Kapitel 1.

Motivation

Server, die über das Internet erreichbar sind und den Mitgliedern einer geschlossenen Benutzergruppe Informationen zur Verfügung stellen, sind sicherheitskritische Komponenten. Einerseits muß unbeteiligten Personen der Zugang zu den Ressourcen gänzlich verwehrt werden, andererseits sind Zugriffe von legitimen Nutzern auf Zulässigkeit hin zu prüfen und gegebenenfalls zu unterbinden. Abbildung 1.1 zeigt das für diese Arbeit ausschlaggebende Szenario: Ein berechtigter Benutzer macht mit Hilfe eines Clientprogramms Fernzugriffe auf Ressourcen, die in einem geschützten Intranet lokalisiert und durch ein Serverprogramm zugänglich sind.

Systembetreiber, beispielsweise kommerzielle Organisationen, wollen die bestimmungsgemäße Nutzung ihrer elektronischen Werte gewahrt wissen. Dazu werden von den Betreibern sogenannte *Schutzziele* verfolgt, die das abstrakte Sicherheitsbedürfnis untergliedern und konkretisieren. Schutzziele unterscheiden sich bezüglich des zu schützenden Gegenstandes und der Akteure. In [116] werden eine Reihe von Schutzzielen und deren Wechselwirkungen diskutiert.

Aus Sicht des Systembetreibers sind die folgenden klassischen Schutzziele von besonderer Bedeutung[1]:

- *Vertraulichkeit*
  Geheimhaltung von Daten

- *Integrität*
  Unverfälschtheit von Daten

- *Verfügbarkeit*
  Zugänglichkeit von Daten, Diensten und der damit verbundenen IT-Infrastruktur

[1] Neben diesen klassichen Schutzzielen sind gegebenenfalls auch Privatheit und Verbindlichkeit von Interesse. Diese werden in dieser Arbeit nicht berücksichtigt.

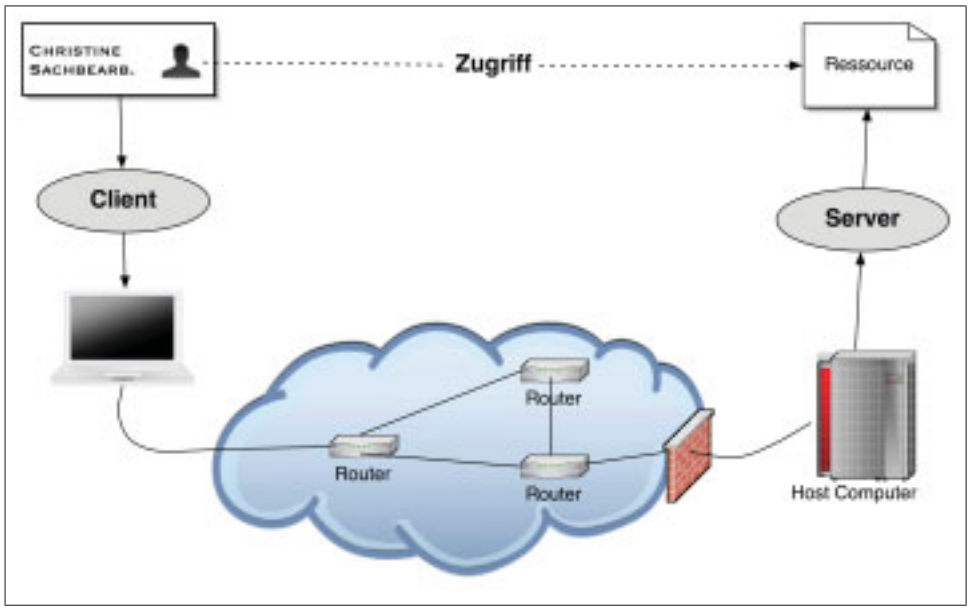

Abbildung 1.1.: Fernzugriff auf Ressourcen, die durch Server über das Internet erreichbar gemacht werden. In dem Bild ist zu sehen, wie ein Benutzer mit einem Clientprogramm, das auf seinem mobilen Rechner läuft, auf ein Dokument zugreift, das sich im geschützten Intranet befindet. Die gestrichelte Linie symbolisiert den logischen Zugriff; die durchgezogenen Linien zeigen den tatsächlichen Kommunikationsfluß.

1.1. Zugriffskontrolle

Um die Schutzziele der Betreiber durchzusetzen, wird Zugriffskontrolle als technische Maßnahme getroffen. Ein System mit Zugriffskontrolle verhindert diejenigen Zugriffe, die Daten unzulässig verändern, löschen oder unberechtigten Personen gegenüber offenbaren würden. Ausgangspunkt dieser Arbeit ist Zugriffskontrolle im oben beschriebenen Szenario des Fernzugriffs. Beim Zugriff und dessen Kontrolle sind die folgenden Begriffe von besonderer Bedeutung:

- *Subjekt*
  Das Subjekt stellt die zugreifende Instanz dar. Diese Arbeit beschränkt sich auf Zugriffe von Personen. Personen agieren im System durch Prozesse oder Netzwerkverbindungen.

- *Ressource*
  Ressourcen[2] sind die schützenswerten Daten und Dienste des Systems, auf die Subjekte potentiell zugreifen können.

- *Kontext*
  Beschreibt die Situation, in der sich das System und das Subjekt befinden. Gerade im vorliegenden Szenario ist die Situation ein wichtiger Bestandteil der Schutzziele des Betreibers. Beispielsweise dürfen in vielen Organisationen vertrauliche Dokumente die Gebäude nicht verlassen. Diese Forderung muß dann auch für elektronische Wege durchgesetzt werden. Dann dürfen beispielsweise Daten nur eingeschränkt eingesehen oder manipuliert werden, sofern sich der Nutzer außerhalb des geschützten Heimatnetzes befindet.

- *Operation*
  Beschreibt die Art des Zugriffs. Es wird unterschieden zwischen universellen Operationen, die auf jede Art von Ressource anwendbar sind, und ressourcenspezifischen Operationen, die durch die Beschaffenheit der jeweiligen Ressource vorgegeben werden.

- *Referenzmonitor*
  Der Referenzmonitor ist zwischen Subjekten und Ressourcen angeordnet und kontrolliert Zugriffe.

Abbildung 1.2 zeigt die obengenannten Einheiten und deren Zusammenspiel.

[2] In der Literatur wird für die zu schützenden Einheiten oft der Begriff *Objekt* benutzt. In dieser Arbeit wird dafür der Begriff *Ressource* verwendet, um Verwechslungen mit der Terminologie der objektorientierten Programmierung zu vermeiden.

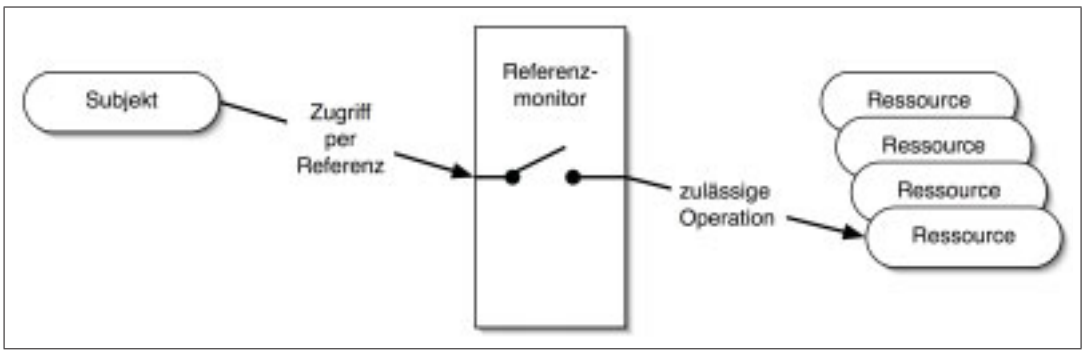

Abbildung 1.2.: Der Referenzmonitor ist den Ressourcen vorgeschaltet und blockt so unzulässige Zugriffe von Subjekten (Benutzern).

1.2. Referenzmonitor

Das Modell des Referenzmonitors[3] geht auf die Arbeit [42] zurück. In diesem Modell können Subjekte ausschließlich über Referenzen auf Ressourcen zugreifen. Zugriffe auf Referenzen werden vom Referenzmonitor kontrolliert, indem Regeln ausgewertet werden, die zulässige Operationen festlegen. Unzulässige Operationen werden vom Referenzmonitor unterbunden. Referenzmonitore finden sich beispielsweise in modernen Betriebssystemen als Teil des Kernels und können rein softwarebasiert sein oder auf Hardwareunterstützung zurückgreifen [87].

Das Referenzmonitormodell läßt sich weiter verfeinern, indem die Rechteprüfung von der Durchsetzung getrennt wird (siehe Abbildung 1.3). Der Policy Decision Point (PDP) ist diejenige Instanz, die bei der Rechteprüfung über die Zulässigkeit eines Zugriffs entscheidet. Der Policy Enforcement Point (PEP) fordert Entscheidungen vom PDP an und setzt diese durch[4]. Diese konzeptionelle Trennung von PDP und PEP kann in einer Client/Server-Architektur umgesetzt werden. Der PDP nimmt dann die Rolle des Servers ein und beantwortet Anfragen von einem oder mehreren PEPs (Clients).

1.3. Anforderungen

Die Einbeziehung von Kontext und die Verschiedenartigkeit der durch Server zugänglich gemachten Ressourcen (und der damit verbundenen Operationen) verlangt nach einem

[3]Der Begriff Referenzmonitor wird in der Literatur sowohl für das abstrakte Modell als auch für konkrete Implementierungen des Modells benutzt. In dieser Arbeit wird mit „Referenzmonitor“ eine Implementierung des Modells bezeichnet. Ist das Modell gemeint, so wird dies durch den Begriff „Referenzmonitormodell“ deutlich gemacht.

[4]In der Literatur tauchen auch die Begriffe Access Control Decision Function (ADF) und Access Control Enforcement Function (AEF) beziehungsweise Access Control Decision Facility (ADF) und Access Control Enforcement Facility (AEF) auf, die äquivalent sind. In dieser Arbeit werden die entsprechenden Einheiten mit PDP/PEP bezeichnet.

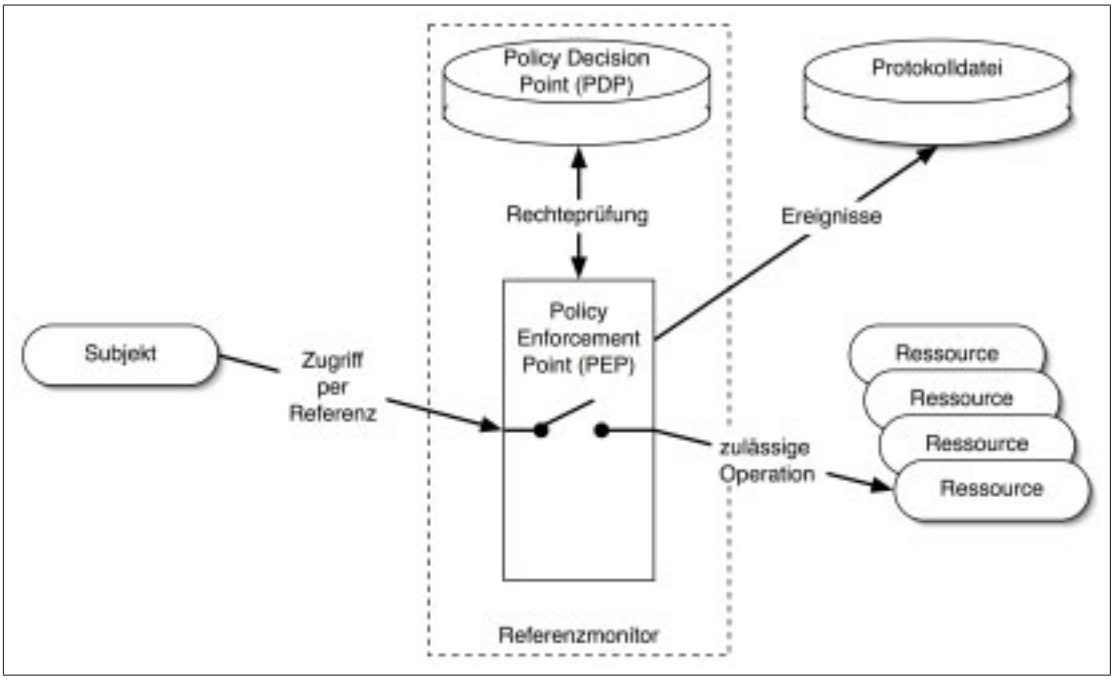

Abbildung 1.3.: Die Komponenten des Referenzmonitors. Entscheidungen werden vom Policy Decision Point (PDP) getroffen. Der Policy Enforcement Point (PEP) sorgt für die Durchsetzung der Entscheidungen. Alle Ereignisse werden in einer Protokolldatei aufgezeichnet.

flexiblen Mechanismus zur Zugriffskontrolle. Um den Anforderungen des Szenarios gerecht zu werden, soll ein Konzept zur Verwendung von Referenzmonitoren in Servern die folgenden Eigenschaften erfüllen:

1. *complete mediation*
   Vollständige Überwachung aller Zugriffe.

2. *isolation*
   Isolation der Referenzmonitore.

3. *separation of concerns*
   Konzeptionelle Trennung von Kernlogik und Zugriffskontrollogik.

4. *no semantic gap*
   Es darf keine Lücke geben zwischen den Rechten, die für Zugriffe vergeben werden können, und den Zugriffen, die Benutzer auf dem Server durchführen können.

5. *obligation*
   Zur Übersetzungszeit werden im Programm explizite Rechteprüfungen erzwungen.

Die beiden ersten Forderungen werden bereits vom ursprünglichen Referenzmonitormodell gestellt[5]. *Complete mediation* bedeutet, daß der Referenzmonitor *jeden* Zugriff von Subjekten auf Ressourcen kontrolliert. Weiterhin muß der Referenzmonitor von den Subjekten isoliert sein, damit er von diesen nicht manipuliert werden kann.

Mit *separation of concerns* wird eine Trennung auf konzeptioneller Ebene gefordert. Dadurch wird eine Vermischung der beiden unterschiedlichen Problemdomänen Kernfunktionalität und Zugriffskontrolle vermieden. Herkömmliche Ansätze führen zu einer Verzahnung der Implementierungen der verschiedenen Problemdomänen im Quelltext und damit – durch erhöhte Kopplung und verringerte Kohäsion der Klassen – zu schlecht handhabbaren Programmen (Wartbarkeit, Fehlerfreiheit, Wiederverwendbarkeit). Die Forderung *separation of concerns* sollte nicht mit *isolation* verwechselt werden. Letztere meint die Trennung zur Laufzeit, die beispielsweise mit getrennten Adreßräumen erreicht werden kann.

Die Forderung *no semantic gap* ist im zugrundeliegenden Szenario besonders wichtig. Denn bei herkömmlichen Ansätzen besteht eine Lücke zwischen den abstrakten Schutzzielen der Betreiber und denjenigen, die sich in konkreten Sicherheitsregelwerken von Serverprogrammen und deren Umgebung formulieren lassen. Diese Lücke wird im folgenden *semantische Lücke* genannt. McGraw stellt dazu in [64] fest:

> „There's plenty of work to be done between the low-level technological things that we understand [...] and the way we express policy. [...] We need to move the level of policy discussion away from things like Java 2 level

[5] Dort wird auch gefordert, daß ein Referenzmonitor korrekt ist. Auf den Nachweis der Korrektheit wird in dieser Arbeit jedoch nicht eingegangen, da dies ein generelles Problem der Informatik ist.

> policy, which is really low-level stuff, or about who called which method on which object, up to something more realistic at a higher level.“

Hier wird kritisiert, daß gegenwärtige Sicherheitsregelwerke eine Granularität aufweisen, die auf einem zu niedrigen Niveau angesiedelt ist, wodurch Erstellung, Validierung und Wartung erschwert werden. Darüber hinaus lassen sich einige Schutzziele gar nicht ohne weiteres formulieren, da die semantische Lücke zu groß ist. So ist zum Beispiel in der Abbildung 1.4 das applikationsspezifische Schutzziel der Vertraulichkeitssicherung von einzelnen e-Mails nicht auf dem Abstraktionsniveau des Dateisystems formulierbar, da für eine Mailbox eine einzige Datei benutzt wird.

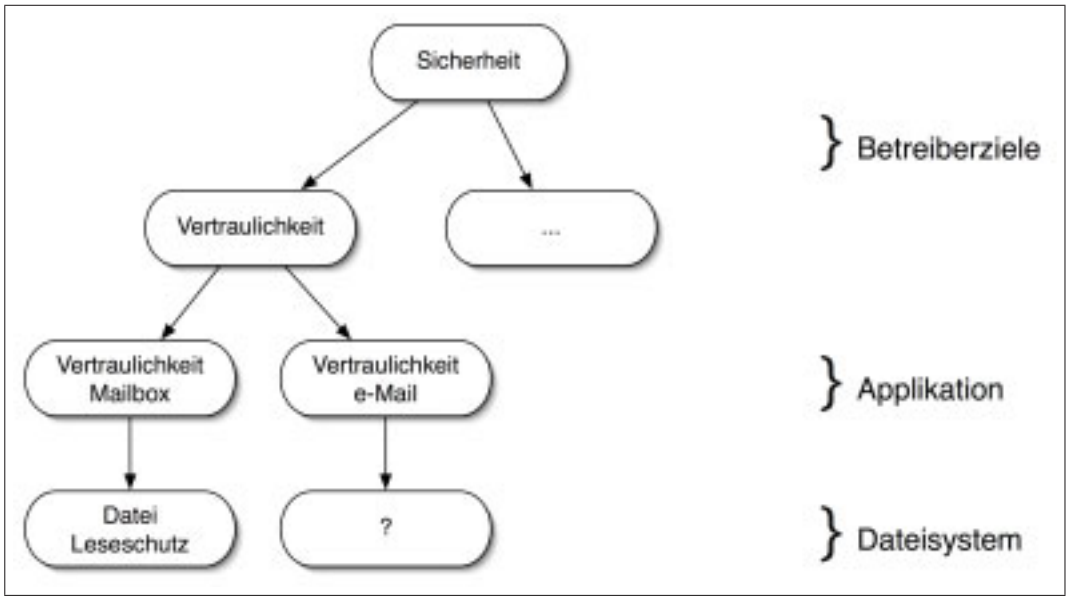

Abbildung 1.4.: Abstraktionsniveaus von Schutzzielen am Beispiel eines Mailservers. Die Schutzziele des Betreibers lassen sich in der Applikationsebene konkretisieren. Auf Ebene des Dateisystems lassen sich die Schutzziele jedoch nicht formulieren.

Um die oben angesprochene semantische Lücke zu schließen, ist es das Ziel, applikationsspezifische Autorisierungsregelwerke auf einem hohen Abstraktionsniveau durchzusetzen. Für ein elektronisches Postfach könnte ein Regelwerk wie folgt aussehen:

```
permit
  user        = 'christine'
  resource    = 'mailBox'
  operation   = 'retrieveMail'
```

In diesem Regelwerk sind die Ressourcen und Operationen auf der Applikationsebene angesiedelt. Der Vorteil von Regelwerken auf hohem Abstraktionsniveau ist die höhere Aussagekraft und die damit verbundene bessere Verständlichkeit und Wartbarkeit.