ABAC – Ein Referenzmodell für attributbasierte Zugriffskontrolle

Torsten Priebe, Wolfgang Dobmeier, Björn Muschall, Günther Pernul

Lehrstuhl für Wirtschaftsinformatik I,
Universität Regensburg, D-93040 Regensburg
{torsten.priebe,wolfgang.dobmeier,bjoern.muschall,guenther.pernul}
@wiwi.uni-regensburg.de

Abstract: Moderne Anwendungen aus dem Bereich des e-Commerce, sowie Enterprise- und e-Government-Portale bringen aufgrund der Vielzahl höchst heterogener Benutzer und der Diversität der Informationsressourcen die Notwendigkeit für flexible Autorisierungs- und Zugriffskontrollverfahren mit sich. Für den Zugriff auf derartige Anwendungen ist sicherzustellen, dass Benutzer die notwendigen Berechtigungen besitzen. Die Identität der Benutzer ist dabei von sekundärer Bedeutung. Vor diesem Hintergrund haben sich einige Zugriffskontrollansätze auf Basis von Benutzerattributen (sog. Credentials) und Metadaten entwickelt. Dieser Beitrag stellt – auf diese vorhandenen Ansätze aufbauend – ein Referenzmodell unter dem Namen ABAC („Attribute-Based Access Control“) vor. Weiterhin wird anhand mehrerer Beispiele gezeigt, dass ABAC als Generalisierung auch traditioneller Zugriffskontrollmodelle gesehen werden kann und eine Abbildung dieser Modelle auf ABAC möglich ist. Der Entwurf von ABAC-Sicherheitspolitiken wird diskutiert und die Implementierung im Rahmen eines Forschungsprototypen präsentiert.

1 Einleitung

Das Internet bringt die Notwendigkeit für flexible Autorisierungs- und Zugriffskontrollansätze mit sich. Moderne Anwendungen, beispielsweise aus dem Bereich des e-Commerce, sowie Enterprise- und e-Government-Portale führen zu einer großen Vielfalt an Benutzern, die auf unterschiedliche Weise mit den IT-Systemen interagieren. Insbesondere kann nicht mehr gewährleistet werden, dass sie im Voraus bekannt sind. Als Beispiel dafür kann bei einem öffentlichen Portal, das zugriffsbeschränkte Ausschreibungen für prospektive Lieferanten zur Verfügung stellt, nicht jeder Interessent im Voraus benannt werden. Zudem bedeutet das, dass die Menge an Benutzern dynamisch und heterogen ist. Ähnliches trifft für die verwalteten Informationsressourcen zu; auch ihre Anzahl kann sehr groß sein, wobei nicht jedem Benutzer alle Dokumente zugänglich sein sollen.

Aus diesen Rahmenbedingungen ergibt sich, dass die Authentifikation in derartigen Umgebungen in den Hintergrund tritt und Autorisierung und Zugriffskontrolle an Bedeutung gewinnen. Die Verwendung traditioneller Zugriffskontrollverfahren wie RBAC (rollen-

basierte Zugriffskontrolle) wird durch die möglicherweise nicht bekannte Identität der Benutzer und die große Anzahl sowohl von Subjekten als auch Objekten erschwert. Tatsächlich ist es bei vielen der genannten Anwendungen auch gar nicht notwendig, die Identität der Benutzer zu kennen. Viel wichtiger ist es, sicherzustellen, dass sie gewisse Berechtigungen besitzen. Vor diesem Hintergrund haben sich einige Zugriffskontrollansätze auf Basis von Benutzerattributen (sog. Credentials) und Metadaten entwickelt [AABF02, OAS03, PS04]. Ein erstes vereinheitlichtes Referenzmodell wurde in [PFMP04] als Sicherheitsmuster vorgestellt. Dieser Beitrag baut darauf auf und stellt ein erweitertes Modell unter dem Namen ABAC („Attribute-Based Access Control“) vor. Ziel dieses Beitrages ist es weiterhin darzustellen, dass ABAC als Generalisierung auch traditioneller Zugriffskontrollmodelle gesehen werden kann und diese auf ABAC abgebildet werden können.

Der Beitrag ist folgendermaßen aufgebaut: Abschnitt 2 dokumentiert das ABAC-Referenzmodell, wobei ein Grund- und ein erweitertes Modell unterschieden werden. Abschnitt 3 schlägt eine grafische Notation zum Entwurf von ABAC-Politiken mit UML vor. Darauf aufbauend zeigt Abschnitt 4, wie traditionelle Zugriffskontrollmodelle (konkret DAC, MAC und RBAC) im ABAC-Modell abgebildet werden können. In Abschnitt 5 wird eine Implementierung des ABAC-Modells vorgestellt; eine Zusammenfassung des Beitrags sowie ein Ausblick auf zukünftige Entwicklungen finden sich in Abschnitt 6.

2 Das ABAC-Referenzmodell

Die Grundidee attributbasierter Zugriffskontrolle besteht darin, Zugriffsrechte zwischen den Subjekten und Objekten nicht statisch zu definieren, sondern ihre Eigenschaften oder Attribute dynamisch als Grundlage der Autorisierung zu nutzen. Die Attribute der Benutzer werden in diesem Zusammenhang auch Credentials genannt. Dabei kann es sich um allgemeine Eigenschaften wie beispielsweise die Position des Benutzers im Unternehmen handeln. Sofern erforderlich, insbesondere bei Zugriff durch Unternehmensexterne (z.B. Kunden), treten aber auch Attribute wie das Alter, die Lieferadresse, oder sogar erworbene Credentials (z.B. Abonnements) an diese Stelle. Attribute können (im Gegensatz zu relativ statisch definierten Rollen) sehr dynamisch sein. Beispielsweise könnte man sich in einem mobilen Umfeld die Verwendung des aktuellen Standorts eines Benutzers als Attribut vorstellen. Zur Kodierung der Benutzerattribute kann beispielsweise X.500 [ITU96] verwendet werden. Auf der Seite der Sicherheitsobjekte lassen sich die Inhalte der Dokumente durch Metadaten beschreiben, beispielsweise basierend auf dem Dublin Core Metadatenstandard[1]. Auch diese Metadaten können als Attribute zur Zugriffskontrolle herangezogen werden.

Wie in der Einleitung dieses Beitrages erwähnt, wurden bereits einige attributbasierte Ansätze in der Literatur vorgeschlagen. So schlagen Adam et al. [AABF02] das Digital Library Access Control Model (DLAM) für Sicherheit in digitalen Bibliotheken vor, welches Zugriffsrechte anhand von Benutzerattributen und mit Objekten assoziierten Konzepten definiert. Andere Arbeiten haben ihre Ursprünge im Bereich der Public Key und Privilege

[1] `http://www.dublincore.org`

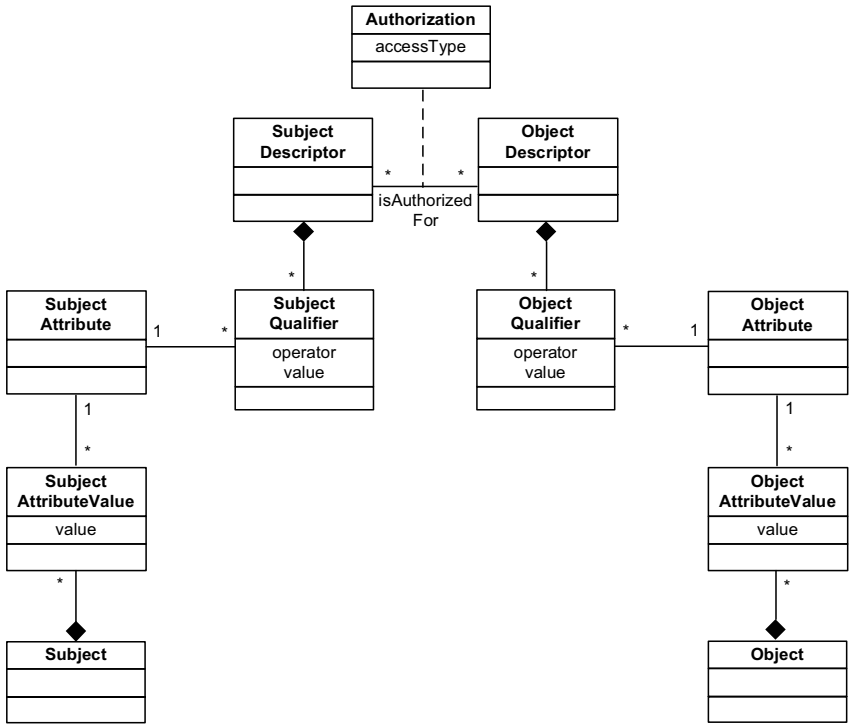

Figure 1: ABAC-Grundmodell

Management Infrastructures und basieren auf X.509-Attributzertifikaten [ITU00]. Erste Vorschläge zur Verwendung von Attributzertifikaten zur Zugriffskontrolle findet sich in [OPS00, Bis02]. Mittlerweile haben sich einige Projekte und Systeme in diesem Umfeld entwickelt (z.B. PERMIS[2] und Shibboleth[3]). Auch der XML-Dialekt zur Definition von Zugriffskontrollpolitiken XACML [OAS03] basiert auf Benutzer- und Objektattributen. Ein neueres Beispiel für die Anwendung eines attributbasierten Modells im Umfeld des Digital Rights Management ist UCON_{ABC} [PS04].

2.1 Grundmodell

Wie erwähnt wurde in [PFMP04] ein erstes Referenzmodell für attributbasierte Zugriffskontrolle (dort noch unter dem Namen MBAC – „Metadata-Based Access Control“) als Sicherheitsmuster vorgestellt. Dieses ist in Abbildung 1 mit leicht angepasster Terminologie als UML-Klassendiagramm dargestellt.

[2] `http://www.permis.org`
[3] `http://shibboleth.internet2.edu`

Subjekte und Objekte (*Subject* und *Object*) sind durch eine Menge von Attributwerten (*SubjectAttributeValue* und *ObjectAttributeValue*) repräsentiert. Die Autorisierungen (*Authorization*) werden zwischen Subjekt- und Objektdeskriptoren (*SubjectDescriptor* und *ObjectDescriptor*) definiert. Diese bestehen aus einer Menge von Attributbedingungen (*SubjectQualifier* und *ObjectQualifier*), z.B. „Alter > 18“, „PLZ beginnt mit 93“. Ein Subjektdescriptor kann demnach mehreren realen Subjekten entsprechen. Das gleiche gilt für die Objektdeskriptoren, die Attributbedingungen auf Basis von Objekteigenschaften, wie „Verbindung zu einem bestimmten Projekt“ oder „Veröffentlichung durch einen bestimmten Verlag“, verwenden. Subjekt- und Objektdeskriptoren stellen so etwas wie Subjekt- und Objektgruppen dar, nur dass die Zuordnung zu diesen Gruppen nicht explizit, sondern implizit durch die Attributwerte erfolgt. Bei dem Klassendiagramm in Abbildung 1 ist zu beachten, dass die Klassen *SubjectAttribute* und *ObjectAttribute* die Schemaebene darstellen (eine Instanz wäre z.B. „Alter“ oder „Verlag“). Die Attributausprägungen sind durch die Klassen *SubjectAttributeValue* und *ObjectAttributeValue* repräsentiert.

Ein Beispiel für eine Sicherheitspolitik nach dem ABAC-Grundmodell findet sich in Abbildung 4 a) und wird in Abschnitt 3 näher erläutert.

Das ABAC-Grundmodell kann als eine Art kleinster gemeinsamer Nenner der in der Literatur betrachteten attributbasierten Zugriffskontrollmodelle gesehen werden. Die dargestellten Konzepte finden sich, zum Teil unter anderem Namen, in allen genannten Modellen (DLAM, XACML, UCON_{ABC}) wieder.

2.2 Erweitertes Modell mit Sitzungen und Bedingungen

Aufbauend auf dem ABAC-Grundmodell sollen zwei Erweiterungen dargestellt werden, die so oder ähnlich auch in den bestehenden attributbasierten Zugriffskontrollansätzen [AABF02, OAS03, PS04] existieren: Sitzungen (*Session*) und Bedingungen (*Condition*).

Der Benutzer hat die Möglichkeit, innerhalb einer Sitzung nur eine Teilmenge der ihm zugewiesenen Attribute zu offenbaren. Nur die ausgewählten (aktivierten) Attributwerte werden zur Zugriffskontrolle verwendet. Da eine minimale Menge an Attributwerten auch eine minimale Menge an Autorisierungen mit sich bringt, wird so das Prinzip der „Least Privileges“ unterstützt, welches besagt, dass Benutzer nur die Berechtigungen besitzen sollten, die sie unbedingt für ihre aktuellen Aufgaben benötigen. Hinzu kommt, dass das Sitzungskonzept aus Datenschutzüberlegungen zur Erhöhung der informationellen Selbstbestimmung und zur Datensparsamkeit beitragen kann. Ein Benutzer muss nur die persönlichen Daten (in Form von Attributwerten) preisgeben, die zur Durchführung der beabsichtigten Tätigkeit (bzw. der Autorisierung dazu) notwendig sind[4]. Ein Besteller bei einem e-Commerce-Anbieter für Videofilme muss sein Alter beispielsweise nur offenbaren, wenn seine Bestellung FSK-geschützte Positionen beinhaltet.

Wir lehnen uns mit dem ABAC-Sitzungskonzept an den aus der rollenbasierten Zugriffskontrolle (RBAC) [FSG+01] bekannten Sitzungen an. XACML [OAS03] und UCON_{ABC}

[4] Siehe hierzu auch die kontrollierte Preisgabe von persönlichen Eigenschaften im Internet bei P3P (Platform for Privacy Preferences) [W3C02].

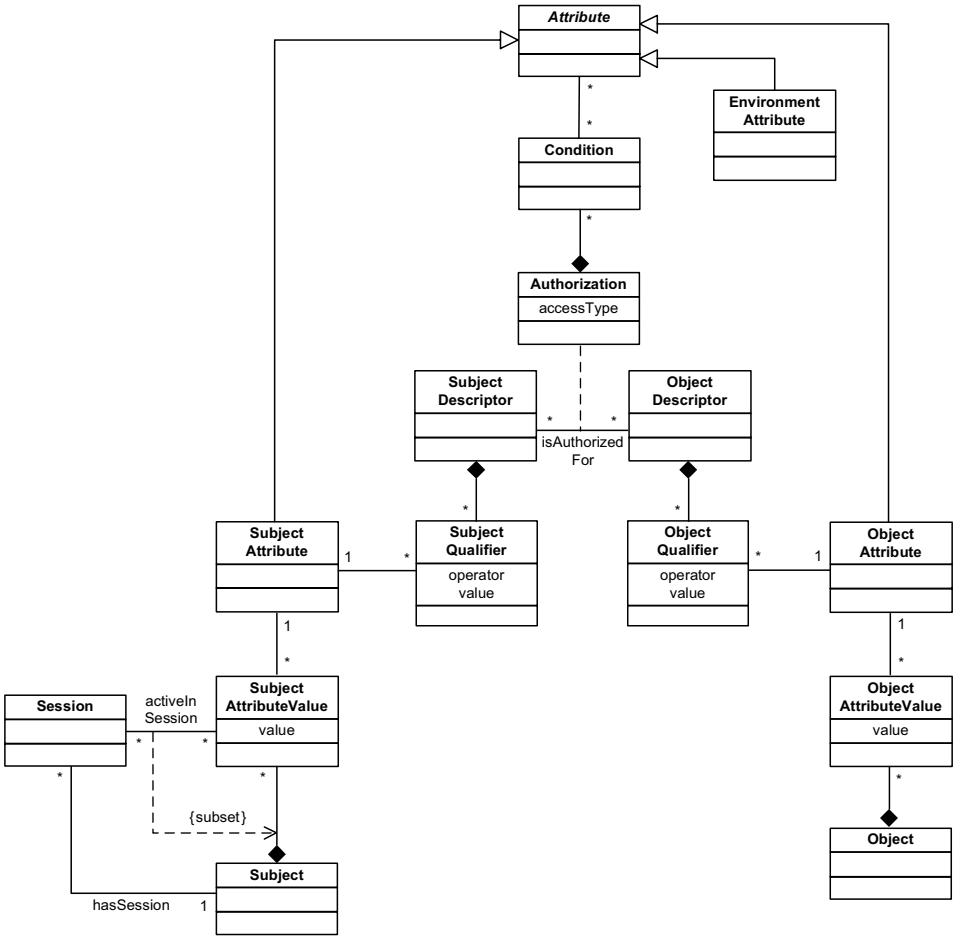

Figure 2: Erweitertes ABAC-Modell

[PS04] gehen von sitzungsloser Kommunikation zwischen Benutzer und System aus. Jedoch ist bei beiden Varianten vorgesehen, dass die zu verwendenden Attributwerte bei jedem Request erneut übermittelt werden. Insofern gilt – aus Sicht des Zugriffskontrollmodells – dass bei jedem Request eine Teilmenge der dem Subjekt zugewiesenen Attributwerte ausgewählt werden kann. Da eine Sitzung ein oder mehrere Requests beinhaltet, lässt sich dieses Vorgehen auch durch das ABAC-Sitzungskonzept abbilden.

Neben Sitzungen beinhaltet das erweiterte ABAC-Modell die Möglichkeit, Autorisierungen mit zusätzlichen Bedingungen (*Condition*) zu versehen. Die Grundidee findet sich bereits im klassischen benutzerbestimmbaren Zugriffskontrollmodell (DAC) [Eck04, S. 242], dort werden Bedingungen jedoch Prädikate genannt. Das Konzept der Bedingungen existiert auch in XACML [OAS03] und UCON_{ABC} [PS04]. Während mit dem ABAC-Grundmodell in den Subjekt- und Objektqualifiern Attribute nur mit konstanten Werten verglichen werden konnten, ist es möglich, in einer Bedingung ein Subjektattribut mit einem Objektattribut zu vergleichen. Beispiele für eine solche Politik finden sich in Abbildung 4 b) und c). Neben Subjekt- und Objektattributen führt das erweiterte ABAC-Modell (wie auch in XACML und UCON_{ABC} vorhanden) Umgebungsattribute (*EnvironmentAttribut*e), wie beispielsweise die Uhrzeit ein. Diese Attribute können ebenfalls in Bedingungen verwendet werden, wodurch sich eine Autorisierung z.B. auf die üblichen Geschäftszeiten beschränken lässt. Das erweiterte ABAC-Modell ist in Abbildung 2 als UML-Klassendiagramm dargestellt.

3 Entwurf von ABAC-Politiken mit UML

Durch die höhere Flexibilität (und Komplexität) attributbasierter Zugriffskontrollmodelle ist die Textform zur Dokumentation der Sicherheitspolitiken nur eingeschränkt geeignet. Auch die XML-basierte Notation von XACML [OAS03] ist für den menschlichen Leser schwer zu erfassen. Daher schlagen wir eine grafische, auf UML basierende Notation vor.

Konkret werden, wie in Tabelle 1 aufgeführt, Subjekt- und Objektdeskriptoren als Klassen in einem UML-Klassendiagamm dargestellt, versehen mit einem Stereotypen (textuell oder als Symbol). Die Subjekt- und Objektattribute der Subjekt- und Objektqualifier werden dabei als zugehörige Klassenattribute notiert, sofern vorhanden mit Operator und Wert als UML Constraint. Eine Autorisierung wird als gerichtete Assoziation zwischen Subjekt- und Objektdeskriptorklasse gezeichnet, benannt mit der erlaubten Zugriffsart. Einer Autorisierung kann eine Bedingung in Form eines UML Constraints beigefügt werden. Diese hier aus Platzgründen nur grob dargestellte Vorgehensweise könnte auch in Form eines UML-Profils spezifiziert werden. Abbildung 3 zeigt die vorgeschlagene UML-Notation für ABAC-Politiken exemplarisch.

Abbildung 4 zeigt nun drei beispielhafte ABAC-Politiken in der dargestellten UML-Notation. Bei den Attributen lehnen wir uns, wie bereits in Abschnitt 2 erwähnt, an X.500 und Dublin Core an. Das erste Beispiel unter a) geht von einem e-Government-Kontext aus. Auf ein bestimmtes Bauvorhaben in der Hemauerstraße in Regensburg betreffende Dokumente sollen nur volljährige Anrainer zugreifen können. Hierzu wird ein Subjekt-

ABAC-Element	UML-Element	Stereotyp	Symbol
SubjectDescriptor	Class	<<SubjectDescriptor>>	
SubjectAttribute	Attribute	–	–
SubjectQualifier	Constraint	–	–
ObjectDescriptor	Class	<<ObjectDescriptor>>	
ObjectAttribute	Attribute	–	–
ObjectQualifier	Constraint	–	–
Authorization	Association	–	–
Condition	Constraint	–	–

Table 1: UML-Elemente und Stereotypen zum Entwurf von ABAC-Politiken

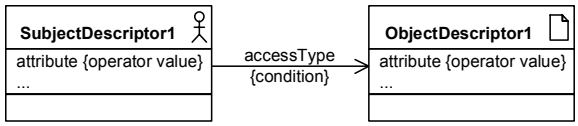

Figure 3: UML-Notation für ABAC-Politiken

deskriptor „NeighborHemauer“ definiert, der als Straße die Hemauerstraße und als Ort, repräsentiert durch das X.500-Attribut „Locality“ (l), Regensburg in Deutschland fordert. Weiterhin wird durch einen Subjektqualifier festgelegt, dass das Alter größer oder gleich 18 sein muss. Auf Objektseite wird ein Objektdeskriptor „ProjectHemauer“ definiert, der alle Objekte repräsentiert, die über das Dublin-Core-Element „Coverage“ mit dem Vorhaben „ProjectHemauer“ verknüpft sind. Ein 29-jähriges Subjekt „Priebe“, welches in der Hemauerstraße in Regensburg wohnt, wird anhand seiner Attributwerte automatisch als „NeighborHemauer“ eingestuft und erhält so Zugriff auf die betreffenden Dokumente. Eine manuelle Rollenzuweisung ist nicht notwendig.

Während das obige Beispiel ausschließlich Konstrukte des ABAC-Grundmodells verwendet, verwenden die Beispiele in Abbildung 4 b) und c) im erweiterten ABAC-Modell definierte Bedingungen. Das Beispiel in b) geht von einem e-Commerce-System zum Verleih von Videofilmen aus. Ein Kunde besitzt die Rolle „Customer“ und ein Alter; er darf Filme nur dann ausleihen, wenn sein Alter größer oder gleich der FSK-Beschränkung des Filmes ist. Diese Einschränkung wird als Bedingung definiert. Ohne diese Möglichkeit zum Vergleich von Subjekt- mit Objektattributen hätten Subjekt- und Objektdeskriptoren für jede mögliche FSK-Klasse definiert werden müssen.

Beispiel c) zeigt schließlich, wie ein Grundproblem der rollenbasierten Zugriffskontrolle durch ABAC-Bedingungen gelöst werden kann, nämlich die Anforderung dass der Ersteller eines Objektes besondere Rechte genießt. Hierzu müsste im rollenbasierten Fall für jeden Ersteller eine eigene Rolle definiert werden, was das Rollenkonzept ad absurdum führen würde. Die dargestellte Politik stellt einen Ausschnitt aus dem Szenario einer Job-Vermittlung dar. Bewerbungen können von jedem Bewerber erstellt werden, jedoch kann ein Bewerber nur seine eigenen Bewerbungen lesen und ändern. Hierzu wird das X.500-Attribut „Distinguished Name“ (dn) mit dem Dublin-Core-Element „Creator“ verglichen.

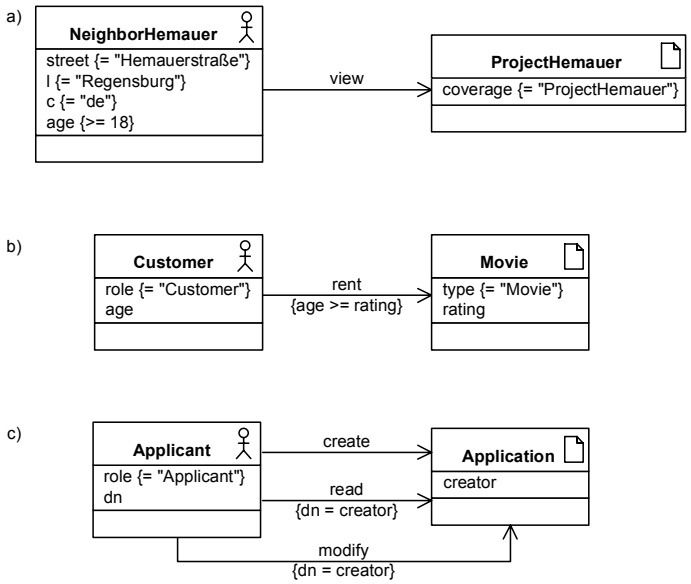

Figure 4: Beispielpolitiken in UML-Notation

4 Abbildung klassischer Zugriffskontrollmodelle

Ziel dieses Abschnittes ist es zu zeigen, dass ABAC als Generalisierung auch traditioneller Zugriffskontrollmodelle gesehen werden kann und dass diese auf ABAC abgebildet werden können. Hierzu wird im Folgenden dargestellt, wie ABAC-Politiken zur Umsetzung benutzerbestimmbarer Zugriffskontrolle (DAC), systembestimmter Zugriffskontrolle (MAC) und rollenbasierter Zugriffskontrolle (RBAC) definiert werden können.

4.1 Benutzerbestimmbare Zugriffskontrolle (DAC)

Die benutzerbestimmbare Zugriffskontrolle basiert auf dem Eigentümerprinzip. Jeder Benutzer ist selbst für die Sicherheit seiner Objekte zuständig, d.h. er allein entscheidet über die Vergabe, Weitergabe und Änderung der Autorisierungen für Objekte, die er selbst erstellt hat. Hierzu wird für jedes Paar (Subjekt, Objekt) festgelegt, ob das Subjekt ein Zugriffsrecht auf das Objekt besitzt und welcher Art ein eventuelles Recht ist, d.h. welche Operationen das Subjekt ausführen darf [Eck04, S. 242]. Üblicherweise kann ein Subjekt auch die Rechtevergabe an andere Subjekte weitergeben (delegieren).

Das DAC zugrunde liegende Zugriffskontrollmodell ist das Prinzip direkter Autorisierungen zwischen Subjekten und Objekten. In ABAC werden dafür Subjekt- und Objektdeskriptoren definiert. Daher ist die Abbildung einer DAC-Politik auf ABAC dadurch zu

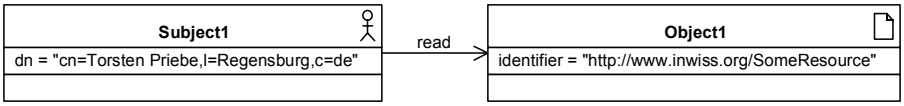

Figure 5: DAC-Beispiel als ABAC-Politik

bewerkstelligen, dass für jedes Objekt ein Objektdeskriptor und für jedes Subjekt ein Subjektdeskriptor definiert wird, wobei jeweils eine Qualifizierung über ein eindeutig identifizierendes Attribut erfolgt. Im Falle von X.500 ist das der „Distinguished Name“ (dn), im Falle von Dublin Core der „Identifier“. Abbildung 5 zeigt ein einfaches DAC-Beispiel, umgesetzt als ABAC-Politik.

Erweiterte DAC-Konzepte sind die oben erwähnte Delegation der Rechtevergabe und Einschränkungen durch Prädikate. Delegation lässt sich durch Konstrukte des ABAC-Modells nicht direkt ausdrücken, da wir explizit keine objektbezogene Administration betrachten. Das Prinzip lässt sich aber ggf. auf Anwendungsebene durch entsprechende Zugriffsarten und Autorisierungen nachbilden. Prädikate in DAC entsprechen hingegen weitestgehend den Bedingungen in ABAC. Die Einschränkung von Autorisierungen abhängig von der Uhrzeit wurde bereits in Abschnitt 2 als Beispiel genannt.

4.2 Systembestimmte Zugriffskontrolle (MAC)

Während die benutzerbestimmbare Zugriffskontrolle eine individuelle Definition von Zugriffsrechten ermöglicht, basiert das systembestimmte Zugriffskontrollmodell auf systemweit festgelegten Regeln. Neben der Kontrolle des direkten Zugriffs wird hier auch der Informationsfluss mit einbezogen. Ein Benutzer mit Zugriff auf bestimmte, sensitive Informationen soll so daran gehindert werden, diese Informationen an andere Benutzer, die keinen direkten Zugriff auf diese Informationen haben, weiterzugeben [Eck04, S. 243].

Der wohl bekannteste Vertreter der systembestimmten Zugriffskontrollmodelle ist das Modell nach Bell und LaPadula. Es entstammt dem militärischen Bereich, wo es üblich ist, Subjekte und Objekte zu klassifizieren bzw. in Schutzklassen (z.B. Top Secret > Secret > Unclassified) einzuteilen. Dabei wird die Schutzklasse des Subjektes als Freigabe (Clearance) bezeichnet und die des Objekts als Klassifikation (Classification). In diesem Modell wird Zugriff und Informationsfluss mittels zweier Regeln kontrolliert: Ein Subjekt hat Leserechte auf ein Objekt, wenn seine Freigabe höher oder gleich der Klassifikation des Objektes ist (sog. Simple-Security-Eigenschaft). Ein Subjekt hat Schreibrechte auf ein Objekt, wenn seine Freigabe niedriger oder gleich der Klassifikation des Objektes ist. Die zweite Regel (die sog. *-Eigenschaft) verhindert dabei einen Informationsfluss von oben nach unten in der Klassifizierungshierarchie [Eck04, S. 266].

Zur Abbildung in ABAC wird für alle Subjekte ein einziger Subjektdeskriptor mit einem Attribut „Clearance“, sowie für die Objekte ein Objektdeskriptor mit Attribut „Classification“ definiert. Es wird vorausgesetzt, dass Subjekte und Objekte über diese Attribute

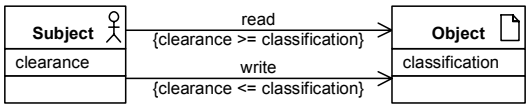

Figure 6: MAC-Modell (nach Bell & LaPadula) als ABAC-Politik

entsprechend klassifiziert sind. Die genannten beiden Regeln nach Bell und LaPadula lassen sich dann problemlos durch ABAC-Bedingungen definieren (siehe Abbildung 6).

4.3 Rollenbasierte Zugriffskontrolle (RBAC)

Das mittlerweile standardisierte rollenbasierte Zugriffskontrollmodell [FSG+01] fußt auf dem Konzept der Rolle als Mittler zwischen Subjekten und Berechtigungen. Die Rollen orientieren sich dabei an der organisatorischen Struktur der Einsatzumgebung. Der Hintergedanke dabei ist, dass eine Rollendefinition im Zeitablauf relativ unveränderlich ist, im Vergleich zu der Menge von Subjekten, die einer Rolle zugeordnet sind. Durch dieses Mittlerkonzept können sowohl die Rollendefinitionen auf der einen Seite als auch die Zuweisung von Subjekten zu bestimmten Rollen auf der anderen Seite unabhängig voneinander verändert werden, was die Rechteverwaltung vereinfacht. Zur Unterstützung des Prinzips der „Least Privileges“ wird ein Sitzungskonzept verwendet, wobei ein Benutzer in einer Sitzung eine Teilmenge der ihm zugewiesenen Rollen aktivieren kann.

Eine Abbildung des RBAC-Modells in ABAC geschieht dermaßen, dass ein Attribut „Role“ eingeführt wird, dessen Werte auf der Subjektseite die Rollenzugehörigkeiten darstellen. Daneben wird für jede Rolle ein entsprechender Subjektdeskriptor definiert. In Kombination mit Objektdeskriptoren, die jeweils ein bestimmtes Objekt bezeichnen (z.B. über das Dublin-Core-Element „Identifier“), bildet die Rechtezuweisung zwischen den Descriptoren das Analogon zur Beziehung zwischen Rollen und Berechtigungen. Das Sitzungskonzept von RBAC wird durch das erweiterte ABAC-Modell direkt unterstützt. Beispiele für die Verwendung eines Rollenattributes finden sich in Abbildung 4 b) und c).

5 Implementierung in CSAP

Um den praktischen Einsatz von ABAC zu untersuchen, wurde das in Abschnitt 2.1 beschriebene ABAC-Grundmodell als Erweiterung des Sicherheitsmoduls CSAP implementiert. Die Implementierung kann in diesem Beitrag nur kurz beschrieben werden; ausführlichere Informationen finden sich in [PMDP04].

Das CSAP-Modul wurde im Zuge des EU-Projekts „Webocracy“ (IST-1999-20364) als Sicherheitskomponente des e-Government-Portals „Webocrat“ entwickelt[5]. Es bietet Di-

[5] `http://www.webocrat.org`

enste zu Authentifikation, Autorisierung, Sitzungsmanagement und Auditing. Das in Java implementierte CSAP basiert auf einem Schichtenmodell, separiert in Dienste- und Datenhaltungsschicht, um die Art der Speichertechnik für die Diensteschicht transparent und austauschbar zu halten. Außerdem stellt CSAP auf der Diensteschicht ein Plug-In-Konzept bereit, das es erlaubt, alternative Implementierungen der Sicherheitsdienste in das Modul einzufügen und so flexibel verschiedene Varianten zur Verfügung zu stellen. In der ersten Version verfügte CSAP über einen passwortbasierten Authentifizierungs- sowie einen rollenbasierten Autorisierungsdienst.

ABAC wurde in der Implementierung als alternativer Autorisierungsdienst verwirklicht, der neben RBAC verwendet werden kann. Außerdem wurde das CSAP-Modul in die Portalumgebung INWISS[6] [Pri04] integriert, wo es die Authentisierung, Autorisierung und die Benutzerverwaltung übernimmt.

Bei der praktischen Anwendung des ABAC-Modells zeigt sich, dass die Performance der Implementierung aufgrund der Komplexität von ABAC beim Einsatz z.B. in einer Suchmaschine, bei der möglicherweise mehrere Tausend Zugriffskontrollentscheidungen in kurzer Zeit getroffen werden müssen, unzureichend ist. Da INWISS für die Objektmetadaten Semantic-Web-Techniken wie RDF und OWL verwendet, erscheint die Ausnutzung einer entsprechenden Inferenzmaschine auch zur Auflösung von ABAC-Subjekt- und Objektdeskriptoren vielversprechend. Noch ausstehend sind außerdem administrative Funktionen zur Pflege der ABAC-Sicherheitspolitik. Analog zur webbasierten RBAC-Administration [DMP04] streben wir eine Portletimplementierung innerhalb von INWISS an. Hier kann die Notation aus Abschnitt 3 möglicherweise als Grundlage dienen.

6 Zusammenfassung und Ausblick

Im vorliegenden Beitrag wurde – basierend auf in der Literatur vorgeschlagenen Modellen – ein Referenz-Zugriffskontrollmodell vorgestellt, das durch die Verwendung von Attributen zur Zugriffskontrollentscheidung eine flexible Rechteverwaltung erlaubt. Durch die Einführung von Sitzungen und die Einbeziehung von Umweltzuständen werden umfassende Möglichkeiten zur Modellierung von Zugriffsrechten eingeräumt. Daneben wurde eine Methode vorgestellt, Zugriffskontrollpolitiken für das ABAC-Modell grafisch unter Verwendung von UML zu entwerfen. Dies wurde an Hand der klassischen Zugriffskontrollstrategien DAC, MAC und RBAC demonstriert. Dabei zeigte sich, dass ABAC weitgehend in der Lage ist, die genannten Modelle zu subsumieren und unter einem gemeinsamen Dach zusammenzufassen. Schließlich wurde eine Implementierung des ABAC-Modells in einem Sicherheitsmodul vorgestellt.

Zukünftige Arbeiten werden vor allem die vollständige Implementierung des Referenzmodells sowie die Entwicklung einer Administrationskomponente, die in ein Portal eingebunden werden kann, umfassen. Weiterhin werden wir die Verwendbarkeit von Semantic-Web-Technologien, insbesondere Ontologien und Inferenzmaschinen, zur optimierten Umsetzung einer attributbasierten Zugriffskontrolle untersuchen.

[6] `http://www.inwiss.org`

References

[AABF02] Adam, N.R., Atluri, V., Bertino, E., Ferrari, E.: A Content-based Authorization Model for Digital Libraries. *IEEE Transactions on Knowledge and Data Engineering*, Volume 14, Number 2, März/April 2002.

[Bis02] Biskup, J.: Credential-basierte Zugriffskontrolle: Wurzeln und ein Ausblick. *32. Jahrestagung der Gesellschaft für Informatik e.V. (GI)*, Dortmund, Germany, September/Oktober 2002.

[DMP04] Dridi, F., Muschall, B., Pernul, G.: Administration of an RBAC System. *Proc.of the 37th Hawaiian International Conference on System Sciences (HICSS 2004)*, Hawaii, USA, Januar 2004.

[Eck04] Eckert, C.: *IT-Sicherheit: Konzepte – Verfahren – Protokolle*, 3. Auflage, Oldenbourg Verlag, 2004.

[FSG+01] Ferraiolo, D.F., Sandhu, R., Gavrila, S., Kuhn, D., and Chandramouli, R.: Proposed NIST Standard for Role-based Access Control. *ACM Transactions on Information and Systems Security*, Volume 4, Number 3, August 2001.

[ITU96] X.520: The Directory – Selected Attribute Types. ITU-T Recommendation, 1996.

[ITU00] X.509: The Directory – Public Key and Attribute Certificate Frameworks. ITU-T Recommendation, 2000.

[OAS03] eXtensible Access Control Markup Language (XACML), Version 1.1. OASIS Community Specification, August 2003. `http://www.oasis-open.org/committees/xacml/`

[OPS00] Oppliger, R., Pernul, G., Strauss, C.: Using Attribute Certificates to implement Role-based Authorization and Access Control. *Proceedings of the 4th Conference on „Sicherheit in Informationssystemen“ (SIS 2000)*, Zürich, Schweiz, Oktober 2000, S. 169-184.

[PS04] Park, J., Sandhu, R.: The UCON_{ABC} usage control model. *ACM Transactions on Information Systems Security*, 7(1), pp. 128-174, Februar 2004.

[PFMP04] Priebe, T., Fernandez, E.B., Mehlau, J.I., Pernul, G.: A Pattern System for Access Control. *Proc. 18th Annual IFIP WG 11.3 Working Conference on Data and Application Security*, Sitges, Spanien, Juli 2004.

[PMDP04] Priebe, T., Muschall, B., Dobmeier, W., Pernul, G.: A Flexible Security System for Enterprise and e-Government Portals. *Proc. of the 15th International Conference on Database and Expert Systems Applications (DEXA 2004)*, Zaragoza, Spanien, September 2004.

[Pri04] Priebe, T.: INWISS – Integrative Enterprise Knowledge Portal. Demonstration at the 3rd International Semantic Web Conference (ISWC 2004), Hiroshima, Japan, November 2004.

[W3C02] The Platform for Privacy Preferences 1.0 (P3P 1.0) Specification. W3C Recommendation, 2002. `http://www.w3.org/TR/2002/REC-P3P-20020416/`