



Die modellgetriebene Entwicklung eines anwendungsunabhängigen Rollen- und Rechtemanagementsystems

DIPLOMARBEIT

zur Erlangung des akademischen Grades
Diplom-Informatiker

FRIEDRICH-SCHILLER-UNIVERSITÄT JENA
Fakultät für Mathematik und Informatik

eingereicht von Steffen Gemein
geb. am 23.03.1986 in Mühlhausen

Themenverantwortlicher: Prof. Dr. Wilhelm R. Rossak
Betreuer: Dipl.-Inf. Christian Schachtzabel

Jena, 17. Mai 2010

Zusammenfassung

Gegenstand dieser Diplomarbeit ist die Entwicklung eines formalen Metamodells für ein Rollen- und Rechtemanagement, welches unabhängig von der Anwendungsdomäne arbeitet und somit für die Rechtemodellierung beliebiger Anwendungen verwendet werden kann. Das als Grundlage verwendete, theoretische Role Based Access Control - Modell wird erweitert, sodass zusätzlich die Kontextinformationen einer Rolle erfasst und für die Zugriffskontrolle einer geschützten Ressource ausgewertet werden können.

Weiterhin wird ein Codegenerator entwickelt, um die Modellinstanz einer Rollen- und Rechtekongfiguration auf lauffähigen Java-Code abzubilden. Modell und Generator können in Kombination genutzt werden, um eine Zugriffsverwaltung einer beliebigen Java-Anwendung mit Mehrbenutzerbetrieb zu erstellen und in diese zu integrieren. Zudem ist es möglich, eine Dokumentation der Rechtekongfiguration aus dem Modell in verschiedene Ausgabeformate automatisch zu generieren. Die Abbildung der modellierten Rollen und Berechtigungen auf bestehende Frameworks ist ebenso durchführbar und wird an der Security-API des Frameworks Apache Shiro demonstriert.

Inhaltsverzeichnis

1. Einleitung	5
2. Grundlagen	7
2.1. Terminologie	7
2.2. Zugriffskontrolle und Zugriffsrechte	8
2.3. Authentifizierung (AuthN), Authentisierung, Autorisierung (AuthZ) . . .	10
2.4. Datenstrukturen für Zugriffskontrolle	12
2.4.1. Zugriffskontrollmatrix (ACM)	12
2.4.2. Befähigungslisten und Zugriffskontrolllisten (ACL)	13
2.4.3. Protection Bits	15
2.5. IT-Sicherheitskriterien	16
2.6. Sicherheitsstrategien und Sicherheitsmodelle	18
2.6.1. Discretionary Access Control (DAC)	19
2.6.2. Mandatory Access Control (MAC)	22
2.6.3. Role Based Access Control (RBAC)	26
3. Anforderungen an das Rechtemanagementsystem	30
4. Bestehende Systeme und Lösungsansätze	34
4.1. Security Patterns	34
4.2. Apache Shiro (ehem. JSecurity)	37
5. Implementierung einer modellgetriebenen Lösung	39
5.1. Begriffsdifferenzierungen und Entwicklungsprozesse	40
5.2. Das Grundmodell	45
5.2.1. Model	45
5.2.2. RoleType	47
5.2.3. Action	48
5.2.4. Matching	49
5.2.5. ContextParam	51
5.3. Das erweiterte Modell	52
5.3.1. Vorbedingungen in Aktionen	52
5.3.2. Abhängigkeiten zwischen Actions	54
5.3.3. Matchings mit Expressions	56
5.3.4. Zusammenfassung von Matchings mit gleichen Expressions . . .	59
5.3.5. Globale Matchings	60
5.4. Codegenerierung für Java-Anwendungen	63
5.5. Codegenerierung für andere Zielplattformen	68
5.6. Dokumentationsgenerierung	70
5.6.1. Ausgabe im Confluence Enterprise Wiki	71
5.6.2. Ausgabe als Datei im Portable Document Format	72
5.7. Einbettung in andere Anwendungen	73

6. Evaluierung der modellgetriebenen Lösung	76
6.1. Umstellung der Zugriffskontrolle im Projekt <code>doc-in</code>	76
6.2. Vergleich zum RBAC-Modell	79
6.3. Mögliche Verbesserungen und geplante Erweiterungen	79
Literaturverzeichnis	83
Verzeichnis der Abbildungen	86
Verzeichnis der Tabellen	87
Verzeichnis der Programmquelltexte	88
7. Anlagen	89
A. CD-ROM	89
B. Klassendiagramm des erweiterten Rollen- und Rechtemodells	90
C. Ausschnitt der Rollen- und Rechtedokumentation aus <code>doc-in</code>	91

1. Einleitung

Im IT-Bereich kommt es nicht selten vor, dass verfügbare Ressourcen von mehr als einem Benutzer verwendet werden. Hier entscheidet eine Zugriffskontrolle (engl. access control), ob einem Benutzer der Zugriff auf eine angeforderte Resource gewährt oder verweigert wird. Die Zugriffskontrolle stellt die Integrität, Verfügbarkeit und Vertraulichkeit der Informationen sicher.

Diese Diplomarbeit beschäftigt sich ausschließlich mit der logischen Zugriffskontrolle in Software. Ein von der Anwendungslogik entkoppeltes Rollen- und Rechtemanagement übernimmt die Autorisierung im Mehrbenutzerbetrieb einer Software. Die Grundlage ist das Role Based Access Control (kurz RBAC) - Modell, welches 1992 von D.F Ferraiolo und D.R. Kuhn beschrieben und 2004 als ANSI-Norm 359-2004 verabschiedet wurde. Das RBAC Modell ordnet Identitäten mit Hilfe von Rollen Zugriffsrechte zu. Dieser Ansatz ist die Weiterentwicklung des Discretionary Access Control (DAC) - Modells, in der die Autorisierung allein anhand der Identitäten entschieden wurde.

Ziel dieser Diplomarbeit ist ein lauffähiges Rollen- und Rechtemanagementsystem für die Zielplattform Java, welches als Grundlage das bereits beschriebene RBAC-Modell verwendet. Dabei gilt es, verschiedene Anforderungen umzusetzen: Die Zugriffskontrolle erfolgt zentralisiert und lose gekoppelt von der Anwendungslogik. Damit im Zusammenhang steht die anwendungsübergreifende Nutzung des Rollen- und Rechtemanagementsystems. Änderungen am Rechtemodell müssen kostengünstig und fehlerfrei umsetzbar sein, auch nach der Auslieferung der Hostanwendung. Weiterhin soll automatisiert eine Dokumentation des Rechtemodells für den Kunden erstellt werden. Die Rollentypen und Autorisierungsmethoden müssen parametrisierbar sein, damit die Zugriffskontrolle im Anwendungskontext erfolgen kann. Der Autorisierungsmechanismus darf keinen Performanceengpass für die Hostanwendung verursachen.

Bei der Implementierung des Rollen- und Rechtemanagementsystems kommt die Technik der modellgetriebenen Softwareentwicklung (englisch Model Driven Software Development, kurz MDSD) zum Einsatz. Das Rechtemodell wird in einer XML-Datei spezifiziert. Aus diesem Modell erzeugt ein Codegenerator mit Hilfe von Schablonen lauffähigen Java-Code. Diese deklarative Herangehensweise verlagert den Schwerpunkt des Entwicklungsprozesses eines Rechteframeworks von der eigentlichen Softwareentwicklung, auf die Modellierung des Sachverhalts. Durch die Trennung des fachlichen Rechtemodells von der technischen Abbildung wird die Anpassbarkeit und Konfigurierbarkeit erhöht. Die Software wird robuster gegen Änderungen, welche spät im Softwarelebenszyklus durchgeführt werden. Wiederkehrende und nahezu identische Codeblöcke, wie sie in den Autorisierungsmethoden häufig anzutreffen sind, werden generiert und nicht von Hand geschrieben. Dadurch sinkt die Fehlerquote.

Die Strukturierung der Arbeit erfolgt in drei Teilen. Der erste Schwerpunkt ist die theoretische Betrachtung der Problemstellung. Es erfolgt eine Einführung in die Terminologie. Verschiedene Rechtemodelle und Ansätze werden unter Anderem an bestehende Systeme

men erläutert und analysiert.

Im Anschluss daran wird der Entwicklungsprozess der Software begleitet und dokumentiert. Die Art und Weise der Anforderungsumsetzung des Rollen- und Rechtemanagementsystems wird begründet und erläutert.

Den Abschluss der Arbeit bildet die Evaluierung der lauffähigen Software. Welche Anforderungen wurden umgesetzt, welche nicht und warum? Was sind mögliche Verbesserungen? Wie kann das System erweitert werden? Weiterhin wird die Umstellung des Autorisierungssystems eines bestehenden Projektes auf die modellgetriebene Variante evaluiert.

Literatur

- [1] Apache Incubator. Incubated Projects. <http://incubator.apache.org/projects/index.html> Zugriff am 19.02.2010.
- [2] Apache Shiro. <http://cwiki.apache.org/confluence/display/SHIRO> Zugriff am 18.02.2010.
- [3] Bell, David E.; La Padula, Leonard J.: Secure Computer System Unified Exposition and Multics Interpretation. MITRE Corporation. Massachusetts 1976. <http://csrc.nist.gov/publications/history/bell76.pdf>. Zugriff am 03.03.2010
- [4] Biba, K. J.: Integrity Considerations for Secure Computer Systems MITRE Corporation. Massachusetts 1977.
- [5] Bishop, Matt: Introduction to Computer Security. Addison-Wesley. Boston 2004.
- [6] Bundesamt für Sicherheit in der Informationstechnik. IT-Sicherheitskriterien. https://www.bsi.bund.de/DE/Themen/ZertifizierungundAkkreditierung/ZertifizierungnachCCundITSEC/ITSicherheitskriterien/itsicherheitskriterien_node.html Zugriff am 02.03.2010
- [7] Bundesamt für Sicherheit in der Informationstechnik. IT-Sicherheitskriterien und Evaluierung nach ITSEC. https://www.bsi.bund.de/DE/Themen/ZertifizierungundAkkreditierung/ZertifizierungnachCCundITSEC/ITSicherheitskriterien/ITSEC/itsec_node.html Zugriff am 05.03.2010
- [8] Confluence 3.0 User Guide (PDF). <http://confluence.atlassian.com/display/ALLDOC/Confluence+Documentation+Directory> Zugriff am 19.11.2009
- [9] Denning, Dorothy E.: A lattice model of secure information flow. Communications of the ACM 19(5). 1976. <http://faculty.nps.edu/dedennin/publications/lattice76.pdf> Zugriff am 03.03.2010
- [10] Eilebrecht, Karl; Starke, Gernot: Patterns kompakt. Entwurfsmuster für effektive Software-Entwicklung. 2. Auflage. Spektrum Akademischer Verlag. Heidelberg, 2006.
- [11] Ferraiolo, David F.; Kuhn, D. Richard: Role Based Access Control. 15th National Computer Security Conference. Gaithersburg 1992. <http://csrc.nist>.

- gov/groups/SNS/rbac/documents/ferraiolo-kuhn-92.pdf Zugriff am 05.03.2010.
- [12] Ferraiolo, David F.; Kuhn, D. Richard: Role-Based Access Control. 2nd Edition. Artech House. Boston 2007.
- [13] FreeMarker: Java Template Engine Library. Online Manual. <http://www.freemarker.org/docs/api/index.html> Zugriff am 12.01.2010.
- [14] Harrison, Michael A.; Ruzzo, Walter L.; Ullman, Jeffrey D.: Protection in Operating Systems. Communications of the ACM 19(8). 1976. <http://www.cs.unibo.it/babaoglu/courses/security/resources/documents/harrison-ruzzo-ullman.pdf> Zugriff am 01.03.2010
- [15] Juric, Matjaz; Nashi, Nadia; Berry, Craig: J2EE Design Patterns Applied. Real World Development with Pattern Frameworks. 1st Edition. Wrox Press. New Jersey, 2002.
- [16] JSecurity. Easy Java Security. <http://www.jsecurity.org/> Zugriff am 18.02.2010.
- [17] Lampson, Butler W.: Protection. Proceedings of the 5th Princeton Conference on Information Sciences and Systems. Princeton 1971. <http://research.microsoft.com/en-us/um/people/blampson/08-Protection/Acrobat.pdf> Zugriff am 01.03.2010
- [18] Massol, Vincent u.a.: Better Builds with Maven. The How-to Guide for Maven 2.0. Version 1.7.0. Exist Global. USA 2008.
- [19] Michaelis, Samuel; Schmiesing, Wolfgang: JAXB 2.0. Ein Programmier tutorial für die Java Architecture for XML Binding. Carl Hanser Verlag. München 2007.
- [20] Microsoft TechNet. How Security Descriptors and Access Control Lists Work. <http://technet.microsoft.com/en-us/library/cc781716%28WS.10%29.aspx> Zugriff am 26.02.2010.
- [21] National Security Institute: 5200.28-STD Trusted Computer System Evaluation Criteria (TCSEC). <http://csrc.nist.gov/publications/history/dod85.pdf> Zugriff am 02.03.2010
- [22] Peh, Diana u.a.: BIRT: A Field Guide to Reporting. 2nd Edition. Addison-Wesley. Laffin, Pennsylvania 2008.
- [23] Pfleeger, Charles P.; Pfleeger, Shari Lawrence: Security in Computing. 4th Edition. Prentice Hall PTR. New Jersey 2006.

-
- [24] Pieprzyk, Josef; Hardjono, Thomas; Seberry, Jennifer: Fundamentals of Computer Security. Springer-Verlag. Berlin 2003.
- [25] ubuntuusers / Wiki / ACL. <http://wiki.ubuntuusers.de/ACL> Zugriff am 02.03.2010.
- [26] Ullenboom, Christian: Java ist auch eine Insel. Programmieren mit der Java Standard Edition Version 5. Galileo Press. Bonn 2006.
- [27] Stahl, Thomas; Völter, Markus; Efftinge, Sven; Haase, Arno: Modellgetriebene Softwareentwicklung. Techniken, Engineering, Management. 2. Auflage. d.punkt Verlag. Mai 2007

Verzeichnis der Abbildungen

1.	Hauptaspekte der IT-Sicherheit	9
2.	Wechselspiel zwischen Authentifizierung und Authentisierung	11
3.	Schematische Darstellung einer Zugriffskontrollmatrix	12
4.	Hasse-Diagramm zur Potenzmenge von $\{\alpha, \beta, \gamma\}$	23
5.	Beispiel für eine Autorisierung im RBAC-Modell	28
6.	Aufbau des Security Frameworks Apache Shiro [2]	38
7.	Schritt 1: Benutzer werden direkt mit ihren autorisierten Aktionen verbunden	41
8.	Schritt 2: Benutzer werden indirekt über ihre Rolle mit den autorisierten Aktionen verbunden	42
9.	Schritt 3: Die Anzahl der statischen Rollentypen und Aktionen wird durch die Übergabe von Parametern und dynamischen Rollen stark reduziert	43
10.	Das Grundmodell des Rollen- und Rechtemanagers	46
11.	Preconditions im erweiterten Modell	53
12.	ActionConstraints im erweiterten Modell	55
13.	Expressions im erweiterten Modell	57
14.	Matching-RoleType-Assoziation im erweiterten Modell	59
15.	MatchConstraints im erweiterten Modell	61
16.	Aus Datenmodell und Schablone wird mit Freemarker eine Java-Klasse erzeugt	64
17.	Interne Darstellung der Rollentypen und deren Kontextinformationen	65
18.	Aktivitätsdiagramm einer Autorisierungsmethode	66
19.	Statische Sicht der generierten Klassen	68
20.	Zusammenspiel der generierten Klassen mit der Security-API Apache Shiro	70
21.	Vergleich der Lines of Code vor und nach der Umstellung des Autorisierungsmoduls von <code>doc-in</code>	77
22.	Verteilung der Expression-Typen im Rollen- und Rechtemodell von <code>doc-in</code>	78
23.	Rechtezuordnung mit Benutzer- und Rollentypgruppen	81
24.	Das erweiterte Modell des Rollen- und Rechtemanagers	90

Verzeichnis der Tabellen

1.	Beispiel für eine Zugriffskontrollmatrix	13
2.	Überführte Zugriffskontrollmatrix als Capability-List	14
3.	Überführte Zugriffskontrollmatrix als Access Control List	14
4.	Rechteübersicht für die Berechtigung (rwx)(r-x)(r-) durch Protection Bits	16
5.	Historie der Entwicklung von IT-Sicherheitskriterien	17
6.	Sicherheitsstufen der Trusted Computer Security Evaluation Criteria . . .	17
7.	Vergleich der Stufen der IT-Sicherheitskriterien	18
8.	Zustandsüberführungen der primitive Operationen des HRU-Modells und deren Vorbedingungen	21

Verzeichnis der Programmquelltexte

1.	Genereller Aufbau eines Commands im HRU-Modell	22
2.	Unix-Command im HRU-Modell zum Anlegen einer Datei	22
3.	Das Rechtemodell in der Grundform	47
4.	Beispielhafte Rollentypdefinitionen	48
5.	Beispielhafte Matchingdefinitionen	50
6.	Precondition-Definition innerhalb einer Action	53
7.	Doppelt definierte Rollenmatchings in einem Read-/Write-Actionpaar . .	54
8.	ActionConstraint-Definition innerhalb einer Action	56
9.	Expression-Definition innerhalb eines Matchings	58
10.	Zusammenfassung aller Matchings mit einer TrueExpression	59
11.	MatchConstraint-Definition innerhalb einer Action	62
12.	Schablonenausschnitt für die Wiki-Dokumentation	71
13.	Ausschnitt des Datenmodells für die PDF-Generierung	73
14.	Beispielhafte Konfiguration des Maven-Plugins	75