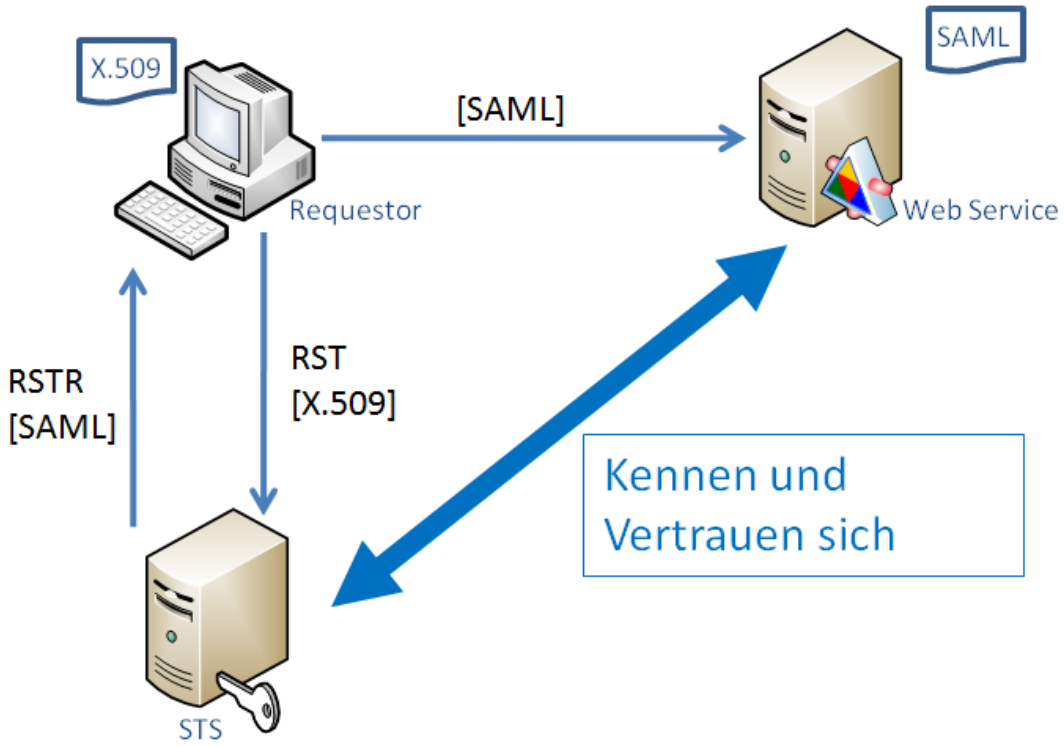

Abbildung 1: WS-Trust

2.5 WS-SecureConversation

Nach WS-Trust muss für jede Anfrage des Requestors erneut ein Token beim STS angefordert werden. Um die Last auf den STS zu reduzieren, beschreibt WS-SecureConversation einen SecurityContextToken mit dessen Hilfe die beiden Teilnehmer einen gemeinsamen Sicherheitskontext aufbauen. Von einem Basisschlüssel leiten die Teilnehmer selber Folgeschlüssel ab und sparen so die ständige Anforderung von Tokens beim STS.

2.6 WS-Federation

Identitäten sind normalerweise an einen Realm gebunden. Wenn gesichert Zugriffe über Unternehmensgrenzen hinweg ermöglicht werden sollen, entstehen besondere Herausforderungen. WS-Federation beschreibt Mechanismen die es erlauben sollen, Identitäten, Authentisierungen und Authorisierungen über Realm Grenzen hinweg nutzen zu können.

3 Performanceuntersuchungen

3.1 Überblick

Es existieren bereits einige Performance Untersuchungen zu dem Thema Web Services Security. Die bestehenden Untersuchungen berücksichtigen jedoch keine Interoperabilität und setzen nur auf die einfache Nutzung von SecurityTokens. Weiterhin sind die verwendeten Anwendungsszenarien nicht der Realität entsprechend und führen zu Ergebnisen, die in einem komplexen Umfeld kaum verwertbar sind.

3.2 Anwendungsszenario

Heutige Unternehmen haben oft IT-Strukturen, die historisch gewachsen und oftmals nur wenig integriert sind. Typischer weise sind solche Unternehmen in Divisionen unterteilt, welche bis zu einem gewissen Grad autonom agieren. Daraus resultiert, dass die in den Divisionen eingesetzten Systeme stark voneinander abweichen. Als Anwendungsszenario werde ich eine fiktive Firma, deren Struktur jedoch realem Vorbild folgt, verwenden.
Die Falkenberg AG hat Divisionen in den USA, Deutschland und Frankreich und ihren Hauptsitz in Deutschland. Als Betriebssysteme werden sowohl Unix basierte, als auch Microsoft Windows eingesetzt. Die verwendeten Programmiersprachen sind C# und Java. Das Unternehmen hat ein Data Warehouse, auf das nur die Geschäftsführer zugreifen können. Die Divisionleiter bekommen monatliche Reports per Email und haben anonsten nur durch beantragte Anfragen die Möglichkeit, Daten aus dem Data Warehouse zu bekommen.
Nun ist beschlossen worden, dass die Divisionen direkt an das Data Warehouse angebunden werden sollen. Durch die direkte Anbindung der Divisionen können die Leiter der Divisionen direkt auf das Data Warehouse zugreifen und so zeitnaher auf die Informationen reagieren. Wenn sich der Zugriff bewährt, wird darüber nachgedacht, später noch weiteren Mitarbeitern der Divisionen den Zugriff auf das Data Warehouse zu erlauben. Da bisher nur aus dem Intranet in der Hauptverwaltung auf das Data Warehouse zugegriffen wurde, hat man auf Sicherheit wenig Wert gelegt. Das Data Warehouse bietet eine erweiterbare Web Services Schnittstelle.
Die Leiter der Falkenberg AG forden die Realisierung einer flexiblen, erweiterbaren und sicheren Zugriffsmöglichkeit der Divisionsleiter auf das Data Warehouse. Um die Erweiterbarkeit auch langfristig zu gewährleisten, ist der CIO an dem Einsatz von WS-Trust zur Integration in die neue Sicherheitsarchitektur interessiert. Da schon länger überlegt wird, ob die IT-Landschaft der verschiedenen Divisionen vereinheitlicht werden sollte. Hierzu wird eine Performance Untersuchung zum verhalten der unterschiedlichen WS-Trust Implementierungen benötigt.
Diese Untersuchung möchte ich in meiner Master Thesis erarbeiten. Um WS-Trust2.4 efizient einsetzen zu können, sollte auch WS-Policy2.3 eingesetzt werden. Da WS-Policy jedoch

keine großen Auswirkungen auf die Leistung des Systems hat, werde ich mich im Kern auf WS-Trust konzentrieren.

3.3 WS-Trust Implementierungen

Die WS-Trust Spezifikation ist inzwischen in Version 1.3 [ws-trust (2006)] so weit gereift, dass bereits einige Implementierungen verfügbar sind. Die derzeit wichtigsten sind

- .NET 2.0 & WSE (Windows)
- .NET 3.0 WCF (Windows)
- Apache Axis & WSS4J (Windows/Linux)
- Sun's Project Tango (Windows/Linux)

Diese sollen hier kurz forgestellt werden.

3.3.1 .NET 2.0 & WSE (Windows)

Für das .NET Framework in der Version 2.0 gibt es die Web Services Enhancements. Diese werden zusätzlich installiert und erweitern die ASP.NET Web Services um diverse WS-* Implementierungen, unter anderem WS-Trust. Die WSEs sind sowohl unter Windows XP, als auch unter Windows Vista lauffähig. Ausführlicher werden die WSE und Ihre Funktionalität in Gailey (2004) beschrieben.

3.3.2 .NET 3.0 WCF (Windows)

Windows Communication ist eine Bestandteil von .NET 3.0 und ist in Windows Vista vorinstalliert. Es ist jedoch auch auf Windows XP installierbar und kann somit auch dort eingesetzt werden. Während seiner Entwicklung lief die WCF unter dem Code Namen *Indigo*, weitere Informationen über die Funktionsweise und Idee zu WCF sind in Pallmann (2005) zu finden.

3.3.3 Apache Axis & WSS4J (Windows/Linux)

Apache Axis ist die Standardtechnologie, um Java Servlets als Web Services verfügbar zu machen. Web Services Security for Java (WSS4J) ist eine Java Bibkiothek, die diverse WS-* Implementierungen zur Verfügung stellt. Die plattformunabhängigkeit von Java ermöglicht den Einsatz sowohl auf Unix basierten Betriebssystemen, als auch auf Microsoft Windows [s. (David A. Chappell, 2003)].

3.3.4 Sun's Project Tango (Windows/Linux)

Das Project Tango ist ein Teil des von der Firma Sun initiierten Open Source Projekts Glasfish. Dieses Subprojekt soll WS-* Implementierungen für Glasfish anbieten. Die Entwickler arbeiten nach eigener Aussage eng mit den WCF Entwicklern von Microsoft zusammen, um einen gesicherten Zugriff zu gewährleisten.

4 Ausblick

4.1 Realisierung

Für die Durchführung der Performance Untersuchung plane ich zunächst mit jeder Technologien einen simplen Client und Server zu schreiben und mit diesem Performance Tests ohne Sicherheitsfunktionen. Weder Client noch Server führen irgendwelche Logik aus, lediglich die Kommunikation soll implementiert werden. Durch diese Tests ohne Sicherheitsfunktionen erhalte ich Werte, die als Basiswerte für die späteren Messungen dienen. Es ist zu erwarten, dass bei dem interoperablem Zugriff bereits erste Abweichungen messbar werden.
Anschließend werden dieselben Tests mit verwendeten Sicherheitsfunktionen durchgeführt. Die Möglichkeit der Zugriffe zwischen unterschiedlichen Typen muss zunächst geprüft werden. Ist dieser möglich, so kann in den entsprechenden Kombinationen die Ergebnisse der Tests interessante Informationen darüber geben, wie sehr die Hersteller abseits der Standards optimiert haben, um ihr System im homogenen Einsatz performanter erscheinen zu lassen. Da die Nachrichten bei Web Services Kommunikation in klartext-XML übertragen werden, lässt sich schnell herausfinden, welche der Implementierungen vom Standard abweicht - wer also Schuld hat.
Ich plane die Tests mit virtuellen Maschinen durchzuführen. Virtuelle Maschinen bieten mit der Snapshot Technologie die Möglichkeit, wiederherstellbare Zustände zu verwenden. Durch diese Snapshots kann jeder Test unter identischen Rahmenbedingungen durchgeführt werden und die Ergebnisse werden vergleichbarer.

4.2 Ziel

Für meine Master Arbeit möchte ich die zuvor vorgestellten Implementierungen untersuchen. Besonderes Augenmerk liegt auf dem interoperablen Performance Verhalten von WS-Trust Implementierungen. Teil der Arbeit soll ebenfalls sein, die Ergebnisse zu interpretieren bzw. deren Ursache zu finden.
Die Arbeit soll Entscheider dabei unterstützen, die zu erwartenden Risiken abzuwägen und Auswirkungen auf das System bei der Einführung der Sicherheitsfunktionen zu schätzen.
Natürlich kann diese Arbeit kein all umfassender Almanach für die Integration von Web Services Security werden, wird aber durchaus eine praktisch nutzbare Erfahrungsbasis für indi-

viduelle Tests sein. Der Mehrwert gegenüber den bereits aufgeführten, bestehenden Arbeiten liegt darin, dass die Messungen in wesentlich realeren Szenarien betrieben werden und somit in der Praxis auch für komplexere Szenarien verwendet werden kann.

4.3 Risiken

Als Risiko ist sicher die Kompatibilität zwischen den einzelnen Implementierungen zu sehen. Es wäre nicht das erste mal, wenn Anbieter von Standard Implementierungen von eben selbigen abweichen, um ihre Produkte performanter erscheinen zu lassen.
Auch der Aufbau einer Versuchsumgebung ist ein durchaus nicht zu unterschätzender Zeitfaktor und muss daher als Risiko bei der Erstellung der Master Thesis betrachtet werden.
Die WS-* Standards sind offen und gut dokumentiert, da die Kommunikation im Klartext statt findet, sehe ich den Teil der Informationsgewinnung nicht als Risiko an.

Literatur

[ws-trust 2006] : *OASIS WS-Trust 1.3*. 2006. – URL http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-rddl.html

[Cyrus Peikari 2004] CYRUS PEIKARI, Anton C.: *Security Warrior*. O Reilly, 2004

[David A. Chappell 2003] DAVID A. CHAPPELL, Tyler J.: *Java Web Services*. O Reilly, 2003

[Eckert 2004] ECKERT, Claudia: *IT Security*. Oldenburg, 2004

[Gailey 2004] GAILEY, Jeannine H.: *Understanding Web Services Specifications and the WSE*. Microsoft Press, 2004

[Hongbin Liu 2005] HONGBIN LIU, Geoffrey F. (Hrsg.) ; Community Grids Lab, Indiana University (Veranst.): *Performance of Web Services Security*. URL http://www.mardigrasconference.org/2005/Presentations/Liu.pdf, 2005

[Kezhe Tang 2006] KEZHE TANG, Levy D. Zic J. Yan B.: A Performance Evaluation of Web Services Security / IEEE. URL http://ieeexplore.ieee.org/iel5/4031176/4031177/04031196.pdf?tp=&arnumber=4031196&isnumber=4031177, 2006. – Forschungsbericht

[O'Neill 2003] O'NEILL, Mark: *Web Services Security*. Mcgraw-Hill Professional, 2003

[Pallmann 2005] PALLMANN, David: *Programming Indigo*. Microsoft Press, 2005

[Sosnoski 2006] SOSNOSKI, Dennis: The year ahead in Java Web services / IBM. URL http://www.ibm.com/developerworks/java/library/ws-java1.html, 2006. – Forschungsbericht