# Sicherheit in verteilten Web-Applikationen durch aspektorientierte Programmierung

Nicolai Kuntze, Thomas Rauch, Andreas U. Schmidt

Fraunhofer-Institut für sichere Informationstechnologie SIT
Dolivostraße 15
64293 Darmstadt
{Nicolai.Kuntze,Thomas.Rauch,Andreas.U.Schmidt}@sit.fraunhofer.de

**Zusammenfassung:** Identity Management erlangt eine immer größere Bedeutung, da immer mehr Firmen ihre IT-Systeme für Partner, Lieferanten oder Kunden öffnen. Die Diplomarbeit stellt einen Ansatz vor, durch den ein Zugriffskontroll- und Authentifikationssystem modular zu bestehenden webbasierten Systemen hinzugefügt werden kann. Das System muss hierfür nicht im Sourcecode vorliegen. In einer Trainingsphase wird das Modul an die zu schützende Applikation angepasst. Dies wird durch den Einsatz von aspektorientierter Programmierung in Form des AspectJ Frameworks erreicht. Für die Authentifikation kommt das Liberty Alliance Protokoll, ein zukünftiger Industriestandard, zum Einsatz, welches ein Single-Sign-On Protokoll unter Verwendung von Identity Federation implementiert. Eine mögliche Verwendung von Hardwaretoken in diesem Framework wird demonstriert und eine Weg vorgestellt, ein proaktives Verhalten in das Systems zu integrieren.

Die Auslagerung der IT-Infrastruktur hat sich zu einer verbreiteten Technik für Firmen und Behörden entwickelt, die unter dem Druck der Rationalisierung zentrale Teile ihrer Abläufe an externe Dienstleister vergeben. Dies ist unter Sicherheitsüberlegungen als kritisch zu beurteilen, besonders, wenn Mitarbeiter externer Firmen und Aussendienstmitarbeiter beschränkten aber direkten Zugriff auf die firmeninternen Daten erlangen [1, 2]. So hat die Distributed Management Task Force einen umfassenden Kanon an Standards veröffentlicht, welcher durch SUN im Rahmen des Toolkit für „Web-based Enterprise Management“ implementiert wird [3]. Aber auch unter Verwendung dieser Frameworks ist ein beträchtlicher Aufwand für die Anwendung dieser expliziten Sicherheitspolitik auf ein komplettes System, besonders wenn bestehende Systeme eingebunden werden müssen[4]. Dies vereint sich besonders bei kleinen und mittleren Unternehmen mit dem Verlangen nach Effizienz bei der Auslagerung der Geschäftsprozesse.

Im Folgenden wird ein Ansatz und dessen prototypische Umsetzung vorgestellt, in dem eine bestehende Web-Dienstleistung mit den drei fundamentalen Sicherheitseigenschaften Authentifikation, Autorisation und Accounting erweitert wird (AAA). Die besonderen Vorteile der vorgestellten Methode stellen sich in ihrer Universalität, Flexibilität und Skalierbarkeit dar. Es werden frei verfügbare und erprobte Technologien unter Verwendung der *aspektorientierten Programmierung* [5] kombiniert und nur ein minimales Wissen über das zu schützende System, besonders den Sourcecode, voraussetzt.

In Abbildung 1 wird die Architektur vorgestellt, in der die Wirtsapplikation beim Service Provider (SP) mit dem generischen Sicherheitsmodul, dem *security aspect*, unter Verwen-

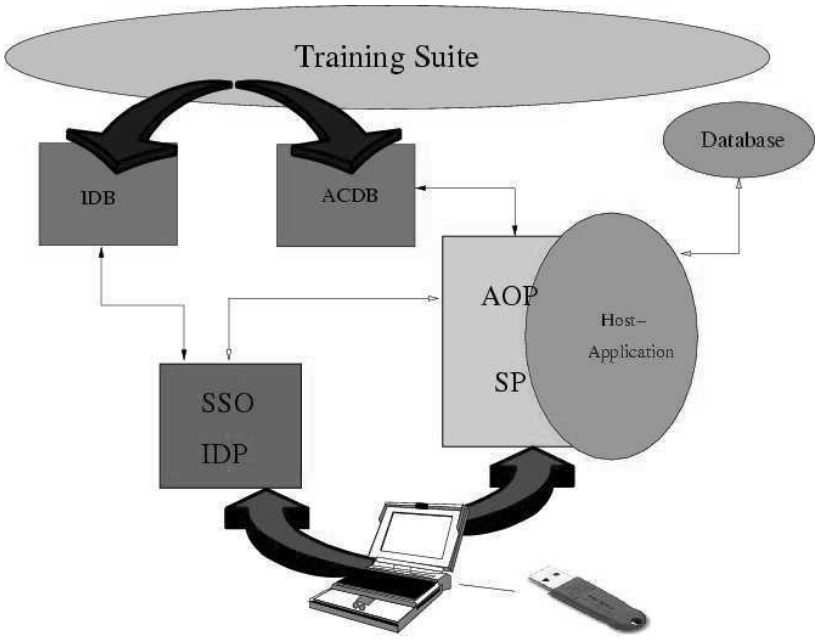


**Abbildung 1:** Aspektorientierte Sicherheitsarchitektur

dung des AspectJ Frameworks verwoben wird. Das Verweben findet an definierten Punkten statt, durch die das Modul Kontrolle über alle für eine AAA-Funktionalität wichtigen Teile erlangt und als zentrale Schnittstelle zu dem Identity Management System dient. Hierfür wird die Referenzimplementierung SourceID verwendet, welche das Liberty Alliance Protokoll unter anderem realisiert [6]. Dieses webbasierte Single-Sign-On (SSO) und Abrechnungssystem des Identity Providers (IDP) vereint alle Aufgaben der Autorisation und Benutzerverwaltung und stellt daher die Basis für ein rollenbasiertes Zugriffskontrollsystem [7], welches durch den SecurityAspect dem System hinzugefügt wird und alle Benutzerzugriffe kontrolliert.

Die für Anmeldung (IDB) und Zugriffskontrolle (ACDB) benötigte Datenbasis wird durch eine intuitive Methode interaktiv erzeugt. Der Produktionsphase der Software wird eine Trainingsphase vorgeschaltet, in der die Arbeitsabläufe der Benutzer erlernt werden. Ein Arbeitsablauf wird in Form einer Sequenz von Webseiten und dort eingegebenen Werten beschrieben. Die Arbeitsabläufe werden Rollen zugewiesen.

Das Sicherheitssystem ist auch in der Wahl der verwendeten Authentifikationsmethoden flexibel. Beispielhaft wurde die Kombination einer Benutzenamen/Passwort-Kombination mit einer Authentifikation durch Besitz, hier ein USB Token [8], demonstriert. Es können verschiedenste Methoden zur Authentifikation zum Einsatz kommen und mit den Rollen verbunden werden. Hierdurch werden verschiedenste Kombinationen dieser Kriterien erzeugt, da jeder Benutzer unterschiedliche Kombinationen von Rollen besitzen kann. Der Einsatz von Hardwaremerkmalen ermöglicht die Verwendung von proaktiven Verfahren,

die einer Erhöhung des Sicherheitsniveaus auf der Client-Seite dienen, indem in regelmässigen Abständen eine Reauthentifikation verlangt wird.

In Kapitel 1 wird ein Überblick über das verwendete Liberty Alliance Protokoll gegeben. Im darauf folgenden Kapitel 2 wird die Verwendung des Hardwaremerkmals und der proaktiven Komponente beleuchtet. Kapitel 3 beschäftigt sich mit dem Einsatz der aspektorientierten Programmierung und deren Verwendung im zentralen Sicherheitsmodul. Das RBAC-Modell wird in Kapitel 4 vorgestellt und auf die Zusammenhänge von Benutzern, Rollen und Arbeitsabläufen eingegangen. Das resultierende Gesamtkonzept wird in Kapitel 5 vorgestellt und im Kapitel 6 des Implementierung betrachtet. Abschliessende Betrachtungen finden sich in Kapitel 7.

## 1 Das Liberty Protokoll

Single-Sign-On Systeme werden zunehmend interessanter, da sie einen Weg bieten die Verwaltung von verschieden Benutzern für verschiedene Programme zu vereinfachen. Im Internetbereich wurden verschiedene Ansätze in den letzten Jahren entwickelt, von denen das Identity Federation Framework (ID-FF) Protokoll [9] der Liberty Alliance [10], eine breite Vereinigung von Firmen, einer der zukunftsträchtigsten ist. Das Ziel von ID-FF ist das Anbieten einer einheitlichen Plattform für die Verwaltung von Identitäten in vereinigten Systemen. Als Beispiel dient hier eine Fluglinie und ein Autovermieter, welche eine Partnerschaft eingehen und nun durch ID-FF in die Lage versetzt werden, Kunden ein einheitliches Login zu ermöglichen aber auch betriebliche Abläufe zu vereinfachen. Der Mitarbeiter der Fluglinie kann bei Buchung auch direkt ein Auto für den Kunden reservieren.

Das Framework setzt voraus, dass jede Webseite eine eigene Benutzerverwaltung besitzt, der gegenüber der Benutzer seine Merkmale präsentieren muss um sich erfolgreich anzumelden. ID-FF bietet einen Weg, die vertraglichen Beziehungen zwischen Firmen im Rahmen des Logins auf verschiedenen Web-Seiten zu reflektieren. Eine solche Beziehung wird als Circle of Trust bezeichnet (Abbildung 2) und besteht aus einem Identity Provider und verschiedenen Service Providern. Der IDP ist ein zentrales vertrauenswürdiges Portal, an dem sich die Benutzer anmelden und welches die Benutzerdaten verwaltet. Ein SP bietet eine Dienstleistung für Benutzer an, die sich gegenüber dem IDP authentifizieren können. Die Zusammenarbeit zwischen IDPs und SPs ist nicht begrenzt oder statisch, andere IDPs und SPs können mit wenig Aufwand integriert werden.

In dem hier vorgestellten Prototypen wird das ID-FF Protokoll in eine Applikation integriert, ohne den Code der Applikation zu kennen oder zu ändern. Dies wird durch eine komplette Ummantelung der Applikation durch den Security Aspekt und der Integration des SP-Teils des ID-FF Protokolls unter Verwendung des SourceID Frameworks [11]. Der Benutzer meldet sich auf den Seiten des IDP an. Hierdurch wird es möglich einen Dienst über verschiedene Server zu verteilen und so verschiedenen Anforderungen wie Load Balancing gerecht zu werden.

Die AAA-Architektur wird in diesem Prototypen auf den IDP und SP verteilt. Dies wird ermöglicht durch vorher errichtete Vertrauensbeziehung zwischen diesen Parteien und einen sicheren Austausch von Nachrichten, sogenannten Tickets.

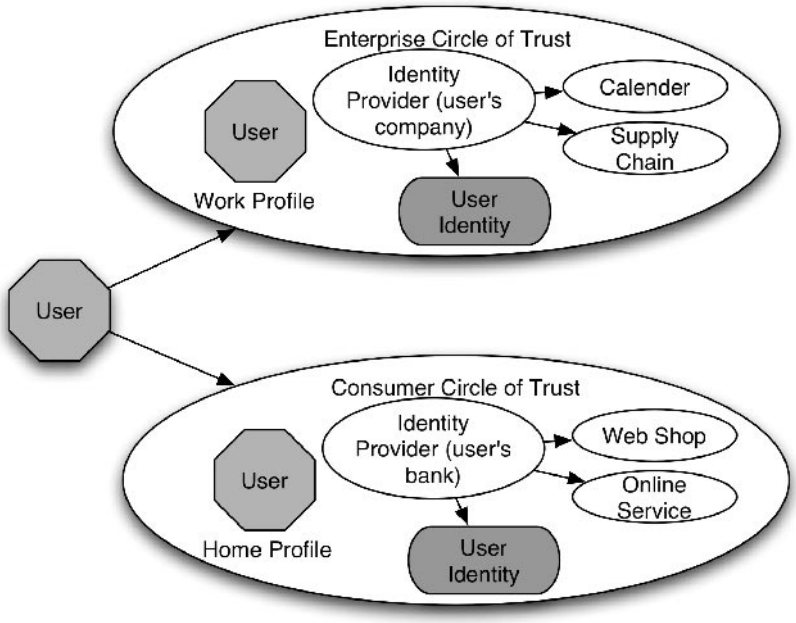


**Abbildung 2:** Circle of Trust

Innerhalb dieses Sicherheitsmodels ist es möglich, spezielle Sicherheitsanforderungen zu modellieren und besondere Authentifikationsmethoden, beispielsweise hardwarebasierte Verfahren, zu integrieren. Hierdurch wird eine besondere Härtung des IDP und dessen Authentifikation erreicht und somit auch eine höhere Qualität der durchgeführten Authentifikation. Auch wird die Administration durch die zentrale Datenbank (IDB) stark vereinfacht.

## 2 Proaktive Massnahmen

Im Rahmen der Anmeldung an ein System muss der Benutzer seine Authentifikationsmerkmale dem System einmal vorweisen. Ein regelmässiges Reauthentifizieren wird oft als zusätzliches Sicherheitskriterium verwendet, um sich der fortwährenden Anwesenheit des Benutzers zu vergewissern. Allerdings ist ein periodisches Abfragen des Passworts für den Benutzer sehr ärgerlich und die Benutzer versuchen, diesen Sicherheitsmassnahmen aus dem Weg zu gehen. Wir verwenden ein Hardwaremerkmal für diese regelmässige Kontrolle. Zum Einsatz kommt der Wibu-Key, der über den USB-Bus an den Rechner des Benutzer angebunden wird.

Dieser Token ersetzt die periodische Passwortabfrage, ist somit eine besitzbasierte Methode. Ein unbeobachtetes Zurücklassen des Token am Arbeitsplatz sollte somit unterbunden werden. Um den Benutzer an die Wichtigkeit der Hardware zu erinnern, wurde von uns ein *proaktives* Applet entwickelt, welches die Authentifikation durchführt und auch den

Benutzer darauf hinweist, dass der Token unbeobachtet zurückgelassen wurde. In einer Erweiterung wäre eine Erfassung dieses Fehlverhaltens des Benutzers denkbar.

In [12] wird Proaktivität als ein System definiert, welches (1) im Auftrag des Benutzers aber auch (2) aus sich selbst heraus handelt.

Das Applet wird auf dem Rechner des Benutzers nach dem Loginprozess gestartet. Es testet das Vorhandensein der Hardware und die Aktivität des Benutzers anhand der Aktivität des Benutzers in der Applikation. Wenn der Benutzer eine längere Zeit inaktiv ist, sendet das Applet eine Nachricht an den IDP und dieser meldet den Benutzer ab. Somit unterstützt das Applet den Benutzer bei der Einhaltung der Sicherheitsrichtlinien. Das Verfahren ist unabhängig von der Authentifikationsmethode und den Kriterien, die die Inaktivität des Benutzers beschreiben, auch lässt sich das Intervall, nachdem der Logout ausgelöst wird, frei definieren. Es werden neben der Unterstützung des Benutzers auch Wege geöffnet, den Benutzer in seinem Verhalten zu überwachen und zu protokollieren. Dies stellt einen Eingriff in die Privatsphäre des Benutzers dar.

## 3 Die aspektorientierte Programmierung

In jedem Softwaresystem existieren *Core Concerns* wie die Bewegungssteuerung oder die Streckenberechnung im Rahmen einer Roboteranwendung. Diese *Concerns* befinden sich in einer Klasse und werden nirgendwo anders benötigt. Andere Concerns werden in verschiedenen Modulen des Systems verwendet wie Logging, Autorisation oder Persistenz und als *Crosscutting Concerns* bezeichnet. Eine Reimplementierung einer Aufgabe in einem anderen Modul wird *Code Scattering* genannt.

Aspektorientierte Programmierung (AOP) stellt einen Lösungsansatz dar, bei dem Crosscutting Concerns in speziellen Modulen, soggenannten Aspekten, isoliert werden. Entsprechend [5] besteht die Entwicklung eines Systems unter Verwendung von AOP aus drei Schritten: (1) Die Aspekte eines Systems werden identifiziert und diese in (2) implementiert. Im Schritt (3) wird dann das System zusammengefügt. In Schritt (1) werden die Crosscutting und Core Concerns identifiziert und separiert und dann in Schritt (2) einzeln Implementiert und getrennt getestet. Abschliessend wird in Schritt (3) unter Verwendung von speziellen Regeln das System mit den Aspekten zusammengefügt, was als *weaving* bezeichnet wird.

Das Weaving ist der zentrale Teil in der aspektorientierten Programmierung. Es werden Regeln für die Definition eines Punktes in der Programmausführung durch AOP eingeführt, welche den Ort und den Zeitpunkt der Ausführung der hinzugefügten Aspekte definieren. Der Ort wird durch den *Pointcut* definiert, welcher ein selektierbarer Punkt (*Join Point*) in einer Programmausführung, wie beispielsweise ein Methodenaufruf, sein muss. Der Zeitpunkt wird durch die Schlüsselworte **before**, **after** und **around** im *Advice* definiert. Dieser stellt den Code dar, der eingefügt werden soll.

Durch das Weaving wird der Java-Bytecode direkt verändert. Es ist also möglich das Programmverhalten durch das Hinzufügen von neuen Eigenschaften im Nachhinein, also nach der Kompilation, zu verändern. Das Resultat ist wieder Java-Bytecode, der keine spezielle Umgebung benötigt um verwendet zu werden. Dies hat zur Konsequenz, dass

alle Aussagen über die Sicherheit der Java-Laufzeitumgebung (JRE) unangetastet bleiben und diese weiterhin die Speicher- und Typensicherheit garantiert und die Verwendung von Ressourcen beschränkt.

Ein Haupterfolg, der durch den Einsatz von AOP erreicht wird, ist eine bessere Verständlichkeit des Sourcecodes [13], der durch die stärkere Modularisierung des Systems begründet ist. Aspektorientierung erleichtert auch eine stärkere Wiederverwendbarkeit der Module. Dadurch müssen Entwickler eine bestimmtes Modul nicht öfters entwickeln und die einzelnen Module können besser getestet werden. Allerdings ist es mitunter schwer, die Korrektheit des Pointcut zu verifizieren. Hier ist weitere Forschung notwendig, um sicherzustellen, dass alle intendierten Punkte, und nur die, in der Programmausführung durch einen Aspekt verändert wurden.

Generell werden Authentifikation und Autorisation als gute Beispiele für den Einsatz von AOP angesehen [5, Kap. 10]. Ein einfacher Ansatz: die Concerns für Authentifikation und Autorisation werden vor jeder Methode ausgeführt und kontrollieren, ob die Ausführung gestattet ist. Dies bremst das System stark aus, so dass der Entwickler genau entscheiden muss, an welchen Stellen eine Überprüfung sinnvoll ist.

AOP eröffnet weitere Möglichkeiten in der Softwareentwicklung durch das *Policy Enforcement*. Es werden Regeln für den Kompilationszeitpunkt definiert, welche Regeln für die Entwicklung der Software formalisieren. Eine einfache Regel besagt, dass der log-Befehl nicht verwendet werden darf.

Diese neue Technik existiert für verschiedene Sprachen wie .net, Java, aber auch C++. Wir haben AspectJ [5] verwendet, da es eins der ausgereiftesten Pakete ist.

## 4 Role Based Access Control

Das zentrale Element eines Sicherheitssystems ist der *Referenzmonitor*, in dem die Sicherheitspolitik definiert ist. Dort werden alle Zugriffe auf *Objekte* (Methoden, Datenbankanfragen, Serviceanfragen) durch *Subjekte* überwacht.

Dieser Ansatz verwendet ein rollenbasiertes Zugriffskontrollsystem (RBAC) um die effektiven Rechte des Benutzers zu modellieren. Ein Benutzer ist Mitglied mindestens einer Rolle und besitzt die kombinierten Rechte aus allen Rollen, in denen er Mitglied ist. Eine Rolle ist eine organisatorische Aufgabe, die sich nur langsam über die Zeit ändert. Wir führen hier das Konzept des *Workflows* ein, der die Schritte, die der Erledigung einer Aufgabe dienen, erfasst. In einem webbasierten System ist ein Schritt der Übergang von einer Webseite zu der nächsten. Somit ist RBAC ein schutzobjektorientiertes System ([14, p. 129] [15]) im Gegensatz zu einer subjektorientierten Beschreibung der Zugriffsrechte.

In Abbildung 3 wird der Zusammenhang der Benutzer, Rollen, Gruppen und Workflows dargestellt. Dieses Modell ermöglicht es, dass die Zugriffsrechte nach dem Prinzip der minimalen Rechte vergeben werden können, wie es durch Schroeder und Saltzer in [16] gefordert wird. Die Idee ist, dass ein Benutzer nur die Dinge sehen oder benutzen darf, die er für die Durchführung seiner Aufgabe benötigt.

R. Sandhu und S. Kandala haben in [17] die Verbindung von RBAC und Workflows formalisiert. Die Zugriffsrechte werden als Recht zur Ausführung eines Workflows interpretiert

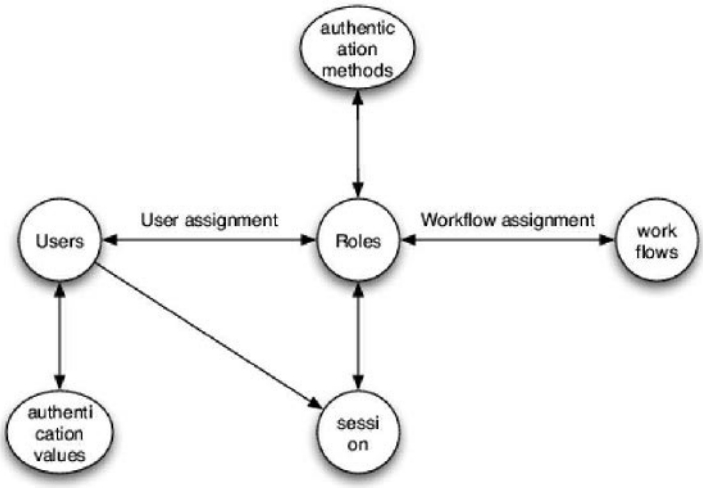


**Abbildung 3:** RBAC Struktur

und als *explizite* Rechte bezeichnet. Dies bedeutet, dass der Benutzer die Zugriffsrechte durch einen endlichen Zustandsautomaten (FSM) erhält, der jeweils den nächsten Schritt berechnet. Hierdurch erhält der Benutzer *implizite* Rechte, die sich im Lauf seiner Arbeit von Seite zu Seite verändern. Er bewegt sich in der Applikation auf einem vordefiniertem Weg.

Die Realisierung der Workflows in einem webbasierten System setzt eine angemessene Reaktion auf das (absichtliche) Verlassen des Weges durch den Benutzer voraus.

Die meisten graphischen Benutzungsoberflächen haben eine zentrale Seite, mit der die Applikation immer den Benutzer empfängt. Als Konsequenz daraus ist diese Seite allen Workflows gemeinsam. Von dieser Seite startend, können zwei Workflows auch weitere Seiten gemein haben. Auch dies muss durch eine Implementierung beachtet werden.

Auch werden Programme benötigt, die den Benutzer bei der Erzeugung des RBAC Modells sowie der Workflows unterstützen.

## 5 Die resultierende Architektur

Unsere Sicherheitsarchitektur umschliesst ein webbasiertes System wie es im Abbildung 4 gezeigt wird. Die einkommenden Daten werden erst durch den *securityAspect* bearbeitet, in dem die Authentifikation und die Prüfung der Autorisation des Benutzers ausgeführt wird. Wenn der Benutzer sich mindestens in einem aktiven Workflow einer zu ihm gehörenden Rolle befindet, werden die einkommenden Daten an die Wirtsapplikation weitergereicht, die dann die resultierenden Webseiten wie gewöhnlich erzeugt. Diese Seiten können im Anschluss durch den *HTMLrewriter* verarbeitet werden, in dem Überprüfungen anhand der erzeugten Webseiten durchgeführt werde können und die Seite verändert werden kann.

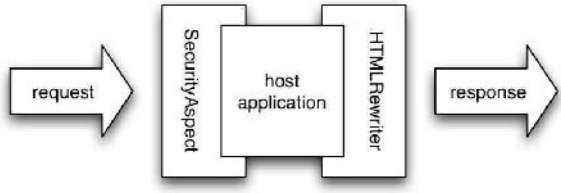


**Abbildung 4:** Umhüllung der Wirtsapplikation

Die ein- und ausgehenden Daten können in einer Webapplikation einfach abgefangen werden, da durch das Vorhandensein von verbindlichen Schnittstellen jede Applikation nach Aussen gleich ist. Jede Antwort der Wirtsapplikation ist ein direktes Ergebnis einer Anfrage an die Applikation. Dies ist ausreichend für eine Autorisationskontrolle und es reicht nur die Schnittstellenmethoden zu kapseln. Es ist auch denkbar jede Methode zu kapseln und so direkte Kontrolle über alle internen Abläufe der Wirtsapplikation zu erlangen. So könnten nachträglich umfangreiche Modultests eingeführt werden, die einen erweiterten Schutz vor Buffer Overflow Angriffen bieten würden oder ungewöhnliche Muster in der Programmausführung erkennen können. Diese Ansätze wurden hier nicht weiter erkundet.

In Abbildung 1 werden die verwendeten Module gezeigt. Die Kommunikation zwischen den Modulen findet in der Vertikalen statt. Die Wirtsapplikation, die in ihrer Funktionalität erweitert wird, ist rechts dargestellt (Database und Host-Application). Der linke Block zeigt die client-seitigen Module und in der Mitte befinden sich die Module, die den IDP bilden.

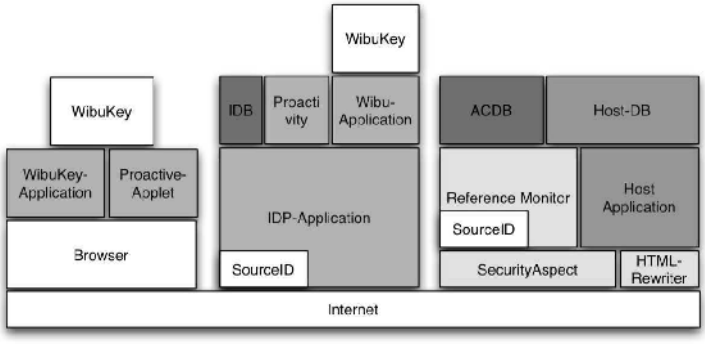


**Abbildung 5:** Aspektorientierte Sicherheitsarchitektur

Software, die um Sicherheitsfunktionalitäten erweitert werden soll, hat oft bereits ein eigenes Benutzermanagement. So ist eine Anmeldung an die Wirtsapplikation eine wichtige Fähigkeit, die der SecurityAspect beherrschen muss. Hierbei muss die Benutzeridentität auf die Identität in der Applikation abgebildet werden. Im aktuellen Ansatz wird das Pro-

blem durch ein Aufrufen der Loginseite der Wirtsapplikation gelöst, der der Benutzername sowie ein Passwort übergeben wird.

Der Referenzmonitor hat kein Wissen über die Wirtsapplikation, alle Entscheidungen werden auf der Basis der ankommenden Daten und des Zustands der Workflows getroffen. Eine Regel, die besagt, dass nur Waren die am Lager sind auch verkauft werden dürfen, kann nicht formuliert werden. Diesem Problem wird durch einen Zugriff auf die Applikationsdatenbank begegnet (siehe weiter unten).

Durch das Einführen von verbundenen Identitäten (federated identities) wird ein IDP, wie er in der Mitte von Abbildung 1 zu sehen ist, notwendig. Um die Identitäten im Kontext des Internets zu verwenden werden (1) gute Authentifikationsverfahren und (2) ein Bewusstsein für den Wert der Identität beim Benutzer benötigt. Teil (1) kann durch die Einführung von Hardwaremerkmalen begegnet werden, (2) lässt verschiedene Wege offen. Wir haben uns für eine Erziehung des Benutzers entschieden, wie es in Kapitel 2 beschrieben wird.

Aus der Trennung von IDP und SP resultiert die Notwendigkeit von zwei unabhängigen Datenbanken. In der IDB (siehe Abbildung 1) werden alle Informationen zur sicheren Authentifikation verwahrt, also beispielsweise die Passwörter, in der ACDB befinden sich die Workflowdefinitionen, das RBAC-Modell und die Authentifikationsmethoden, die für einen Benutzer abhängig von seinen Rollenzugehörigkeiten gefordert werden müssen (siehe Abbildung 3). Darüber hinaus speichert die ACDB den Status jedes aktiven Workflows jeder Session, was es ermöglicht, dass ein Benutzer mehr als eine aktive Session besitzen kann. Durch Veränderungen am Sicherheitsmodell können hierdurch auch exklusive Zustände oder Workflows definiert werden, was den Eigenschaften eines Chinese-Wall Modells [18] entspricht.

Wie oben erwähnt benötigt der Administrator Werkzeuge die ihn bei der Konfiguration des SecurityAspects unterstützen. Die Werkzeuge müssen also bei der Erzeugung der Rollen, Benutzer, Workflows und den Beziehungen zwischen diesen dem Menschen zur Seite stehen. Benutzer und Rollen können als eine herkömmliche Aufgabe angesehen werden, Workflows hingegen sind wesentlich schwerer, da der Administrator keinen Einblick in den Sourcecode besitzt und dies auch nicht gewünscht ist. Daher wurde von uns eine Trainingsphase als Ansatz gewählt, in der dem System beispielhaft die Workflows beigebracht werden. Hierzu wird das Wirtssystem mit einem speziellen Trainingsaspekt verwebt, der die Benutzerinteraktion aufzeichnet und durch eine GUI fernsteuerbar ist. In der Aufzeichnung ist jeder Parameter mit den vom Trainer eingegebenen Werten zu sehen. Der Trainer muss diese durch Regeln ersetzten. Als Regelbeschreibung haben wir uns für reguläre Ausdrücke entschieden [19]. In dem angesprochenen erweiterten Modell ist es zusätzlich möglich SQL-Terme einzugeben, in deren Ergebnismenge der Parameterwert enthalten sein muss (siehe Abbildung 7). Jede Antwort eines Online Transaction Systems (OLTP) basiert direkt auf einem Zustand einer Datenbank, daher sind wir überzeugt, dass diese Herangehensweise die meisten Probleme lösen sollte, die aus der Notwendigkeit entstehen, dass der interne Zustand der Wirtsapplikation für die Erzeugung von Regeln benötigt wird.

In der Produktionsphase wird das Wirtssystem mit dem SecurityAspect verwebt, in dem die AAA-Funktionalität konzentriert ist. Dieser überprüft bei jedem Request zuerst ob eine initiale Authentifikation durchgeführt wurde und danach, ob eine Reauthentifikation

notwenig ist. Falls dem so ist, wird über das Liberty-Framework eine angefragt, die durch den IDP durchgeführt wird. Danach wird der Referenzmonitor mit allen übergebenen Parametern aufgerufen. Der Monitor entscheidet auf der Basis der Parameter und den aktiven Workflows des Benutzers, ob der Zugriff zugelassen werden darf. Der Monitor bestimmt die aktuellen Zustände und alle Folgezustände und prüft für jeden, ob dieser erreicht werden kann. Workflows, in denen es keine erreichbaren Folgezustände gibt, werden als nicht-aktiv markiert. Ein Zugriff wird erlaubt, falls ein Zustand erreicht werden kann. Falls keine aktiven Workflows übrig bleiben, wird eine Fehlerseite angezeigt, in der der Benutzer die Möglichkeit erhält zu der Eingabemaske zurückzukehren oder auf eine Seite zu gelangen, die in wenigstens einem Workflow eine initiale Seite darstellt. Wie oben ausgeführt existiert in einem normalen OLTP zumeist eine zentrale Seite, die die bevorzugte Seite für einen Abbruch ist. Nachdem das Wirtssystem die Eingabe verarbeitet hat, kann die resultierende HTML-Seite, die an den Benutzer übertragen werden soll, zunächst durch den HTMLrewriter bearbeitet werden, der in unserem Fall das proactiv-Applet hinzufügt.

Als Nebeneffekt dieser Methode stellt sich die Implementierung eines Audit sehr einfach dar. Jede Aktion eines Benutzers kann problemlos protokolliert werden, so dass es theoretisch möglich ist, häufige Fehler in der Softwarebenutzung zu erkennen aber auch zu registrieren, wie hoch die Latenzzeiten eines Mitarbeiters sind. Dies stellt einen starken Eingriff in die Privatsphäre des Mitarbeiters dar.

## 6 Einsatzbeispiel

In diesem Abschnitt wird ein kurzer Überblick über den Einsatz der prototypischen Implementierung gegeben. Der Einsatz ist aufgeteilt in die Trainings- und Produktionsphase.

Abbildung 6 zeigt ein Bildschirmphoto der Workflow-Aufzeichnung durch unser Werkzeug. Die Aufzeichnung wurde bereits gestartet und in der gerade laufenden Applikation kann der Administrator Workflows erzeugen einfach indem er die Applikation benutzt.


**Abbildung 6:** workflow recording

Nachdem der Workflow gespeichert ist, kann dieser bearbeitet werden (Abb. 7). Hier werden die regulären Ausdrücke und die SQL-Abfragen angegeben, die später in der Produktionsphase den Zugriff der Benutzer regulieren.

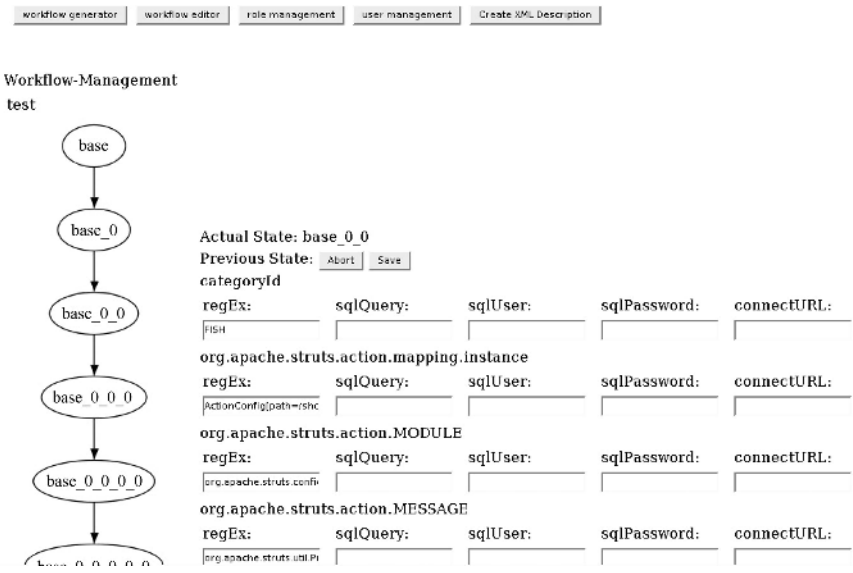


**Abbildung 7:** workflow editor

Nach der Aufzeichnung aller notwenigen Workflows werden die Rollen und Benutzer angelegt wie es in Abbildung 8 gezeigt wird. Jeder Benutzer muss mindestens Mitglied einer Rolle und einem IDP zugeordnet sein. Für die Authentifikation beim IDP bekommt er ein Passwort zugewiesen und, falls er den WibuKey verwendet, einen spezifischen FirmCode und UserCode. Falls eine Anmeldung an dem Wirtssystem notwendig ist, wird noch der lokale Benutzername und das zugehörige Passwort angegeben.

Zu Beginn der Produktionsphase wird die Wirtsapplikation mit dem SecurityAspect und dem HTMLrewriter verwebt. Der Letztere verändert den Output der Applikation wie es in Abbildung 9 zu sehen ist. Er fügt verschiedene Debug-Meldungen und ein Applet hinzu. Der grüne Punkt bedeutet, dass der WibuKey am Rechner angeschlossen ist und der Benutzer in der letzten Zeit gearbeitet hat. Falls eine Zeit lang nicht gearbeitet wird, verändert sich die Anzeige zu einem roten Punkt.

## 7 Abschluss

Wir haben gezeigt, dass es möglich ist SSO-Fähigkeiten einer Software nachträglich hinzuzufügen, ohne dass ein Wissen über den Sourcecode der Applikation von Nöten ist. Auch haben wir zeigen können, dass eine workflowbasierte Autorisation hinzugefügt und an das System intuitiv angepasst werden kann.

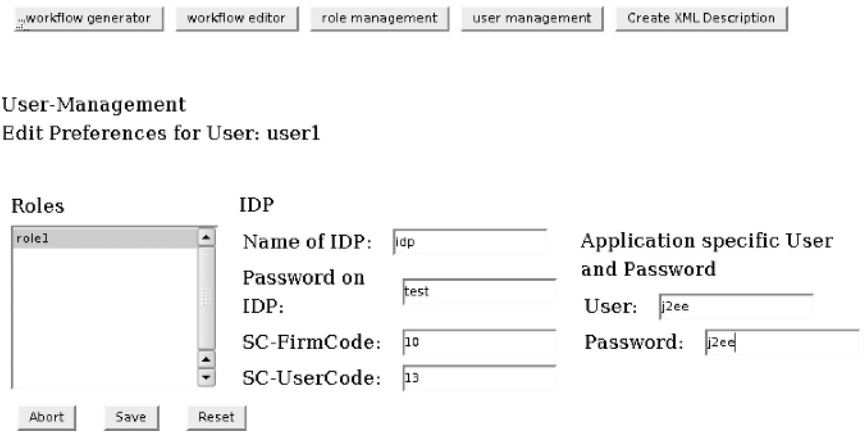


**Abbildung 8:** user editor

Hierdurch ist deutlich geworden, das aspektorientierte Programmierung ein gutes Mittel ist für die Implementierung von Sicherheitsfunktionen für bestehende und neue Systeme. Es eröffnet neue Wege in der Modularisierung und Codewiederverwendung.

SSO eröffnet neue Möglichkeiten in der Firmeninteraktion. Auf der Basis des Liberty-Frameworks ist es möglich Vertrauensbeziehungen mit geringem Aufwand zu erstellen.

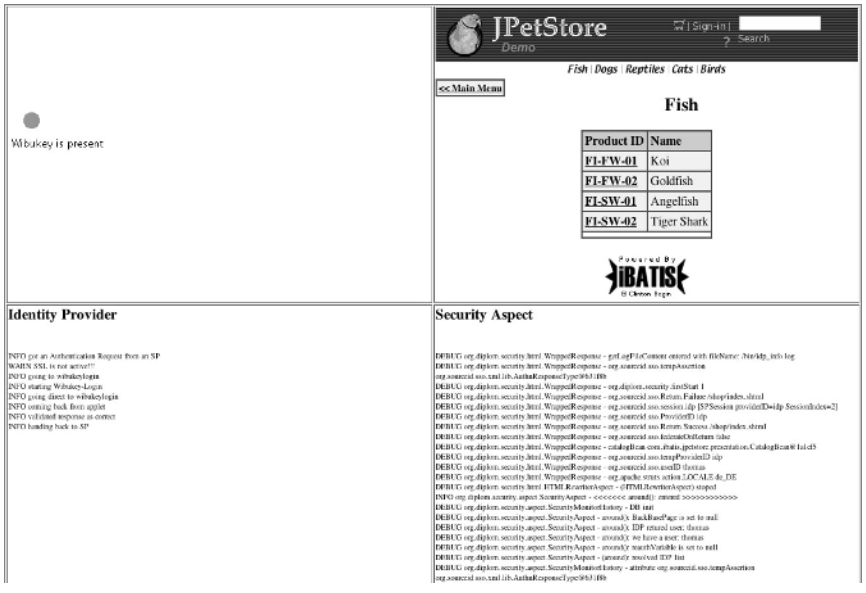


**Abbildung 9:** productive phase

In Kombination mit den Workflows und dem Trainingssystem können auch kleine und mittlere Unternehmen interagieren und webbasierte Abläufe etablieren.

Aus der Sicherheitsperspektive ist der Schwachpunkt beim IDP, da dieser anfällig für einen Denial-Of-Service Angriff ist, so dass alle Applikationen, die auf den IDP aufbauen, nicht verwendbar sind für die Benutzer. Daraus folgt, dass der IDP speziell gegen Angriffe geschützt werden muss und auch ein Fall-Back Plan benötigt wird. Ein zweiter Angriffspunkt ist die ACDB, da hier alle Merkmale gespeichert sind, die für eine erfolgreiche Erkennung eines Benutzers benötigt werden.

Aus der hohen Modularisierung des Sicherheitsmoduls ergibt sich auch ein weiteres Problem in der Einbeziehung von Informationen aus der Wirtsapplikation in die Entscheidungsfindung des Referenzmonitors. In unserem Modell wurde dies durch die Integration von SQL-Abfragen in die Entscheidungsfindung gelöst, allerdings fehlt hier eine ausführliche Betrachtung von Alternativen.

Auf der anderen Seite bedeutet eine hohe Modularisierung und Unabhängigkeit des Sicherheitsmoduls, dass es möglich ist ein „Plug-In“ zu entwickeln, welches stetig weiterentwickelt werden kann und somit ein eigenständiges Produkt darstellt. Dieses kann im Fall von Weiterentwicklungen oder Fehlerkorrekturen einfach und schnell, sogar beim Kunden vor Ort, ausgetauscht werden.

## Literatur

[1] Abdulwahed Mo. Khalfan. Information security considerations in IS/IT outsourcing projects: a descriptive case study of two sectors. International Journal of Information Management 24 (2004) 29-42.

[2] Michael Mehrhoff. Outsourcing unter Sicherheitsaspekten. In *D-A-CH Security 2004*, 62-70. P. Horster (Ed.), 2004.

[3] Jeff Prosise. DMTF Web-Based Enterprise Management Initiative. http://www.dmtf.org/standards/wbem.

[4] Michael Herfert, Andreas U. Schmidt, Peter Ochsenschläger, Jürgen Repp, Roland Rieke, Martin Schmucker, Steven Vettermann, Uwe Böttge, Cristina Escaleira und Dirk Rüdiger. Implementierung von Security Policies in offenen Telekollaborationen. In *D-A-CH Security 2004*, 37-49. P. Horster (Ed.), 2004.

[5] Ramnivas Laddad. *AspectJ in Action: Practical Aspect-Oriented Programming*. Manning Publications Co., 2003.

[6] Liberty Alliance Project. Liberty Alliance Developer Tutorial. http://www.projectliberty.org/resources/tutorial_draft.pdf.

[7] D. F. Ferraiolo und D. R. Kuhn. Role Based Access Control. *15th National Computer Security Conference*, 1992. http://csrc.nist.gov/rbac/Role_Based_Access_Control-1992.html.

[8] Wibukey Systems AG. WIBU-KEY. http://wibu.de/de/wibukey.php.

[9] Liberty Alliance Project. Liberty ID-FF Architecture Overview Version 1.2 http://www.projectliberty.org/specs/liberty-idff-arch-overview-v1.2.pdf.

[10] Liberty Alliance http://project-liberty.org.

[11] SourceID – Open Source Federated Identity Management http://www.sourceid.org.

[12] Antti Salovaara and Antti Oulasvirta. Six modes of proactive resource management: a user-centric typology for proactive behaviors. Proceedings of the third Nordic conference on Human-computer interaction pages 57-60. ACM Press, 2004. http://doi.acm.org/10.1145/1028014.1028022.

[13] Bart De Win, Wouter Joosen und Frank Piessens. AOSD & Security: a practical assessment. February 28, 2003.

[14] C. Eckert. IT-Sicherheit. Oldenbourg Wissenschaftsverlag GmbH 2001

[15] R. Sandhu. Role-based access control. In Advances in Computers, Jgg. 46, Seiten 237-286. M. Zelkowitz Eds. Academic, 1998.

[16] Jerome H. Saltzer und Michael D. Schroeder The Protection of Information in Computer Systems. 1975. http://web.mit.edu/Saltzer/www/publications/protection/

[17] Savith Kandala und Ravi Sandhu. Secure role-based workflow models. In Proceedings of the fifteenth annual working conference on Database and application security, pages 45-58. Kluwer Academic Publishers, 2002.

[18] D. F. C. Brewer und M. J. Nash The Chinese Wall Security Policy In: IEEE Symposium on Security and Privacy, pages 206-214, 1989

[19] Jeffrey E. F. Friedl. Mastering Regular Expressions. O'REILLY, 1997.