Sicherheitsdienste für mobile Agentenanwendungen

Jürgen Bohn[1] und Günter Karjoth[2]

[1] ETH Zürich, Haldeneggsteig 4, IFW,
CH-8092 Zürich, Schweiz
bohn@inf.ethz.ch
[2] IBM Forschungslabor Zürich, Säumerstrasse 4,
CH-8803 Rüschlikon, Schweiz
gka@zurich.ibm.com

Zusammenfassung Mobile Agenten ermöglichen eine flexible Ausführung von verteilten Anwendungen, was sie attraktiv macht für den Einsatz im Bereich des elektronischen Handels, Netzwerkmanagements und der Groupware. Der erfolgreiche Einsatz mobiler Agenten hängt aber auch davon ab, in wieweit der Schutz von Agenten gewährleistet ist. Wir identifizieren generische Sicherheitsdienste zum Schutz mobiler Agenten, welche obige Anwendungsszenarien abdecken, und stellen diese in einem Rahmenwerk dem Anwendungsentwickler bereit. In einer prototypischen Implementierung bieten wir unterschiedliche Realisierungen des gleichen Dienstes an, um gegebenenfalls von der Verfügbarkeit sicherer Hardware zu profitieren. Eine mobile Agentenanwendung benutzt diese Dienste, um eine sichere Produktrecherche zu realisieren.

1 Motivation

Die Attraktivität der mobilen Agententechnologie [9] beruht insbesondere auf möglichen Anwendungsszenarien aus den Bereichen elektronischer Handel, Netzwerkmanagement und Groupware. Eine Stärke des Paradigmas mobiler Agenten liegt darin, daß es den Entwurf, die Implementierung und die Wartung verteilter Anwendungen wesentlich einfacher und intuitiver gestaltet. In verteilten Systemen lassen sich die autonomen, miteinander kooperierenden Parteien in natürlicher und verständlicher Weise als mobile Agenten modellieren. Während für viele Einzelfälle herkömmliche Lösungsansätze ohne den Gebrauch mobiler Agenten existieren, so stellt das Paradigma mobiler Agenten ein Rahmenwerk zu Verfügung, welches all diese Fälle gleichzeitig abdeckt und behandelt [3].

Trotz der anfänglichen Euphorie hat sich die Technologie nur zögerlich verbreitet. Dies liegt nicht zuletzt daran, daß ein Grundproblem der mobilen Agenten, die Frage nach der Sicherheit, noch nicht zufriedenstellend gelöst ist. Während der Schutz des Wirtssystems vor böswilligen Agenten bereits relativ gut verstanden ist, besteht noch Forschungsbedarf beim *Schutz des Agenten* vor zufälliger oder böswilliger Manipulation [10,11]. Für den Nutzer von mobilen Agentenanwendungen ist dieser Sicherheitsaspekt von besonderer Bedeutung. Mag der Ausfall einer Plattform im Gesamtsystem geduldet werden, die fehlende Zuverlässigkeit und der mangelnde Integritätsschutz der

eigenen mobilen Agenten wird jedoch kaum toleriert. Die Sicherheitseigenschaften mobiler Agenten sind deshalb für die Akzeptanz und Verbreitung von Agentenanwendungen ein kritischer Faktor.

Heutige Wirtssysteme schützten Agenten vor Manipulationsversuchen durch andere Agenten, in dem diese in isolierten Adressräumen ausgeführt werden. Dabei erlauben proxy-basierte Zugriffsmechanismen den Austausch von Agentenobjekten. Die Ausführungsplattform selber hat aber Zugriff auf alle unverschlüsselten Daten des Agenten und kann daher den internen Kontrollfluß ausspionieren und zum eigenen Vorteil manipulieren. Es stellt sich damit die Frage nach der Verlässlichkeit der ausführenden Plattform, und wie man sich gegen betrügerische Manipulationen schützen oder diese zumindest zuverlässig erkennen kann.

Einen ersten Ansatz von allgemeinen Sicherheitsdiensten für mobile Agenten findet man bei Karnik [7], der drei konkrete Mechanismen für Containerklassen zum Schutz der Daten mobiler Agenten beinhaltet.[1] Wir erweitern diese Dienste hin zu einem Rahmenwerk von generischen Sicherheitsdiensten, bei dem von der konkreten Realisierung abstrahiert wird und verschiedene Implementierungen der gleichen Dienstschnittstelle, z.B. mit oder ohne Verwendung sicherer Hardware, bereitgestellt werden können. Als Anwendungszenario zur Veranschaulichung dient die preisvergleichende Produktrecherche (*comparison shopping*): Der mobile Agent eines Kunden besucht eine Reihe von elektronischen Marktplätzen (Agentenplattformen), auf denen er von den jeweils vorhandenen Händleragenten ein Produktangebot einholt. Nachdem der mobile Agent eine gewisse Anzahl solcher Angebote gesammelt hat, kehrt er zum Ursprungsort zurück und übergibt seinem Auftraggeber das beste gefundene Angebot.

2 Ansätze zum Schutz mobiler Agenten

Die größte zu bewältigende Gefahr für mobile Agenten bleiben betrügerischer Rechnerplattformen, von denen eine vielzahl möglicher Angriffe ausgehen können, wie zum Beispiel das Auspionieren oder Manipulieren von Daten, Code oder Kontrollfluß [6,12]. Existierende Lösungen zum Schutz des Agenten vor unbefugten Dritten können danach unterschieden werden, ob sie aktiven oder passiven Schutz gewähren. Während die Verwendung von Kryptographie es erlaubt Daten innerhalb des Agenten zu verbergen und ihre Integrität zu überprüfen, scheint nur die Verwendung spezieller, vertrauenswürdiger Hardware in der Lage zu sein, die Manipulation von Agenten zu verhindern. Es gibt zwar bereits mehrere Lösungsansätze zum Schutz mobiler Agenten, die nur auf Software beruhen (u.a. verschlüsselte Funktionen, Detektionsobjekte, Agenten als zeitweilige Software-Blackbox, oder redundante Ausführung und Replikation von Agenten), ihre praktische Einsatzfähigkeit ist jedoch begrenzt [6].

Sichere, vertrauenswürdige Geräte erscheinen bezüglich ihrer Funktionsweise nach außen hin als eine Blackbox. Kritische Operationen des Agenten werden so ausgelegt, daß sie nur innerhalb dieser sicheren Geräte ausführbar sind und sich somit dem Einfluß des zugehörigen Rechners entziehen. Die Bandbreite derartiger Geräte reicht von nur

[1] Eine genaurere Untersuchung zeigt aber, daß es möglich ist nachträglich einzelne Objekte aus Karnik's `AppendOnlyContainer` zu entfernen, ohne das dies später erkannt werden kann [2].

kreditkartengroßen Chipkarten über komplette geschützte Rechner bis hin zur teuren Spezialhardware.

Auf Grund dieser Sachlage muß aber ein Rahmen von Sicherheitsdiensten gewährleisten, daß der Schutz der Ausführung mobiler Agenten gegen Manipulation und Ausspionieren sowohl durch existierende Lösungen, wie die Verwendung von sicheren, vertrauenswürdigen Geräten, als auch durch noch zu entwickelnde Verfahren realisiert werden kann.

Interoperable und sichere mobile Agentenanwendungen erfordern offene Agentensysteme, welche einen Rahmen von Sicherheitsdiensten besitzen, deren Schnittstellen bekannt sind und allen Teilnehmern zur Verfügung stehen. Neben den kryptographischen Dienstprimitiven, wie z.B. Methoden zur Verschlüsselung, digitale Signaturverfahren oder kryptographische Hashfunktionen oder den darauf aufbauenden bereits vorgestellten elementaren Sicherheitsdiensten, treten mit zunehmender Nähe zur Anwendung weitergehende Sicherheitsbedürfnisse und Forderungen auf. Um den zusätzlichen Anforderungen Rechnung zu tragen, werden problemspezifische Sicherheitsmechanismen und -protokolle erforderlich.

3 Klassifikation der zu schützenden Daten

Die Sicherheit mobiler Agenten erfordert den Schutz des Zustandes des Agenten und seiner Ausführung auf unsicheren Plattformen. Der Zustand eines Agenten ist dabei durch das auszuführende Programm und die mitgeführten Daten bestimmt. Sicherheitsdienste gewähren den Schutz der Integrität und Vertraulichkeit seiner mitgeführten Daten bzw. seiner Ausführung.

Während das Programm eines Agenten zur Laufzeit einen statischen Charakter[2] besitzt, lassen sich die Daten bezüglich ihrer Persistenz nach als statisch oder dynamisch klassifizieren. Eine weitere Untergliederung des Datenanteils ist nach semantischen Gesichtspunkten in Nutzdaten, Metadaten und Prüfdaten möglich. Als Nutzdaten bezeichnen wir alle jene Daten und Informationen, die zur Erledigung der primären Aufgaben des Agenten benötigt bzw. gesammelt werden, z.B. die gesammelten Angebote in der Produktrecherche. Die Metadaten seien diejenigen Daten, die nur indirekt zur Lösung der durch den Agenten zu bearbeitenden Aufgabenstellung beitragen, etwa Auftragsparameter oder die Liste der zu besuchenden Händler. Die Prüfdaten tragen nicht zur Erfüllung der eigentlichen Agententätigkeit bei, sondern dienen ausschließlich dem Schutz der Integrität und Vertraulichkeit der Nutzdaten, Metadaten oder gar des Codes selbst.

Die auf den verschiedenen Datenarten operierenden Parteien lassen sich in die folgenden drei Gruppen einteilen: Der mobile Agent selbst, eine von allen teilnehmenden Parteien als vertrauenswürdig anerkannte Instanz (Trusted Third Party, `TTP`), sowie alle anderen am System beteiligten Parteien (`Dritte`). Abbildung 1 zeigt eine baumartige Klassifikation der Datenarten mobiler Agenten. Die Begriffe in den Blättern nennen die Parteien, die i.a. auf den jeweiligen Datentyp zugreifen. Darunter werden in einer Liste die anfallenden Primitiven identifiziert, die typischerweise auf den entsprechenden Daten operieren, zusammen mit einem kurzen praktischen Beispiel. Bei den dynamischen

[2] Das dynamische Laden von Code ist nicht Bestandteil dieser Arbeit.

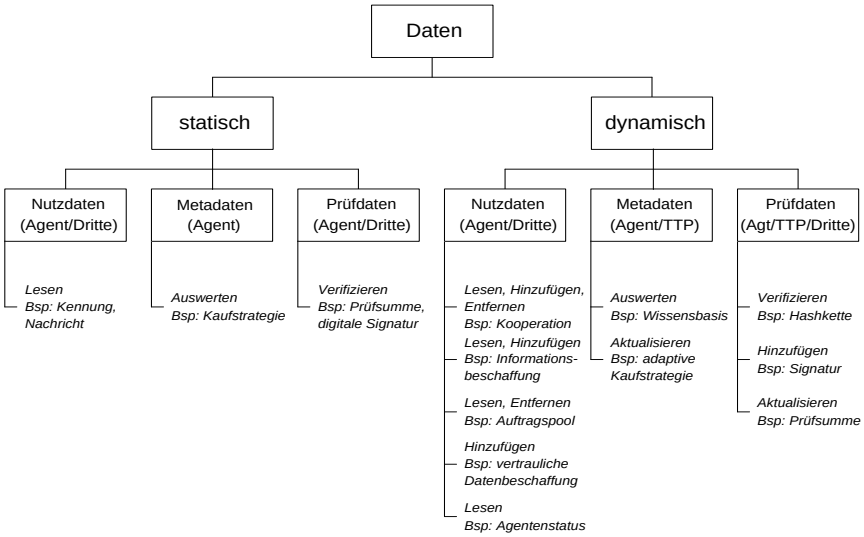

Abbildung 1: Klassifikation der Daten mobiler Agenten und Identifikation der darauf operierenden Dienstprimitiven.

Nutzdaten werden die Primitiven darüber hinaus gemäß ihrem möglichen Auftreten gruppiert.

Anhand dieser Gliederung lassen sich nun grundlegende Datentypen und Sicherheitsdienste für mobile Agentenanwendungen ableiten, wie in Tabelle 1 dargestellt. Es zeigt sich, daß viele elementare sichere Dienste sich durch verschiedene Ausprägungen von sicheren Containerklassen bereitstellen lassen. Als Container bezeichnen wir dabei – in der allgemeinsten Ausprägung – einen Behälter, dem Objekte beliebigen Typs hinzugefügt und entnommen werden können. Die Containerklasse stellt entsprechende Methoden bereit. Spezialisierungen sind beispielsweise möglich durch Einschränkung der zugelassenen Objektklassen (Typisierung) oder der Zugriffsmethoden (z.B. nur lesenden Zugriff gestatten).

Derartige Containerklassen haben zwei Hauptvorteile: Erstens sind sie für den Entwickler einfach zu benutzen und in Anwendungen zu integrieren, weil sie sich in Bezug auf die Funktionalität nur unwesentlich von herkömmlichen Containerklassen unterscheiden. Zweitens läßt sich so leicht die Trennung von Schnittstelle und Implementierung realisieren, so daß die tatsächlich verwendeten Sicherheitsmechanismen transparent zur Anwendungsentwicklung und zur Laufzeit gewählt werden können. Mit der gewählten Modellierung in Java wird diese Trennung analog zum SPI-Konzept aus `Java 2` gelöst, durch Definition von sogenannten Diensterbringer-Schnittstellen (*Service Provider Interfaces*, SPI).

Tabelle 1: Grundlegende Sicherheitsdienste für mobile Agentenanwendungen.

Datenart	Operation	sicherer Dienst	Java Dienstklasse
statisch			
Nutzdaten	Lesen	ausschließlich lesen	`ReadOnlyContainer`
Metadaten	Auswerten	sichere Berechnung	`SecureComputation`
Prüfdaten	Auswerten	kryptograph. Primitive	`java.security.*`
dynamisch			
Nutzdaten	Lesen, Hinzufügen, Entfernen	integritätsgeschützt abschließbar	`IntegrityStack`, `LockableContainer`
	Lesen, Hinzufügen	integritätsgeschützt und kein entfernen	`PushOnlyIntegrityStack`, `AppendOnlyContainer`
	Lesen, Entfernen	kein hinzufügen	`RemoveOnlyContainer`
	Hinzufügen	ausschließlich hinzufügen	`SecretAppendOnlyContainer`
	Lesen	ausschl. lesen, dyn. Inhalt	`DynamicReadOnlyContainer`
Metadaten	Auswerten	sichere Berechnung	`SecureComputation`
	Aktualisieren	sichere Berechnung	`SecureComputation`
		kryptograph. Primitive	`java.security.*`
Prüfdaten	Verifizieren	kryptograph. Primitive	`java.security.*`
	Auswerten	sichere Berechnung	`SecureComputation`
	Aktualisieren	kryptograph. Primitive	`java.security.*` bzw.
		sichere Berechnung	`SecureComputation`

4 Rahmen von Sicherheitsdiensten

`SecureComputation` ist eine abstrakte Dienstschnittstelle und steht für die sichere Berechnung bzw. Ausführung von Operationen, z.B. die Aktualisierung von Prüfdaten oder die Auswertung eines Angebotes auf einer SmartCard. Konkrete Realisierungen sind abhängig von der jeweiligen Anwendungskategorie. Die nutzdatenbezogenen Containerdienstklassen können jeweils auf einen bestimmten Kreis von Benutzern eingeschränkt werden. Daraus ergeben sich dann entsprechende Gruppencontainerdienste, wie z.B. `GroupLockableContainer` oder `GroupAppendOnlyContainer`, bei denen etwa ein bei der Benutzung erforderlicher geheimer Containerschlüssel mit den öffentlichen Public-Key Schlüsseln der ausgewählten Gruppenmitglieder verschlüsselt wird.

Diese Gerüst von elementaren Sicherheitsdiensten stellt Bausteine für den Entwurf sicherer Agentensysteme auf anwendungsnaher Ebene dar. Es dient als Grundlage für die Erstellung problemspezifischer, komplexerer Sicherheitsdienste.

Der `AppendOnlyContainer` erlaubt nur das Anhängen von Daten an den bereits vorhandenen Datenbestand sowie das Lesen der vorhandenen Daten. Die Realisierung erfolgt mit Hilfe des `PushOnlyIntegrityStack`-Dienstes, bei dem Objekte auf den Stack abgelegt, nicht aber entnommen werden dürfen. Die unerlaubte Entnahme oder Veränderung von Objekten wird je nach Implementierung entweder durch einen Software-Mechanismus erkannt oder durch entsprechende Prüfmaßnahmen innerhalb einer sicheren Hardware unterbunden.

Der `PushOnlyIntegrityStack` ist eine Spezialisierung von `IntegrityStack`, dessen Implementierung in zwei Abstraktionsschritten erfolgt. In der ersten

Stufe fußen die abstrakten Stackklassen auf einer generischen Diensterbringer-Schnittstelle nach dem SPI-Konzept, dem Interface `StackCryptoServices`, das allgemeine Methoden zur Bereitstellung der Stackfunktionalität definiert. Die zweite Stufe stellen Implementierungen dieser SPI-Schnittstelle dar, wie z.B. die Klasse `IntegrityStack`. Hier werden die sicherheitskritischen Funktionen verborgen und realisiert. Dabei findet die Abbildung der allgemeinen Stackprimitiven auf konkrete kryptographische Softwarebibliotheken oder sichere Hardware statt.

Eine prototypische Implementierung und Anwendung der beschriebenen Sicherheitsdienste erfolgte auf dem Aglets Agentensystem [8].

5 Anwendungsszenario: Sichere Produktrecherche

Um die Einsetzbarkeit der vorgestellten Sicherheitsdienste zu prüfen, implementierten wir eine mobile Agentenanwendung zur preisvergleichenden Produktrecherche [2]. Der mobile Agent verwendet den Dienst `AppendOnlyContainer`, um die Integrität der gesammelten Daten zu gewähren. Dabei kann man unter zwei Varianten auswählen, um mit oder ohne Einbezug von Chipkarten die gesammelten Daten sicher zu verknüpfen.

Für die chipkartenlose Variante implementierten wir ein Protokoll von Karjoth *et al.* [5], in der die kryptographischen Operationen auf dem Wirtssystem ausgeführt werden. Die andere Variante setzt auf den händlerseitigen Einsatz von Chipkarten. Die Implementierung beruht auf einer Spezialisierung eines Protokolls von Devanbu und Stubblebine, in dem sichere, aber resourcenlimitierte Hardware Stacks und Queues auf unsicheren Rechnern speichern können, während nur eine konstante Speichermenge innerhalb der sicheren Hardware benötigt wird [4]. Durch Verwendung der entsprechenden `AppendOnlyContainer`-SPI-Implementierungen können beide Protokolle in einfacher Weise in die Anwendung integriert werden.

Der sparsame Umgang mit Speicher auf der Chipkarte ist für mobile Agenten sehr wichtig, da nur einzelne Elemente des Stacks und die Kontrollinformationen in der sicheren Hardware für Prüfzwecke bearbeitet werden müssen, während die unter Umständen sehr umfangreichen restlichen Daten im externen Speicher gehalten werden können. Die sicherheitskritische Datenstruktur enthält dabei nur den geheimen Anker und den zuletzt berechneten Wert der Hashkette, mit deren Hilfe sich die Integrität der externen Daten überprüfen läßt, wie in Abbildung 2 gezeigt.

Unsere Implementierung benützt eine JavaCard [1], deren Leistungsfähigkeit und Speicherplatz groß genug ist, die Angebote der einzelnen Händler sicher in die Angebotsliste einzufügen. Immer wenn der mobile Agent auf einer unsicheren Plattform ein neues Angebot im `AppendOnlyContainer` ablegt, wird innerhalb der JavaCard die integritätsschützende Kontrolldateneinheit des unterliegenden `PushOnlyIntegrityStacks` entsprechend aktualisiert. Unerlaubte Operationen wie etwa das Entfernen von Objekten aus dem Container werden von der sicheren Hardware blockiert; Manipulationen an der Datenstruktur unter Umgehung der Blackbox können stets nachgewiesen werden (mit Ausnahme von Replay-Attacken). Alle JavaCards teilen sich dabei ein Public-Key-Schlüsselpaar. Zum Zeitpunkt der Initialisierung des Container werden die Kontrolldaten mit dem öffentlichen Blackbox-Schlüssel versiegelt und können fortan nur noch innerhalb der JavaCards entschlüsselt und modifiziert werden; sie entziehen

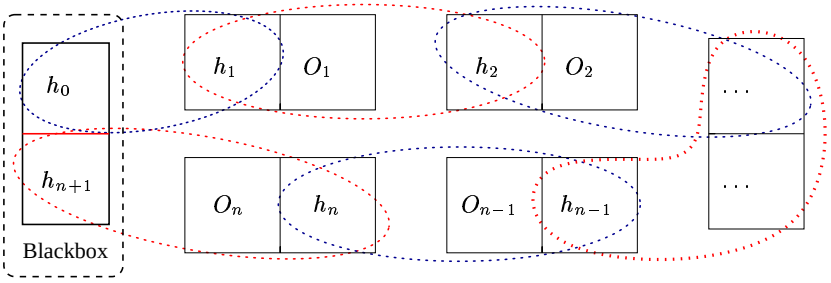

Abbildung 2: Aufbau der integritätsschützenden Hashkette: h_0 ist der geheime Startwert, h_{n+1} der Abschluß. Die rekursive Berechnung der Hashwerte h_i erfolgt über den durch die gestrichelten Ellipsen umfaßten Daten. Datentupel $\langle h_0, h_{n+1} \rangle$ erlaubt die Überprüfung der Hashkette und wird nur innerhalb der sicheren Hardware ausgewertet und aktualisiert.

sich somit dem Wirkungskreis der Händlerplattformen. Zur Ausgabe und Verwaltung der JavaCards wird eine Trusted Third Party benötigt, die sich z.B. aus dem Zusammenschluß der beteiligten Händler zu einem Händlerring ergeben könnte.

6 Ausblick

Ein Rahmen von Sicherheitsdiensten erlaubt die Erstellung sicherer, mobiler Agentenanwendungen in offenen Agentensystemen. Generische Basisdienste, wie etwa die Containerklassen, wie auch komplexere, anwendungsorientierte Sicherheitsdienste verringern den Entwicklungsaufwand. Am Beispiel der Produktrecherche wurde die Realisierung und der Einsatz exemplarischer Sicherheitsdienste des Rahens demonstriert. Eine Implementierung in Java mit dem Aglets Agentensystem liegt vor [2].

Da es zahlreiche Anwendungsgebiete und Einsatzszenarien für mobile Agenten gibt, stellt sich die Frage, ob ein Sicherheitsrahmen wie vorgestellt neben Basisdiensten auch anwendungsspezifische Sicherheitsdienste bereitstellen kann. Dazu untersuchen wir gegenwärtig verschiedene Anwendungskategorien, wie etwa E-Commerce (Comparison Shopping, Auktionen, Börsenmakler), Netzwerk- und Systemmanagement (Routing, Konfiguration, Wartung, Überwachung), Störungsmanagement (Unterstützung für fehlertolerante, robuste Dienste; Intrusion Detection), und Workflow-Management.

Die Auseinandersetzung mit dem Paradigma mobiler Agenten zeigt, daß diese Technologie ein großes Potential für zukünftige Entwicklungen und Einsatzszenarien bietet, nicht nur im elektronischen Handel. Die Frage der Sicherheit und des Schutzes mobiler Agenten ist und bleibt jedoch *das* Kernproblem. Der vorgestellte Rahmen von Sicherheitsdiensten bringt die mobilen Agenten dem praktikablen Einsatz einen Schritt näher. Dennoch ist das Grundproblem, der Schutz der Ausführung von Agenten auf nicht vertrauenswürdigen Plattformen, nicht grundsätzlich gelöst, da der Einsatz sicherer Hardware die Einbeziehung einer dritten, vertrauenswürdigen Partei erforderlich macht.

Literatur

1. M. Baentsch, P. Buhler, T. Eirich, F. Höring, and M. Oestreicher. JavaCard – from hype to reality. *IEEE Concurrency*, 7(4):36–43, Dez. 1999.
2. J. Bohn. Sicherheitsdienste für Mobile-Agenten-Anwendungen in elektronischen Märkten. Diplomarbeit, Universität Karlsruhe, Feb. 2000.
3. D. Chess, C. Harrison, and A. Kershenbaum. Mobile agents: Are they a good idea? In *Mobile Object Systems – Towards the Programmable Internet*, Lecture Notes in Computer Science 1222, S. 25–47. Springer-Verlag, 1997.
4. P.T. Devanbu and S.G. Stubblebine. Stack and queue integrity on hostile platforms. In *IEEE Symposium on Research and Privacy*, S. 198–206. IEEE Press, 1998.
5. G. Karjoth, N. Asokan, and C. Gülcü. Protecting the computation results of free-roaming agents. In *Mobile Agents (MA'98)*, Lecture Notes in Computer Science 1477, S. 195–207. Springer-Verlag, 1998.
6. G. Karjoth and J. Posegga. Mobile agents and Telcos' nightmares. *Annales des Télécommunications*, 55(7/8):29–41, 2000.
7. N.M. Karnik. *Security in Mobile Agent Systems*. PhD thesis, University of Minnesota, 1998.
8. D. Lange and M. Oshima. *Programming and Deploying Java Mobile Agents with Aglets*. Addison Wesley Longman, 1998.
9. F. Mattern. Mobile Agenten. *it+ti – Informationstechnik und Technische Informatik*, (4): 12–17, 1998.
10. G. Vigna (Hsg.). *Mobile Agents and Security*. Lecture Notes in Computer Science 1419. Springer-Verlag, 1998.
11. J. Vitek and C. Jensen (Hsg.). *Secure Internet Programming: Security Issues for Mobile and Distributed Objects*, Lecture Notes in Computer Science 1603. Springer-Verlag, 1999.
12. B. Yee. A sanctuary for mobile agents. In Vitek and Jensen [11], S. 261–273.