Business Technology

www.bt-magazin.de 3.2010 Deutschland: € 14,90 Österreich: € 16,80 Schweiz: sFr 29,80 Luxemburg: € 16,90

Business Technology

Architektur & Management Magazin

Expertenwissen für IT-Architekten, Projektleiter und Berater

Winter:
„Die Welt steht nicht still."

PROZESSE

Darf's ein bisschen weniger sein?
Nutzen und Grenzen von Geschäftsprozess-management

BPM außerhalb der Verwaltung
Ein Blick über den Tellerrand

Der Mensch ist (k)ein asynchroner Service
Gefahren der Übermodellierung – notwendige Grenzen des BPM

Der Roundtrip mit BPMN 2.0
Methodik und Projektvorgehen – Roundtrip mit Köpfchen

Enterprise SOA Security
Teil 3: Standards

BPMN: GOOD NEWS, BAD NEWS

DER ARCHITEKTUR-PROZESS IN DER IT

AGILITÄT TROTZ KOMPLEXER ARCHITEKTUREN

03
4 191858 414906

Standards, Teil 3

Enterprise SOA Security

AUTOREN: DR. DIRK KRAFZIG, JOST BECKER, OLIVER MAHNKE UND MARKUS MAZUR

Im ersten Teil dieser Artikelserie haben wir begründet, warum im Zuge der Transformation traditioneller IT-Landschaften hin zu einer serviceorientierten Architektur neue Securitykonzepte nötig sind. Die Auslagerung von Querschnittsfunktionalität in Services sowie die automatisierte Ausführung unternehmensweiter Geschäftsprozesse stellen neue Herausforderungen dar, die durch die Securitykonzepte traditioneller Silo-Lösungen nicht angemessen realisierbar sind. So ist Security zukünftig transparent, rollenbasiert, unternehmensweit einheitlich und standardisiert umzusetzen, um auf Unternehmensebene beispielsweise die durch relevante Compliance-Vorgaben geforderte Transparenz und Kontrolle zu schaffen, Single Sign-on einfach realisieren oder auch zusätzliche Kunden mit geringen Zusatzkosten anbinden zu können.

Im zweiten Teil dieser Serie haben wir konkrete Lösungsmuster vorgestellt, die die Realisierung von Security in verteilten Systemen ermöglichen. Das Prinzip des Security Tokens wurde als Basismuster verteilter Security identifiziert. Auf dieser Grundlage wurden Mechanismen wie Single Sign-on (SSO), frühe und späte Berechtigungsprüfungen, ESB-basierte Security sowie verschiedene Aspekte von domainübergreifender Security näher beleuchtet.

Im vorliegenden dritten Teil unserer Serie vertiefen wir die Betrachtung der relevanten technischen Standards, die uns helfen, die beschriebenen Lösungsmuster umzusetzen. Um die Sicherheitsinfrastruktur nicht an spezifische Produkthersteller- oder Lösungstechnologien zu binden, ist eine auf offenen Standards basierende Umsetzung der im zweiten Teil unserer Serie vorgestellten Lösungsmuster entscheidend.

Auf Web Services basierende Implementierungen serviceorientierter Architekturen, die im Fokus dieses Teils stehen, werden durch eine Vielzahl an Standards

Vierteilige Reihe: Enterprise SOA Security

Teil 1: Herausforderungen Erschienen in Ausgabe BT 1.10	Der erste Teil befasst sich mit den Herausforderungen heutiger IT-Landschaften. Abteilungsübergreifende, integrierte Lösungen und die Öffnung von Kernsystemen für Kunden und Lieferanten überfordern häufig traditionelle Sicherheitslösungen.
Teil 2: Lösungsmuster Erschienen in Ausgabe BT 2.10	Im zweiten Teil der Serie werden Lösungsmuster vorgestellt, mit deren Hilfe moderne SOA Security konzipiert werden kann. Im Zentrum der Diskussion stehen Security Token, die in einer verteilten Umgebung sicherstellen, dass jede Komponente gesicherte Annahmen über ihre Nutzer und deren Berechtigungsprofile treffen kann.
Teil 3: Web-Services-Standards Vorliegende Ausgabe	Zahlreiche Web-Services-Standards wie SAML, WS-Security, XACML oder WS-Trust helfen bei der Umsetzung interoperabler Sicherheitslösungen in einer SOA. Der dritte Teil gibt einen Überblick über existierende Standards und wie sie in der Praxis angewandt werden.
Teil 4: Organisatorische Maßnahmen Erscheint in Ausgabe BT 4.10	SOA Security ist keinesfalls ein reines Technologiethema. Abteilungsübergreifende Prozess- und Rollenkonzepte erfordern auch abteilungsübergreifende Governance und ein Umdenken in den Risikoabteilungen großer Unternehmen.

im Bereich Security unterstützt. Die große Anzahl an sich ist aber kein Grund für Verunsicherung, da sich die Standards bei näherer Betrachtung in vier wesentliche Bereiche gliedern lassen (**Abb. 1**).

Der erste Bereich befasst sich mit Security Tokens. Ein Security Token ist eine kleine Datei bzw. ein Fragment einer Datei, das sicherheitsrelevante Daten über Web Service Consumer bzw. über Personen, die mit ihnen arbeiten, enthält. Dies kann z. B. die Adresse des ursprünglichen Absenders einer Nachricht sein oder die Rolle des Benutzers im Geschäftsprozess. Der zweite Bereich umfasst die Kommunikation zwischen Web-Service-Providern und -Consumern. Hier wird beschrieben, wie Provider und Consumer sicherheitsrelevante (Meta-)Daten austauschen und wie Nachrichten signiert und verschlüsselt werden können. Im dritten Bereich befassen wir uns mit der Kommunikation zwischen Serviceprovider und Security Token Service. Hier geht es insbesondere darum, wie ein Service-Consumer ein Token erhalten kann. Im letzten der vier Bereiche behandeln wir die Auswertung der Security Token beim Serviceprovider, bei dem die Berechtigungsprüfung vorgenommen wird.

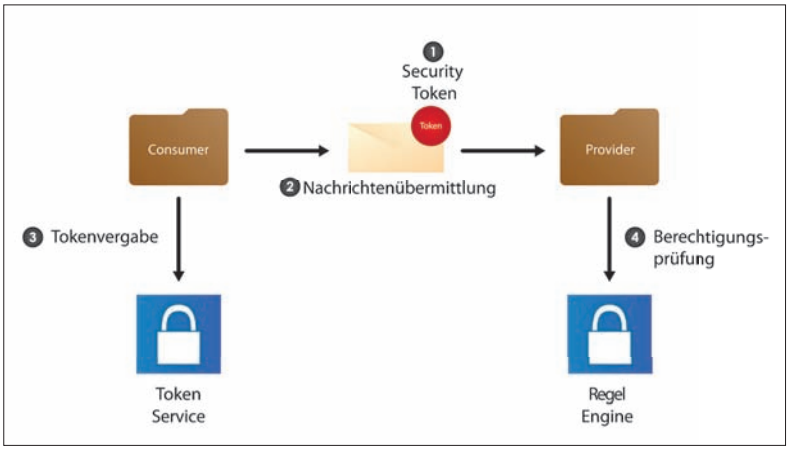

Abb. 1: Standards für Web-Service-Security gliedern sich in vier Bereiche

1. SECURITY TOKEN

Silo-Anwendungen in gewachsenen Anwendungslandschaften sind typischerweise durch den Aufbau technischer Sessions und die Integration von Middleware-Komponenten, z. B. spezifische Datenbanktreiber, eng an die zu nutzenden Ressourcen gebunden. Häufig speichert jede einzelne dieser Anwendungen zusätzlich die Identitäten ihrer Nutzer und deren anwendungsspezifische Berechtigungen. Das Kernprinzip verteilter Security ist demgegenüber, die in einer Anwendungslandschaft gültigen Nutzer sowie alle sicherheitsrelevanten Informationen zentral durch einen Identity-Provider zu verwalten. Security Token transportieren Nachweise von Identitäten sowie Zusicherungen bezüglich deren spezifischer Eigenschaften. Ausgegeben werden sie durch eine zentrale Vergabestelle, den so genannten Security Token Service (STS), der eng mit dem Identity-Provider kooperiert.

Es existierten vielfältige Standards zur Erstellung von Security Tokens. Sie beschreiben konkret, wie Zusicherungen, kryptografische Schlüssel oder Zertifikate übertragen werden können. Hervorzuheben sind u. a. so genannte User Name Tokens [1], X.509-basierte Tokens [2] oder auch SAML- [3], Kerberos- [4] und REL-Tokens [5].

SAML ist der wichtigste dieser Standards, der z. B. vom Bundesministerium des Innern für die Verwendung im deutschen E-Government empfohlen wird [6]. SAML definiert eine XML-basierte Sprache zur Formulierung von sicherheitsrelevanten Aussagen in Bezug auf ein so genanntes Subject, das beispielsweise einen Nutzer repräsentieren kann, sowie ein Token-Format zur Verpackung dieser Informationen. Die Aussagen können sich dabei auf eine bereits erfolgte Authentifizierung beziehen, auf Eigenschaften eines Nutzers oder auch auf Entscheidungen, einem Nutzer Zugriff auf eine spezifische Ressource zu gewähren.

Die Übertragung eines SAML-Tokens ist durch eine Vielzahl von Protokollen möglich, die der SAML-Standard in Form so genannter Binding-Spezifikationen definiert. Diese reichen über einen einfachen Austausch eines Security Tokens in Form eines Request-Response-Musters bis hin zu Logout-Protokollen in Single-Sign-on-Szenarien. Wir beschränken uns hier auf die Verwendung von SAML-Tokens im Rahmen des unten beschriebenen WS-Security-Standards.

Gebräuchliche Versionen des Standards sind SAML 1.1 [7] und SAML 2.0 [8], die im Jahr 2005 von der ausgebenden Organisation OASIS ratifiziert wurden. Das neuere SAML 2.0 bietet dabei durch eine stärkere Modularisierung und eine erweiterte Anzahl von Protokollen unter anderem Vorteile bei Architekturszenarien, die auch verteilte Identitäten beinhalten. Als kleines Beispiel, um die Formulierung von Zusicherung als SAML-Token zu demonstrieren, sei angenommen, dass der Benutzer *Karl Krach* in einem Versicherungskonzern die Rolle *Schadensbearbeiter* inne hat. Zusätzlich ist er fachlich für die Sparten *KFZ* und *HAUS* zuständig. Diese Eigenschaften können nun mittels SAML-Assertions formuliert und als SAML-Token verpackt werden.

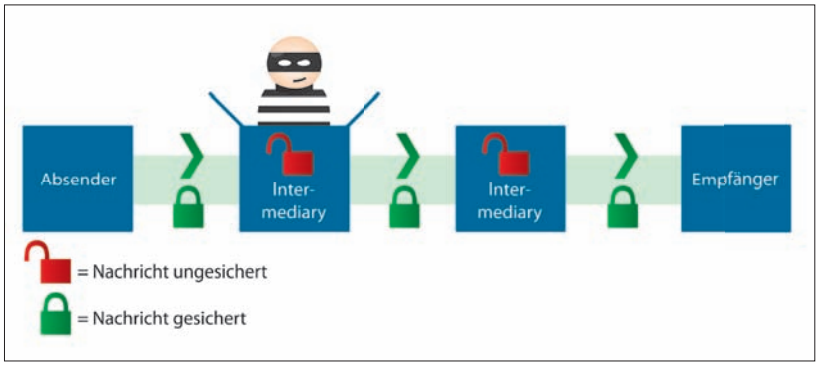

Abb. 2: In vielschichtigen und verteilten Architekturen ist Schutz auf Transportebene oft unzureichend. ESBs, Router oder Gateways sind Angriffspunkte

Listing 1 zeigt ein SAML-Token, dass diese Informationen zusichert. Das äußere Assertion-Element repräsentiert dabei das Security Token, das Informationen über den Nutzer im Subject- bzw. NameID-Element beinhaltet sowie eine Aussage über dessen Eigenschaften in den Elementen *AttributeStatement*, *Attribute* bzw. *AttributeValue*.

Listing 1

Beispiel eines SAML-Tokens

```
<saml:Assertion
 xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion" Version="2.0"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 IssueInstant="2010-08-12T10:00:00Z">
 <saml:Issuer>meinVersicherungskonzern.de</saml:Issuer>
 <saml:Subject>
<saml:NameID format="urn:oasis:names:tc:SAML:2.0:nameid-format:persistent">
   Karl Krach
</saml:NameID>
</saml:Subject>
<saml:Conditions
 NotBefore="2010-08-12T10:00:00Z" NotOnOrAfter="2010-08-12T16:30:00Z">...
</saml:Conditions>
 <saml:AttributeStatement>
  <saml:Attribute Name="Rollen">
<saml:AttributeValue xsi:type="xs:string">
 Schadenssachbearbeiter
</saml:AttributeValue>
  </saml:Attribute>
  <saml:Attribute Name="Sparten">
<saml:AttributeValue xsi:type="xs:string">
 KFZ, HAUS
</saml:AttributeValue>
  </saml:Attribute>
 </saml:AttributeStatement>
</saml:Assertion>
```

DEFINITION VON SICHERHEITS-ANFORDERUNGEN

Ein Service kann Anforderungen an seine Consumer, so genannte Policies, als Teil seines Servicevertrags veröffentlichen. Auch die hier betrachteten sicherheitsrelevanten Anforderungen können so potenziellen Nutzern mitgeteilt werden. Sie werden damit in die Lage versetzt, die geforderten Informationen vorab zu beschaffen und dem Serviceprovider beim Aufruf in Form eines Security Tokens mitzuteilen. Die Formulierung allgemeiner Anforderungen wird durch die Spezifikation WS-Policy [9] abgedeckt. Sie beschreibt ein generisches Framework zur Formulierung von Policies für Web Services. Auf dieser Grundlage definiert WS-Security Policy mögliche sicherheitsrelevante Policies, mit denen Web Services entsprechende konkrete Anforderungen an ihre Nutzer beschreiben können. Beispielsweise kann ein Service mittels einer Policy verlangen, dass der Request eines Consumers ein SAML-Token transportieren muss, das bestimmte Claims, also Zusicherungen über eine Menge von Eigenschaften des Anfordernden, beinhaltet, oder auch eine Verschlüsselung definierter Nachrichtenteile fordern.

2. SICHERE NACHRICHTEN

Nachdem mit dem Security Token die Grundlage zur Realisierung verteilter Security etabliert wurde, muss nun auch die sichere Übertragung sämtlicher sicherheitsrelevanter Informationen gewährleistet werden. Sowohl Integrität und Vertraulichkeit des Security Tokens als auch der relevanten Nutzdaten müssen sichergestellt sein, d. h. diese Informationen dürfen während der Übertragung weder verfälscht werden noch dürfen sie für Nichtberechtigte zugreifbar sein. Die Basis für die sichere Übertragung von Nachrichten liefert WS-Security [10]. Im Gegensatz zur Sicherheit auf der Transportebene bietet WS-Security Sicherheit auf der Nachrichtenebene, die es erlaubt, nicht nur eine Kommunikationsstrecke als Ganzes abzusichern, sondern selektiv auch einzelne Bestandteile einer zu übertragenden Nachricht (**Abb. 2**).

WS-Security beschreibt dazu konkret, wie Security Tokens und die nötigen Schlüssel im Header von SOAP-Nachrichten eingebettet und zwischen Web-Service-Providern und -Consumern ausgetauscht werden können. Die genannten Standards für Security Tokens werden dabei durch unterschiedliche Profile [1], [2], [3], [4], [5] unterstützt. WS-Security liefert zusätzlich die nötigen

Verfahren, um die Vertraulichkeit, Integrität und Verbindlichkeit der übertragenen Nachrichten zu gewährleisten.

Die Integrität von Inhalten einer SOAP-Nachricht kann durch Verfahren zur digitalen Signatur überprüft werden. Der entsprechende Nachrichtenteil wird dazu von seinem Produzenten mit einer eindeutigen digitalen Signatur versehen, die in der Regel dem verschlüsselten Hash-Wert des Nachrichtenteils entspricht. Ein Security Token wird beispielsweise durch den ausgebenden STS verschlüsselt, der dazu seinen privaten Schlüssel nutzt. Der Body einer Nachricht, die einen Serviceaufruf repräsentiert, wird hingegen typischerweise durch den Servicenutzer selbst signiert. Er kann dazu beispielsweise seinen privaten Schlüssel nutzen, falls der korrespondierende öffentliche Schlüssel dem aufgerufenen Service bekannt ist.

Der Empfänger einer Nachricht kann die Integrität der Nachrichteninhalte verifizieren, indem er deren assoziierte Signaturen prüft. Er entschlüsselt dazu den verschlüsselten Hash-Wert des zu prüfenden Nachrichtinhalts, der in Form einer Signatur vorliegt, und vergleicht ihn mit dem von ihm selbst berechneten tatsächlichen Hash-Wert. Eine Übereinstimmung des berechneten Hash-Werts mit dem Hash-Wert aus der Signatur bestätigt, dass der Nachrichteninhalt während des Transports nicht modifiziert wurde. Um beispielsweise die Integrität eines Security Tokens zu überprüfen, wird der öffentliche Schlüssel des STS verwendet werden, der das Token ausgestellt und mit seinem privaten Schlüssel signiert hat. Listing 2 zeigt, wie die oben beispielhaft dargestellte SAML-Assertion bzw. das Security Token mittels WS-Security in den Header einer SOAP-Nachricht eingebettet werden.

3. SECURITY TOKEN SERVICE

Neben einer standardisierten Form des Security Tokens und der sicheren Übertragung von Nachrichten ist auch eine standardisierte Kommunikation mit dem Security Token Service (STS) für den Aufbau einer verteilten Sicherheitsinfrastruktur notwendig, deren Komponenten nicht durch die Verwendung herstellerspezifischer Technologien oder durch unternehmensspezifische Insellösungen stark aneinander gekoppelt sind. An dieser Stelle kommt der WS-Trust-Standard zum Einsatz, der eine durch den STS zu implementierende standardisierte Schnittstelle beschreibt. Diese bietet Operationen zur Ausstellung, zum Abruf sowie zur Validierung von Security Token. WS-Trust [11] unterstützt dabei prinzipiell unterschiedliche Token-Typen, unter anderem auch SAML-Token. Der Typ des gewünschten Tokens, die Zusicherungen sowie die Art der Verschlüsselung müssen dabei in der entsprechenden Anfrage (Request for Security Token, RST) angegeben werden.

Zusätzlich zur Token-Vergabe unterstützt WS-Trust die Teilnehmer einer Sicherheitsinfrastruktur darin, den im Rahmen der Kommunikation zu nutzenden Sicherheitsmechanismus auszuhandeln. WS-Trust ermöglicht dazu die Nutzung des so genannten SPNEGO-Protokolls [12], das einen solchen Mechanismus bietet. Listing 3 zeigt, wie von einem STS ein Token angefordert wird, das die Assertions unseres Beispielnutzers enthält. Das XML-Fragment beinhaltet die Eingabeparameter der entsprechenden Serviceoperation (nicht abgebildet ist der SOAP-Header der Anfrage).

4. BERECHTIGUNGSPRÜFUNGEN

Zusätzlich zum Konzept des Security Tokens und den Mechanismen zur sicheren Übertragung sind in einer verteilten Securityinfrastruktur Elemente zur konkreten Durchführung von Berechtigungsprüfungen nötig. Zusätzlich ist zu klären, wie diese Elemente miteinander kommunizieren und in welcher Form die Regeln zur Prüfung von Berechtigungen beschrieben werden. Diese Aspekte wollen wir im Folgenden näher betrachten.

Listing 2

Beispiel einer Nachricht mit eingebettetem SAML-Token

```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2005/05/soap-envelope">
 <soap:Header>
  <wsse:Security>
   <saml:Assertion ...>
    ...
   </saml:Assertion>
  </wsse:Security>
 </soap:Header>
 <soap:Body>...</soap:Body>
</soap:Envelope>
```

Listing 3

Beispielanfrage an den STS nach Ausstellung eines Security Tokens

```
<wstrust:RequestSecurityToken>
 <wstrust:TokenType>
  http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLV2.0>
 </wstrust:TokenType>
 <wstrust:RequestType>
  http://schemas.xmlsoap.org/ws/2005/02/trust/Issue
 </wstrust:RequestType>
</wstrust:RequestSecurityToken>
```

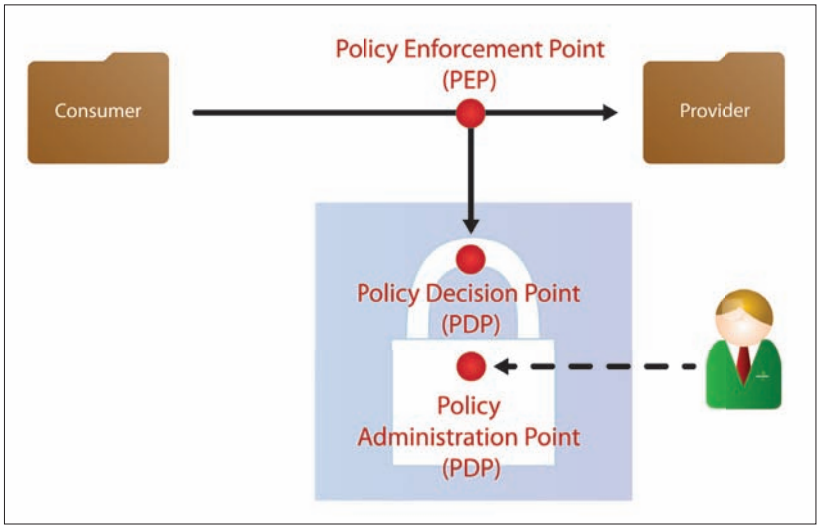

Abb. 3: PEP und PDP interagieren zur Laufzeit bei der Auswertung von Berechtigungsanfragen

Im zweiten Teil dieser Serie haben wir das Konzept der Regel-Engine zur Berechtigungsprüfung beschrieben, die auch als Policy Decision Point (PDP) bezeichnet wird. Der PDP entscheidet, ob ein gewünschter Zugriff auf eine Ressource zulässig ist oder nicht. Der so genannte Policy Enforcement Point (PEP) hingegen stellt sicher, dass vor einem solchen Zugriff eine Berechtigungsprüfung durch den PDP auch tatsächlich durchgeführt wird. Nur im Fall einer positiv erfolgten Berechtigungsprüfung wird der Zugriff gewährt. Der PEP kann durch den Serviceprovider realisiert werden oder durch ein separates Architekturelement (**Abb. 3**).

In diesem Kontext ist der XACML-Standard der OASIS [13] hervorzuheben, der momentan in den Versionen 1.0, 1.1, 2.0 und 3.0 vorliegt. Er definiert eine XML-basierte Sprache, die vorrangig der Beschreibung von Regeln bzw. Policies zur Steuerung des Zugriffs auf eine Ressource dient. Auf diese Weise kann ausgedrückt werden, welche Entität unter welchen Umständen bzw. aufgrund welcher Eigenschaften zum Zugriff autorisiert ist.

Außerdem erlaubt es XACML, eine Anfrage nach der Zulässigkeit eines Zugriffs auf eine Ressource sowie eine entsprechende Antwort zu formulieren. XACML kann damit in der Kommunikation zwischen PEP und PDP verwendet werden, um einen Zugriff zu autorisieren oder eben nicht.

Ein konkretes Beispiel für leichtgewichtige PEPs stellen die so genannten Fedlets der OpenAM-Technologie (ehemals OpenSSO von Sun) dar. Sie versetzen einen Serviceprovider in die Lage, in einer SAML-basierten verteilten Securityinfrastruktur an SSO-Szenarien teilzunehmen. Das Fedlet ermöglicht es dem Serviceprovider, von einem Identity-Provider Zusicherungen in Form von SAML-Assertions in Empfang zu nehmen und im Zusammenspiel mit einem PDP auf Basis von XACML eine Berechtigungsprüfung durchzuführen. Listing 4 zeigt, wie mittels XACML eine Regel beschrieben werden kann, die unserem Nutzer Karl Krach aufgrund seiner Rolle als Schadensbearbeiter einen lesenden Zugriff auf die von ihm angeforderten Daten eines Schadenfalls gewährt.

Listing 4

Beispielregel, die für die Rolle Schadensbearbeiter lesenden Zugriff auf einen Schadensfall gewährt

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Policy xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os" ... Version="2.0"
                                                    PolicyId="SchadenfallPolicy">
<Target>
<Resources> <!-- Spezifikation der Ressource Schadenfall, auf
                                    die zugegriffen werden soll --> ... </Resources>
</Target>
<Rule Effect="Permit" RuleId="LeseSchadenfallRegel">
    <Target>
      <Actions>
<Action>
 <ActionMatch MatchId="string-equal">
<AttributeValue DataType="string">read</AttributeValue>
<ActionAttributeDesignator DataType="string" AttributeId="action-id"/>
</ActionMatch>
</Action>
</Actions>
</Target>
    <Condition>
<Apply FunctionId="string-equal">
<AttributeValue DataType="string">
 Schadensbearbeiter
</AttributeValue>
<SubjectAttributeDesignator DataType="string" AttributeId="role" />
</Apply>
</Condition>
</Rule>
</Policy>
```

ZUSAMMENFASSUNG UND AUSBLICK

In diesem Teil unserer Serie haben wir gezeigt, wie sich auf Basis aktueller Standards Konzepte zur verteilten Security in Web-Services-basierten Implementierungen

serviceorientierter Architekturen erfolgreich umsetzen lassen. Zu diesem Zweck haben wir die Vielzahl existierender Spezifikationen eingeordnet und gezeigt, wie diese im Zusammenspiel genutzt werden können. Über die Formulierung von Security Token – dem Basiskonzept verteilter Security – hinaus, lassen sich so auch umfangreiche Szenarien realisieren, die durch eine Aufteilung der Zuständigkeiten zur Sicherstellung sowie zur Überprüfung von Zugriffsberechtigung auf so genannte Policy Enforcement (PEP) sowie Policy Decision Points (PDP) charakterisiert sind und auf diese Weise eine hohe Flexibilität bieten. Der folgende und gleichzeitig abschließende Teil dieser Serie wird sich den organisatorischen Maßnahmen widmen, die eine erfolgreiche Umsetzung von Securitykonzepten in einer verteilten IT-Landschaft über die rein technologischen Herausforderungen hinaus maßgeblich bestimmen.

Dr. Dirk Krafzig ist Gründer von SOAPARK. Als Sprecher auf Konferenzen und Autor von Artikeln und Büchern gilt Dr. Krafzig als ein Protagonist der serviceorientierten Architektur (SOA) und hat maßgeblich zu der Begriffsbildung in diesem Bereich beigetragen. Insbesondere die SOA-Fallstudien mit Deutsche Post, Credit Suisse, Halifax Bank of Scotland und Winterthur Versicherung in seinem Bestseller „Enterprise SOA" (Prentice Hall, 2004) haben viel Aufmerksamkeit auf sich gezogen. Derzeit arbeitet Dr. Krafzig in einem strategischen SOA-Programm bei einem Mobilfunkanbieter an dem Thema Security.

Jost Becker ist gemeinsam mit Ilja Pavkovic und Oliver Mahnke Gründer der binaere bauten gmbh. Sein Interesse gilt parallel zu informationstechnologischen Themen seit jeher ökonomischen und organisatorischen Aspekten im Unternehmen – eine Kombination, die sich in seinem aktuellen Tätigkeitsfeld als Berater für Software- und IT-Unternehmensarchitekturen auszahlt. Aktuell richtet er seinen Fokus auf die speziellen Securityanforderungen in verteilten Systemen.

Oliver Mahnke ist Chefarchitekt der binaere bauten gmbh. In dieser Rolle verantwortet er sowohl Anwendungs- als auch Unternehmensarchitekturen. Derzeit erarbeitet er die zentralen Architekturrichtlinien für eines der größten deutschen Versicherungsunternehmen.

Markus Mazur arbeitet als SOAPARK-Berater in Repository-Projekten. Als Mittelsmann zwischen IT und Fachbereich kennt er die unterschiedlichen Bedürfnisse der verschiedenen Anwendergruppen und ist aktiv an der Gestaltung der Repositories beteiligt. Überzeugt davon, dass ein SOA Repository zentrales Planungs-, Überwachungs- und Kommunikationswerkzeug zum Aufbau einer SOA ist, beschäftigt er sich mit der Frage, wie das perfekte SOA Repository aussehen sollte. Sein zweites Thema ist die Servicemodellierung, die Markus Mazur vor seinem Hintergrund als Betriebswirt aus einer sehr fachlichen Sichtweise angeht. Er vertritt dieses Thema auch im Rahmen des SOAPARK-Schulungsprogramms.

Links & Literatur

[1] Web Services Security, UsernameToken Profile 1.1, OASIS Standard Specification, 1 February 2006: http://www.oasis-open.org/committees/download.php/16782/wss-v1.1-spec-os-UsernameTokenProfile.pdf [PDF]

[2] Web Services Security, X.509 Certificate Token Profile 1.1, OASIS Standard Specification, 1 February 2006: http://www.oasis-open.org/committees/download.php/16785/wss-v1.1-spec-os-x509TokenProfile.pdf [PDF]

[3] Web Services Security, SAML Token Profile 1.1, OASIS Standard, 1 February 2006: http://www.oasis-open.org/committees/download.php/16768/wss-v1.1-spec-os-SAMLTokenProfile.pdf [PDF]

[4] Web Services Security, Kerberos Token Profile 1.1, OASIS Standard Specification, 1 February 2006: http://www.oasis-open.org/committees/download.php/16788/wss-v1.1-spec-os-KerberosTokenProfile.pdf [PDF]

[5] Web Services Security, Rights Expression Language (REL) Token Profile 1.1, OASIS Standard, 1 February 2006: http://www.oasis-open.org/committees/download.php/16687/oasis-wss-rel-token-profile-1.1.pdf [PDF]

[6] Standards und Architekturen für E-Government-Anwendungen, SAGA 4.0, Bundesministerium des Inneren, March 2008: http://www.CIO.bund.de/cae/servlet/contentblob/77116/publicationFile/3995/saga_4_0_download.pdf [PDF]

[7] Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V1.1, OASIS Standard, 2 September 2003: http://www.oasis-open.org/committees/download.php/3406/oasis-sstc-saml-core-1.1.pdf [PDF]

[8] Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0, OASIS Standard, 15 March 2005: http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf [PDF]

[9] Web Services Policy 1.5 Framework, W3C Recommendation, 04 September 2007: http://www.w3.org/TR/ws-policy/ws-policy-framework.pdf [PDF]

[10] Web Services Security, SOAP Message Security 1.1 (WS-Security 2004), OASIS Standard Specification, 1 February 2006: http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf [PDF]

[11] WS-Trust 1.4, OASIS Standard, 2 February 2009: http://docs.oasis-open.org/ws-sx/ws-trust/v1.4/ws-trust.pdf [PDF]

[12] The Simple and Protected GSS-API Negotiation Mechanism (SPNEGO), The Internet Engineering Task Force (IETF), December 1998: http://www.ietf.org/rfc/rfc2478.txt [TXT]

[13] eXtensible Access Control Markup Language (XACML) Version 2.0, OASIS Standard, 1 February 2005: http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf [PDF]