Institut für Parallele und Verteilte Systeme

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Masterarbeit

Zugriffskontrollmodel für Data Lakes

Nicole Keppler

Studiengang:	Softwaretechnik
Prüfer/in:	PD Dr. rer. nat. habil. Holger Schwarz
Betreuer/in:	Dr. rer. nat. Peter Reimann, M. Sc. Corinna Giebler
Beginn am:	09. Mai 2018
Beendet am:	09. November 2018

Kurzfassung

Zugriffskontrollmodelle werden in Systemen zur Absicherung gegen unberechtigte Systemzugriffe angewandt. Die Verschiedenen Zugriffskontrollmodelle unterscheiden sich vor allem in ihrer Ausdruckskraft, die benötigten Zugriffsregelungen für das System abzubilden. Mit der wachsenden Komplexität und Größe der Systeme wachsen auch die Anforderungen an ein Zugriffskontrollmodell. Während sich für seit Jahren bestehende Speicherkonzepte passende Zugriffskontrollkonzepte etabliert haben, fehlt es neueren Konzepten an einem passenden Zugriffskontrollmodell. Dies ist bei Data Lakes der Fall.

Data Lake ist ein Konzept, dass die Datenvielfalt und Flexibilität im Datenspeicher in den Fokus stellt. Während in anderen Konzepten, wie zum Beispiel dem Data Warehouse, eine Vorverarbeitung der Daten stattfindet, erlaubt das Data Lake eine native Datenspeicherung. Dies bedeutet zwar eine notwendige Anreicherung der Daten in einem Vorverarbeitungsprozess, der Datensatz an sich bleibt jedoch in seinem Rohformat erhalten. Somit können sich in einem Data Lake unterschiedliche Daten mit verschiedenen Datenformaten sammeln.

Diese Arbeit behandelt die Ausarbeitung eines Zugriffskontrollkonzeptes für Data Lakes. Mit der Erklärung des Data Lake Konzeptes und dessen Charakteristika wird zu Beginn ein Grundverständnis geschaffen. Unabhängig vom Data Lake erfolgt eine Aufstellung bestehender Zugriffskontrollmodelle, um einen Überblick der Möglichkeiten für ein Zugriffskontrollmodell für Data Lakes zu schaffen. Zur Auswahl eines Zugriffskontrollmodells für Data Lakes werden aus dem Grundverständnis des Konzeptes und der Vorstellung der bestehenden Zugriffskontrollmodelle Anforderungen herausgearbeitet. Anhand dieser Anforderungen erfolgt eine Evaluation der Modelle. Die Auswertung zeigt, dass durch die Kombination mehrerer bestehender Zugriffskontrollmodelle eine alle Kriterien abdeckende Lösung für Data Lakes gefunden werden kann. Daraus ergibt sich ein hybrider Ansatz basierend auf dem attributbasierten Zugriffskontrollmodell, das mit Elementen aus dem Bell-LaPadula, dem Biba und dem rollenbasierten Zugriffskontrollmodell erweitert wird. Mithilfe eines Anwendungsfall aus dem Bereich E-Health wird eine exemplarische Modellierung und Implementierung des ausgearbeiteten Zugriffskontrollmodells für Data Lakes vorgenommen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Listings

1 Einleitung

James Dixon formulierte im Jahr 2010 in einem Blog seine Idee eines Data Lakes. Über die folgenden Jahre hinweg entwickelte sich aus der Grundidee von Dixon ein vielversprechendes Konzept zur Datenspeicherung vor allem im Big Data Kontext. Obwohl einige Verwendungen des Data Lakes von Dixons ursprünglicher Idee abweichen, verfolgen alle das grundlegende Ziel der Rohdatenspeicherung. Ein Datenverlust durch eine Vorverarbeitung der Daten zur Speicherung soll in Data Lakes verhindert werden. Trotz unterschiedlichen Schwerpunkten bestehen ähnlich dem Data Warehouse Anforderungen an das Data Lake. Das Konzept entwickelt sich stetig weiter um den grundlegenden Anforderungen an Speicherkonzepten gerecht zu werden.
Ein Aspekt dieser Anforderungen ist die Absicherung von Systemen gegenüber Unberechtigten. Sowohl der Schutz der Daten als auch des gesamten Systems stehen dabei im Fokus. Im Zusammenhang mit Datenschutz und Informationssicherheit etablierte sich ein fundamentales Konzept, das kurz CIA-Triade abgekürzt wird [Chi12]. CIA steht für Confidentiality (dt. Vertraulichkeit), Integrity (dt. Integrität) und Availability (dt. Verfügbarkeit). Vertraulichkeit von Informationen bedeutet den Schutz vor Weitergabe an Unbefugte, Integrität den Schutz vor Modifizierung der Daten durch Unbefugte und Availability die Sicherstellung, das Befugte jederzeit auf die Daten zugreifen können. CIA definiert die Aspekte eines Systems, die geschützt werden sollen. Dabei können unterschiedliche Sicherheitskontrollen zur Einhaltung und Sicherstellen der CIA-Eigenschaften eines Systems zum Einsatz kommen. Eine entsprechende Einhaltung der Sicherheitsziele sollte ebenfalls im Data Lake Konzept verankert sein. Manche Aspekte, wie zum Beispiel die Zugriffskontrolle, sind jedoch noch nicht ausreichend herausgearbeitet. Etliche Artikel und Beiträge behandeln das Rechtemanagement und die Zugriffskontrolle für bestehende Datenbanklösungen. Das Konzept des Data Lakes ist noch nicht soweit ausgereift und evaluiert um eine passende Zugriffskontrolllösung zu bestimmen.

In der Literatur findet sich eine logische Architktur, die sich auf einer 3-Tier-Architektur basierend auf drei Funktionsschichten stützt. Die Zugriffskontrolle ist innerhalb der Sicherheitsschicht vorgesehen. Diese Sicherheitsschicht bündelt alle sicherheitsrelevanten Funktionen zur Verwendung durch die anderen Schichten und Stufen [PP15]. Bestehende Umsetzungen eines Data Lakes greifen für die Zugriffskontrolle auf Mechanismen der verwendeten Technologien zurück. Da das Data Lake jedoch ein technologie-unabhängiges Konzept darstellt, darf es nicht an technologische Eigenheiten binden. Mit der Verwendung in bekannten Big Data Einsatzbereichen, z.B. IoT (Internet of Things) und Industrie 4.0, rücken Sicherheitsanforderungen für das Data Lake Konzept immer mehr in den Vordergrund.

Der Bereich E-Health stellt ebenso einen möglichen Anwendungsfall für Data Lakes da. Im Rahmen seiner Digitalisierungspläne möchte der aktuelle Gesundheitsministers Jens Spahn Patientenakten elektronisch (ePA) für Ärzte und Patienten bereitstellen [May18]. Die ePA soll elektronische Fallakten (eFA) oder elektrische Mediaktionspläne (eMP) enthalten und jederzeit von Patienten, Apothekern oder Ärzten eingesehen werden können [Gem18; Wil18]. So scheint der Gesundheitsminister alle analogen Artefakte abschaffen zu wollen und einen digitalen Austausch

zwischen Patienten und Heilberuflern zu fördern. Da Daten für das Gesundheitswesen weit umfangreicher als Patientenakten sind, seien es Forschungsdaten, Pharmadaten oder persönliche Daten des Patienten, fällt eine massive Datenmenge unter die Digitalisierungspläne des Ministers. Eine Idee wäre, dabei auf das Konzept des Data Lake umzustellen und die Datensilos einzelner Parteien zusammenzulegen. Mit einem entsprechenden Zugriffskontrollmodell und Rechtemanagement könnte auch der Datenschutz sichergestellt werden.

Aufgabenstellung Mit der möglichen Speicherung sensibler Daten innerhalb des Data Lakes entstehen Sicherheitsanforderungen zum Schutz der Daten. Sie müssen vor nicht-autorisierten, unerlaubten Zugriffen geschützt werden. Eine Zugriffsregelung ist dafür nötig. Mithilfe von Zugriffskontrollmodellen lassen sich Zugriffsregeln und -beschränkungen definieren und eine Zugriffskontrolle durchführen. Zugreifende Instanzen müssen die Zugriffskontrolle erfolgreich durchlaufen um einen Zugriff durchführen zu dürfen. Innerhalb von Data Lakes wirkt sich eine Zugriffskontrolle auf die Analyse- und Anfrageprozesse aus. Diese Prozesse können zum Beispiel von Problemlösungsprozessen angestoßen werden.
Ziel dieser Arbeit ist Definition eines abstrakten Zugriffskontrollmodell, das speziell für Data Lakes angepasst ist. Dazu sollen verschiedene bestehende Zugriffskontrollmodelle verglichen werden. Mithilfe eines Anwendungsfalls, der sich durch die Ausführung eines Problemlösungsprozesses ergibt, soll eine Modellierung und prototypische Implementierung eine Anwendung des erarbeiteten Zugriffskontrollmodells aufzeigen.

Gliederung Diese Arbeit befasst sich mit der Herangehensweise und Ausarbeitung eines Zugriffskontrollmodelles für Data Lakes. Nach dem Heranführen an das Data Lake Konzept und der Vorstellung der damit einhergehenden Herausforderungen, stehen die unterschiedlichen Zugriffskontrollmodelle herkömmlicher Datenbank- und Systemlösungen im Fokus. Darauf aufbauend werden Kriterien für ein Data Lake Zugriffskontrollmodell herausgearbeitet und ein Lösungskonzept in der Theorie vorgestellt. Der Anwendungsfall aus dem Bereich E-Health befasst sich mit der Problematik von abgekapselten Datensilos und fehlenden Zugriffsmöglichkeiten auf andere Datenspeicher innerhalb eines Problemlösungsprozesses, wie zum Beispiel dem 8D-Prozess. Aus den Ergebnissen des 8D-Prozesses motiviert sich die Modellierung und prototypische Implementierung eines Zugriffskontrollmodells für den Einsatz eines Data Lakes.

2 Verwandte Arbeiten

Zugriffskontrolle ist ein Thema, dass bei jeder Speicherlösung eine Rolle spielt. Im Allgemeinen bauen die unterschiedlichen Zugriffskontrollmechanismen auf zwei Ansätzen, der systembestimmten und der rollenbasierten Zugriffskontrolle auf. Big Data und daraus resultierende Konzepte, wie Data Warhouses und Data Lakes, erfordern eine umfassende, kontextabhängige Zugriffskontrolle mit feingranularen Zugriffsrechten. Ein entsprechendes Zugriffskontrollkonzept für Data Lakes ist in der Literatur nicht auffindbar. Aus diesem Grund umfasst dieses Kapitel verwandte Arbeiten aus dem Umfeld von Data Lakes: dem Big Data Kontext und Data Warehouses.

2.1 Privacy aware access control (PAAC) in Big Data

Umsetzungen von Big Data Lösungen bauen auf den Grundformen der Zugriffskontrollmodelle auf. In der Literatur behandeln etliche Arbeiten eine sogenannte “privacy aware access control“ basierend auf attributbasierter oder rollenbasierter Zugriffskontrolle. [NBL+10] stellt die Einschränkungen konventioneller Zugriffskontrollmodelle im Bezug auf Privacy vor. Es fehle an Zweckbindung, Verpflichtungen und Bedingungen, sowie an dem Einfluss von Privacy-Richtlinien bei der Zugriffskontrolle. In dem Paper ist das rollenbasierte Zugriffskontrollmodell erweitert. [CF15] erläutert ebenfalls die rudimentäre Zugriffskontrolle bei Big Data Lösungen und weist auf fehlenden Schutz der sensiblen Daten hin. Mit der Anpassung von PAAC an die darunterliegenden verwendeten Technologie könne diese Problematik behoben werden. Gängige Technologien, wie Hadoop und MongoDB, nutzen eine rollenbasierten Zugriffskontrolle, auf die PAAC aufbauen könne. [YJR+14] fokussiert sich weniger auf den Schutz sensibler Daten sondern auf eine effiziente Zugriffskontrolle mit anpassbaren Richtlinien anhand einer attributbasierten Zugriffskontrolle.

2.2 Data Warehouse

Die Zugriffskontrolle in Data Warehouses erschwert sich durch das verteilte, multidimensionale Konzept [FTVP06]. Basierend auf der systembestimmten Zugriffskontrolle dient die Klassifikation von Subjekten und Objekten der Erstellung von spezifischen Zugriffskontrollregeln und erweitert die Zugriffskontrolle des Datenmodells. Dahingegen stellt [KQS+98] ein Zugriffskontrollmodell mithilfe von Metadaten vor. Metadaten werden bei der Zugriffsrechtevergabe als Attribute hinzugezogen, sodass es sich um eine attributbasierte Zugriffskontrolle handelt. Sogenannte Zugriffsmetadaten speichern für die Zugriffskontrolle relevante Informationen und ermöglichen eine Unterscheidung verschiedener Rollen. Daraus ergibt sich eine auf Rollen gestützte attributbasierte Rechtevergabe. Die Zugriffsmetadaten eines Subjektes beschreiben die erlaubte Sicht (View) des Subjektes auf das Data Warehouse (siehe Abbildung 2.1).

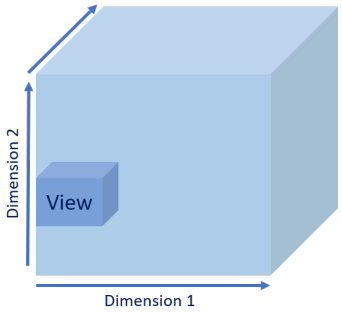


Abbildung 2.1: Datenwürfel eines Data Warehouse mit drei Dimensionen

[RS00] verfolgt ebenfalls den view-basierten Ansatz der Zugriffskontrolle anhand der systembestimmten Vergabe von Zugriffsrechten. Mithilfe von Views erhalten Subjekte einer bestimmten Subjektgruppe Zugriff auf einen beschränkten Teil des Data Warehouses. Ebenso beschreibt [KQS+98] einen hybriden Ansatz aus systembestimmter und rollenbasierter Zugriffskontrolle, wobei zwischen unterschiedlichen Zugriffsbereichen (Views) auf Objekte unterschieden wird.

Gegen die Einschränkung in Views spricht sich [AA10] aus. Der Nachteil der verschiedenen Sichten auf das Data Warehouse bestünde in der fehlenden Navigationsmöglichkeit. Subjekte erhalten Zugriff zu erlaubten Bereichen und hätten keinen Überblick über den gesamten Datenwürfel des Data Warehouse. Im Ansatz von [AA10] sind sich die Subjekte über die Existenz und die Dimensionen des Würfels bewusst, ihr direkter Zugriff beschränkt sich weiterhin auf einen erlaubten Zugriffsbereich. Eine rollenbasierte Zugriffskontrolle legt die Rechte des Subjektes für die verschiedenen Datensichten fest.
Einen gleichen hybriden Ansatz verfolgt [PP00] mit der Kombination von Views und Rollen. Subjekte erhalten einen Überblick über den gesamten Datenwürfel, somit allen Sichten und mithilfe von Rollen Einblicke in verschiedene Ausschnitte des Datenwürfels. Auf diese Weise lassen sich sowohl komplexe Regeln über Rollen, als auch verschiedene Sichten über Zugriffsmetadaten definieren.

2.3 Fazit

Big Data Lösungen enthalten eine rudimentäre Zugriffskontrolle basierend auf den darunter liegenden Technologien. Verschiedene Arbeiten behandeln die Erweiterung der technologiebasierten Zugriffskontrolle mit einem attribut- oder rollenbasierten Ansatz. Die Zugriffskontrolle bei Data Warehouses besteht aus einer Mischung von systembestimmten, rollen- und attributbasierten Zugriffskontrollmodellen. Hybride Zugriffskontrollmodelle erlauben eine für Data Warehouses angepasste Regelung der Rechtevergabe. Zugriffsmetadaten und die Einteilung des Data Warehouses in Zugriffsbereiche (Views) sind wesentliche Bestandteile der beschriebenen Ansätze.

Die vorgestellten Ansätze sind ein Auszug aus der vorliegenden Literatur zu Zugriffskontrollmodellen im Big Data Kontext und bei Data Warehouses. Für die Zugriffskontrolle von Data Lakes zeigen diese Ansätze vor allem, das eine Kombination von mehreren Zugriffskontrollstrategien in

einem hybriden Zugriffskontrollmodell ein Lösungsweg darstellt. Dabei erlaubt eine Vereinigung verschiedener Modelle die Anpassung an die Besonderheiten eines Data Lakes. Zur Ausarbeitung eines Zugriffskontrollmodells schafft das folgende Kapitel einen Überblick über das Data Lake Konzept. Im Anschluss sind die verschiedenen grundlegenden Zugriffskontrollstrategien vorgestellt. Aufbauend auf diesen Inhalten ist das ausgearbeitete Zugriffskontrollmodell für Data Lakes vorgestellt.

3 Data Lake

Big Data ist seit Jahren ein Schlagwort, das im Bereich der Datenspeicherung viel Aufmerksamkeit erhalten hat. Der Begriff umfasst zwei Aspekte. Einerseits beschreibt der Begriff das Vorhandensein großer Datenmengen, die durch neue Anwendungen und Techniken entstehen. Andererseits beschäftigt er sich mit der Problematik, dass mit sehr großen Datenmengen die Kapazität und Leistung von den bisher verwendeten, traditionellen Speicherlösungen und -verarbeitungsanlagen überstiegen wird. Grundcharakteristika von Big Data sind die sogenannten drei V's: Volume, Velocity und Variety [FM16]. Darüber hinaus gibt es noch zwei weitere V's, Value und Veracity. Diese Begriffe umfassen die Grundcharakteristika von Big Data und sind in der Abbildung 3.1 zusammengefasst.

Ein in der Industrie etabliertes Konzept ist das Data Warehouse. Data Warehouses bündeln bisher separierte Datensilos in einer einzelnen zentralen Datenbank. Dabei werden die einzelnen Datensätze aus unterschiedlichen heterogenen Datenquellen zusammengeführt, für Analysezwecke aufbereitet und zur Verarbeitung bereitgestellt. Dies ist der Kernprozess von Data Warehouses und wird als Extract-Transform-Load-Prozess (ETL) (siehe Abbildung 3.2) bezeichnet [BK15]. Für die Speicherung in einem Data Warehouse erfolgt eine Datenvorverarbeitung, um die Daten in das gewünschte Datenformat zu transformieren. Dabei nutzt das Data Warehouse ein Schema-on-Write Ökosystem [Ver18].

Im Gegensatz zum Schema-On-Write-Ansatz des Data Warehouse basiert das Data Lake, auf einem Schema-on-Read Ökosystem und stellt keine festgelegten Strukturanforderungen an den Datensatz [Ver18]. Durch die fehlende einheitliche Struktur der Elemente müssen vorhandene Mechanismen zur Anfrageverarbeitung, Speicherorganisation und Verwaltung angepasst bzw. erstellt werden. Die Zugriffskontrolle ist einer dieser Mechanismen. Um in den folgenden Kapiteln auf

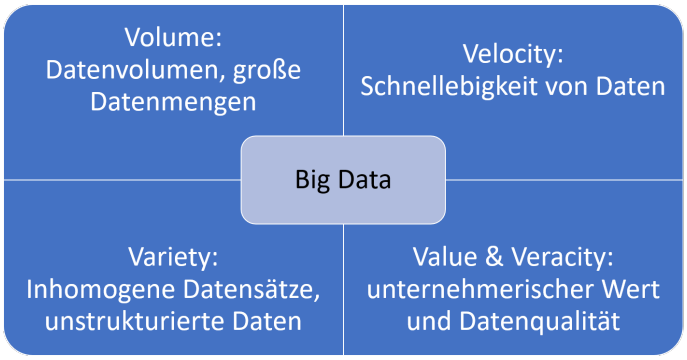


Abbildung 3.1: Big Data Charakteristika [FM16]

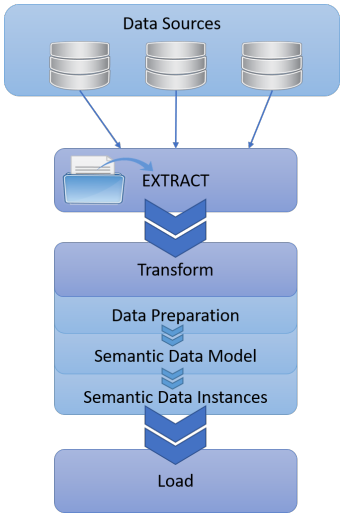


Abbildung 3.2: Extract-Transform-Load-Prozess [BK15]

die Zugriffskontrolle innerhalb von Data Lakes eingehen zu können, stellt dieses Kapitel das Konzept des Data Lakes vor. Nach der Grundidee sind die Basiselemente eines Data Lakes und eine Referenzarchitektur vorgestellt.

3.1 Data Lake Konzept

Das Konzept eines Data Lakes setzt die Idee der Rohdatenspeicherung um. Den Begriff prägte vor allem James Dixon im Jahr 2010. Er veröffentlichte in seinem Blog "Pentaho, Hadoop and Data Lakes" [Dix10] seine Grundidee mit folgenden Worten:

> "If you think of a datamart as a store of bottled water -– cleansed and packaged and structured for easy consumption -– the data lake is a large body of water in a more natural state. The contents of the data lake stream in from a source to fill the lake, and various users of the lake can come to examine, dive in, or take samples."
>
> -– James Dixon [Dix10]

Diese Metapher als ein Vergleich zwischen abgepackten Daten in Flaschen und einem See aus Daten fasst den Grundgedanken zusammen. Dabei geht es vor allem um die Speicherung der Daten in ihrer ursprünglichen Form, dem "more natural state" [Dix10]. Es erfolgt keine Vorverarbeitung der Daten zum Zweck der Anpassung an andere Datenbestände. Dadurch entsteht ein Sammelbecken aus Rohdaten.
Weitere Gesichtspunkte bei der Rohdatenspeicherung sind die Datenvollständigkeit und Reduzierung der Zykluszeit zwischen Datengenerierung und Verfügbarkeit [Ver18]. Durch die fehlende Vorverarbeitung wird die Datenbereitstellung nicht verlangsamt und hat keinen Datenverlust zur Folge. Ein Data Lake erlaubt das Speichern von unterschiedlichen Datenstrukturen und ermöglicht

somit eine Datenvielfalt innerhalb des Speichers.

Das von Dixon vorgestellte Konzept fand viel Zuspruch in der Fachwelt. Viele griffen die Idee und den Begriff auf und erstellten ihre eigenen Definitionen. Zur Festlegung einer Definition für diese Arbeit sind im Folgenden einige verschiedene Definitionen vorgestellt. Als einer der Ersten beschäftigte sich Dan Woods von Forbes mit dem Thema Data Lakes. Er stellte im Jahr 2011 in seinem Artikel "Big Data Requires a Big, New Architecture"[Woo11] das Potenzial von Rohdatenspeicherlösungen sowie Tools zur Umsetzung von Data Lakes vor. Dixons Data Lake Idee entsprang aus einem Kontext innerhalb der Software-Suite von Apache Hadoop. Darüber hinaus zeigt Dan Woods weitere Lösungen, wie zum Beispiel Pervasive's Data Rush, ThingWorx, Splunk, auf. Andere Abhandlungen über Data Lakes folgten. So zum Beispiel Steve Jones von Capgemini [Jon13]. Er vergleicht vor allem die Notwendigkeit eines Data Lake gegenüber Data Warehouses. Dabei schneidet das Data Warhouse als ein im Vergleich zu Data Lake veraltetes Konzept ab. Data Lakes seien im Big Data Umfeld deutlich geeigneter, da die Flexibilität und Anpassbarkeit an neue Gegebenheiten zeitgemäßer seien. Des Weiteren erwähnt Jones, dass eine Data Governance (siehe Abschnitt 3.2.3) maßgeblich für die Organisation und Konsistenz ist.

Zum Thema Data Lake äußerten sich kritische Stimmen. Einige frühe Kritiker sind zum Beispiel Barry Devlin oder Andrew White und Nick Heudecker. Barry Devlin weist in seinem Artikel vor allem auf die Gefahr eines Data Swamps, eine unorganisierte Menge unstrukturierter Daten, hin [Dev14]. Des Weiteren kritisiert er die Fehldeutung eines Data Lakes. Seiner Ansicht nach ist die Assoziation eines Datensees falsch gewählt, da es in der Architektur um die Verknüpfung von Datenspeicher ginge. Er stößt sich vor allem an der Begrifflichkeit als an der Idee.
White und Heudecker von Gartner Inc. (weltweit führende Forschungs- und Beratungsunternehmen) kommentieren 2014 ebenfalls die Verwirrung, die rund um Data Lakes herrsche [Inc14]. Es fehle an einer Übereinkunft zwischen Anbietern, wie ein Data Lake aussieht und welchen Wert es für Kunden darstelle. So sei das Data Lake zu einem Marketing-Hype ausgeartet. Der scheinbare Gewinn durch die Analyse von Daten in ihrer Rohform sei mehr ein Vorwand, diese unverarbeitet abzuspeichern, als wirklich ein Mehrwert für den Kunden darzustellen. Dies läge jedoch nicht an der Idee an sich, sondern an der fehlenden Organisation eines solchen Datenspeichers. Man spare sich die vorhergehende Überlegungen über den Zweck von Daten und werfe sie direkt in den Datensee. Dadurch würde dieser Datensee zu einem unübersichtlichen Datensumpf verkommen. Darüber hinaus seien jegliche Fragen über die Datenqualität und die Datensicherheit im Bezug auf die Zugriffskontrolle offen gelassen worden. Die Definition einer Data Governance, sowie der Einsatz von Datenanalysten für die Verarbeitung und Interpretation der Rohdaten sei unumgänglich, um ein Data Lake realisieren zu können. Hauptkritik ist die fehlende Ausgereiftheit des Konzeptes.

Nach vielen Fehlinterpretationen und -auslegungen des Konzeptes von Data Lakes, veröffentlichte James Dixon einen weiteren Blogeintrag mit der Richtigstellung seiner Begrifflichkeiten [Dix14]. Hier zeigt Dixon seinen Unmut über die ungenaue Bezeichnung von Data Lake. Die korrekte Auslegung sei, dass ein Data Lake die Datensätze aus einer Datenquelle bündele und nicht ein Cross-over aus mehreren Datenquellen und Datensilos darstelle. Devlin, White und Heudecker dahingegen hätten in ihren Artikeln von sogenannten "Water Garden"[Dix14] gesprochen. Dies sei die Datenspeicherung aus mehreren Systemen und Datenquellen in einem Data Lake. So fasst Dixon selbst den Begriff des Data Lake sehr eng, indem er ausschließlich die Rohdaten aus einer bekannten Datenquelle umfasst.
Eine weitere relevante Definition aus der Fachliteratur stellte H.Fang auf. In seinem Paper "Managing

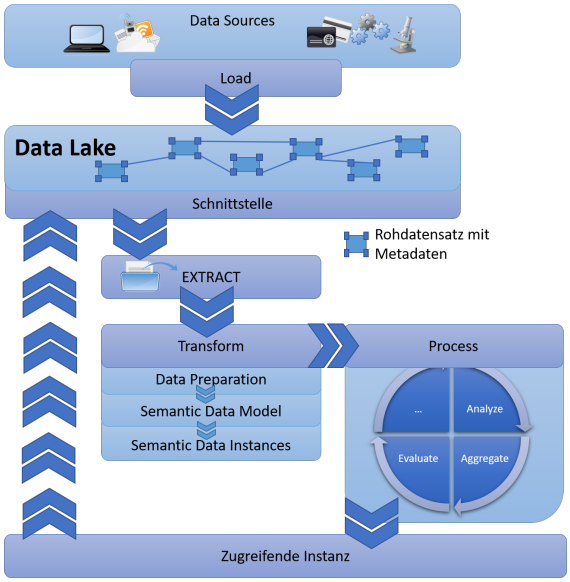


Abbildung 3.3: Data Lake Konzept

data lakes in big data era"[Fan15] fokussiert sich Fang auf die Bedeutung des Sammelbeckens für Rohdaten aus unterschiedlichen Quellen. Für ihn stellt der Begriff Data Lake eine Methodik zur Verwendung nativer Datenformate bei der Erfassung, Archivierung und Analyse dar. Dabei setzt Fang gezielt seinen Schwerpunkt auf Hadoop als Speicherlösung.

In der Literatur finden sich etliche Definitionen von Data Lakes. Folgende Ausarbeitung basiert auf einer kombinierten Begriffsdefinition aus den vorhergegangenen Definitionen. Ein Data Lake ist ein Konzept, das sich mit der Rohdatenspeicherung aus verschiedenen Datenquellen auseinandersetzt. Dabei werden die Grenzen von Datensilos aufgehoben und ein durch Metadaten organisierte, zentrale Datenverwaltung angelegt. In einer Data Governance sind Kriterien zur Qualitätssicherung und Konsistenz von Daten hinterlegt. An dieser Data Governance orientiert sich die Speicherverwaltung. Die Verarbeitung der Datensätze in ein Schema und die Auswertung sowie Kombination erfolgt während des Zugriffes. Aus dieser Definition lässt sich das Schema aus Abbildung 3.3 ableiten.

3.2 Architektur und Strukturierung

Unterschiedliche Aspekte wirken sich auf die Strukturierung und die Architektur eines Data Lakes aus. Folgende Auflistung fasst die wesentlichen Kernpunkte eines Data Lakes zusammen:

- Metadaten: Metadaten beschreiben den Datensatz näher und beinhalten wichtige Informationen über den Ursprung, die Struktur und auch den Inhalt der Daten. Da Metadaten eine zentrale Rolle bei der Speicherorganisation in einem Data Lake einnehmen, behandelt Abschnitt 3.2.4 das Metadatenmanagement genauer.
- Datenzuordnung: Die Datenzuordnung (“integration map “ [Inm16]) beschreibt den Zusammenhang von Daten. Die sogenannte Integration Map ist eine detaillierte Spezifikation, welche Anwendungsdaten aus welchen Datenquellen anhand welcher Charakteristika (vor allem Metadaten) miteinander verknüpft werden.
- Data Lake Kontext: Der Kontext beschreibt, welcher übergeordnete Anwendungsfall dem Data Lake zu Grunde liegt. Die Auswahl der benötigten Datenquellen ist somit zielgerichteter. Auf diese Weise wird der Missbrauch eines Data Lakes als Data Swamp vermieden.
- Datenkontext: Die einzelnen Datensätze verfügen ebenfalls über einen Kontext, um sie für Analysezwecke besser einordnen zu können. Als Kontext für Datensätze kann der Datenursprung, eine Kategorisierung oder ein anderes Kontextmerkmal in den Metadaten herhalten.
- Verarbeitungsprotokollierung: Innerhalb des Data Lakes findet eine Verarbeitung der Rohdaten statt. Dabei werden der Datensatz und auch dessen Metadaten manipuliert (“Metaprocessing “ [Inm16]). Vor allem für Datenanalysten ist sind diese Verarbeitungsdaten besonders interessant um die Nutzung des Data Lakes, des Datensatzes und den Anwendungsfall zu analysieren.

3.2.1 Datenmanagement

Das Datenmanagement dient zur Verwaltung der Daten in einem Datenspeicher. In einem Data Lake lassen sich unterschiedliche Daten verschiedenster Strukturen speichern. Abbildung 3.4 veranschaulicht die unterschiedlichen Datenarten:

- Analoge Daten: Datenquellen für Data Lakes sind unter anderem Maschinen, Geräte oder Sensoren. Diese erzeugen automatisch Daten in einen definierten und somit bekannten Datenformat. Durch die automatische Generierung der Daten fallen diese in sehr großer Zahl an und wiederholen sich in der Regel. Aus diesem Grund werden sie in sogenannten “log tapes “ [Inm16] (Abbildung 3.4) meist in tabellarischer Form gespeichert.

- Anwendungsdaten: Zwar verfügen Anwendungsdaten ebenfalls über eine bekannte Struktur, doch unterscheiden sie sich von analogen Daten maßgeblich durch ihren Datenursprung. Während analoge Daten in der Regel physikalische Messdaten darstellen, entstehen Anwendungsdaten im laufenden Betrieb und während Transaktionen einer Anwendung. Zu den Anwendungsdaten gehören unter anderem übertragene Formulardaten, Systemdaten oder Analysedaten. Als gängige Speicherlösung dieser Daten werden sogenannte “records “ [Inm16] verwendet (siehe Abbildung 3.4). Typisch für records ist ihre einheitliche Struktur. So besteht

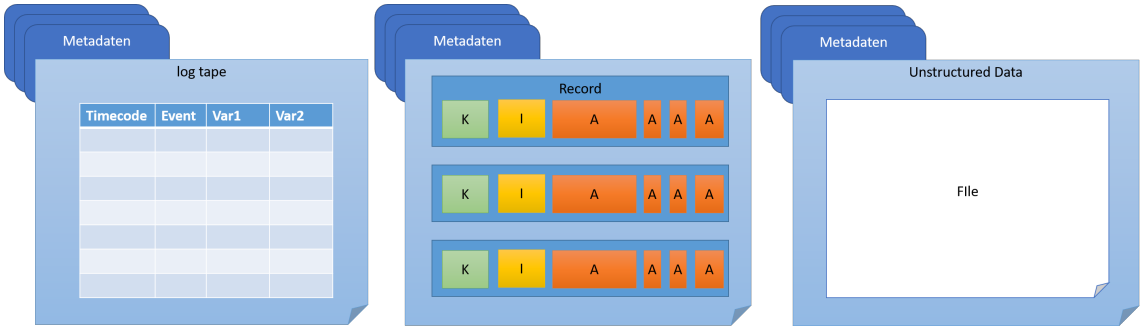


Abbildung 3.4: Datensätze in einem Data Lake

ein record meist aus einem Schlüsselattribut K, einem Indexattribut I und weiteren vordefinierten Attributen A. Je nach Datenursprungs und Datentyp von Anwendungsdaten können die vordefinierten Attribute voneinander abweichen. Diese Struktur der Anwendungsdaten orientiert sich an Datenbankmanagementsystemen (DBMS).

- Textbasierte Daten: Als weitere Kategorisierung gibt es die textbasierten Daten. Diese sind ebenfalls eng mit einer Anwendung verknüpft, werden jedoch als separate Dateien mit Metadaten abgelegt. Für die weitere Verarbeitung muss eine Transformation erfolgen. Der Prozess der Transformation von textbasierten Daten zu analytisch verarbeitbaren Daten heißt "textual disambiguation" [Inm16][Inm14].

3.2.2 Data Wrangling Process

Data Wrangling ist im Big Data Kontext die Bezeichnung für einen Prozess zur Identifizierung, Extraktion, Aufbereitung und Integration von Daten in ein Datenbanksystem [FGL+16]. Am Ende dieses Prozesses sind die Daten von Analyseanwendungen verwendbar und per Zugriffskontrolle vor unberechtigten Zugriffen geschützt. Folgende Aufzählung beschreibt die nacheinander ablaufenden Schritte des Data Wrangling Prozesses für Data Lakes. Abbildung 3.5 veranschaulicht diesen Prozess und referenziert mit Nummern auf die unterschiedlichen Schritte. Der erarbeitete Prozess basiert auf der Zusammenstellung der Inhalte aus den Paper "Data Wrangling:The Challenging Journey from the Wild to the Lake" [TSRC15] und "Data Wrangling for Big Data:Challenges and Opportunities" [FGL+16]. Der Data Wrangling Prozess dient der Aufbereitung der Daten für spätere Zugriffe und liefert Inhalte (Sicherheitsartefakte) für den Einsatz eines Zugriffskontrollmechanismus.

1. Datenselektion: Der allererste Schritt im Data Wrangling Prozess stellt die Datenauswahl dar. Dabei erfolgt eine Selektion und Identifizierung benötigter Datensätze. Dieser Schritt kann erhebliche Auswirkungen auf den Data Lake haben. Auf der einen Seite müssen die Daten selektiert werden um ein Überlauf des Data Lakes an ungenutzten, unbenötigten und wertlosen Daten zu verhindern. Auf der anderen Seite dürfen nicht zu viele Daten herausgefiltert werden, sodass der Mehrwert der Daten verloren geht. So findet bei der Datenauswahl eine Bewertung des Datensatzes nach seinem Wert statt. Besteht ein Mehrwert, so werden danach die Verfügbarkeit und die Konditionen zur Verwendung der Daten und der nachfolgenden Daten dieser Datenquelle überprüft.

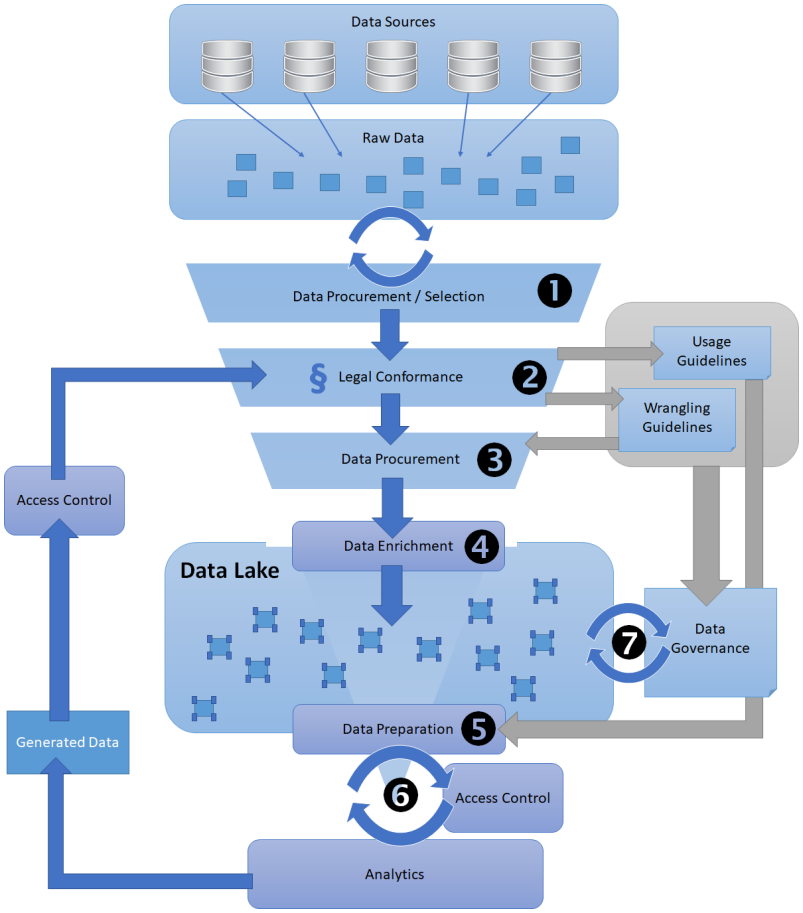


Abbildung 3.5: Data Wrangling Process (teilweise angelehnt an [TSRC15])

2. Datenüberprüfung: Dieser Schritt befasst sich mit den rechtlichen Gegebenheiten. Daten sind ein Gut, das rechtlich vom Eigentümer geschützt werden kann. Durch die Vergabe von Lizenzen bestimmt der Eigentümer Bedingungen die an die Verwendung der Daten geknüpft sind. Liegt eine Lizenz vor, so findet im nächsten Schritt eine Überprüfung der Nutzungskonditionen statt. Diese definieren den erlaubten Nutzungszweck der Daten. Für die weiteren Schritte werden die gewonnen Erkenntnisse aus den Nutzungskonditionen in den sogenannten “usage guidelines “ [TSRC15] dokumentiert. Die Nutzungsrichtlinien werden bei der Zugriffsrechtevergabe hinzugezogen um deren Einhaltung zu garantieren. Darüber hinaus können in den Lizenzen außerdem die Beschaffungskonditionen festgelegt sein. Diese legen die erlaubten Zugriffsformen auf die Daten fest und werden in den “wrangling guidelines “ [TSRC15] dokumentiert.

3. Datenbeschaffung: In dieser Phase des Data Wrangling Prozesses erfolgt die physische Datenübertragung in das Data Lake. In der Regel unterstützen die Datenquellen einen sogenannten “bulk download “ bei dem eine Menge an Daten gebündelt in Dateien von den Datenservern heruntergeladen werden. Entweder triggert der Datenserver den Download oder er wird durch einen Request angestoßen. Für die initiale Füllung des Data Lakes empfiehlt sich eine Datenübertragung über Speichermedien anstatt über das Internet, da so schneller große

Datenmengen übertragen werden können. Der Datendownload über das Internet kann anhand Standardprotokollen, wie FTP und HTTP erfolgen. Doch es gibt auch spezialisierte Protokolle, die für den Dateidownload für Data Lakes interessant sind. So reichner die Protokolle CKAN oder Socrata die Daten mit extra Metadaten an [TSRC15]. So ist der Datensatz schon mit einigen wesentlichen Metadaten angereichert.

4. Datenaufbereitung: Metadaten sind für die Organisation und Strukturierung des Data Lake maßgeblich. Bei der Zugriffskontrolle können Metadaten als Objektattribute dienen und eine Bedingung zur Zugriffsrechtevergabe darstellen. In der Regel verfügt der Rohdatensatz über keinerlei Metadaten, sodass diese im Nachhinein hinzugefügt werden müssen. Die Anreicherung eines Datensatz erfolgt mit unterschiedlichen Metadaten:

    - Schemantische Metadaten geben grundlegende Informationen zur Verarbeitung und der Datenaufnahme. Dafür analysiert der Data Wrangler die Datensätze nach einem vorliegenden Schema, wie zum Beispiel die Spaltennamen in Tabellen. Nicht immer klappt die Schemafeststellung automatisch, sodass ein Datenanalyst manuell eingreift.

    - Eine weitere Art von Metadaten sind semantischen Metadaten. Sie verleihen den Daten eine Bedeutung unabhängig von ihrem vorliegenden Schema. Datenanalysten nutzen diese Informationen zur Einordnung, zum Finden oder zum Filtern von Datensätzen. Eine oft gewählte Art semantische Metadaten umzusetzen sind Kategorien oder die Kennzeichnung zusammengehöriger Daten per Tagging (Stichworten). So entsteht ein Netz aus miteinander verknüpfter Daten über deren Kategoriezuordnungen oder Tags. Daneben enthalten semantische Metadaten in der Regel Informationen über den Autor und den Datenursprung. Außerdem speichert der Data Wrangler anhand von Metadaten Informationen über die Versionierung eines Datensatzes. Mittels diesen Metadaten finden Überprüfungen zur Zuverlässigkeit und zur Datenaktualität statt. Semantische Metadaten können automatisch ermittelt werden. Das Übertragungsprotokoll CKAN liefert einen Teil der semantischen Metadaten bereits mit. Die Ermittlung von Kategorien oder die Zuordnung von Tags ist mittels Standardanalysetools, die Ontologien und Taxonomien der einzelnen Datensätze auswerten und miteinander vernküpfen, möglich.

    - Die letzte Art von Metadaten unterliegt keinen Strukturvorgaben. In den sogenannten Konversationsmetadaten (“conversation metadata “ [TSRC15]) findet ein Austausch zwischen den zugreifenden Instanzen statt. So ist der Grundgedanke dahinter, Informationen die während der Verarbeitung oder Analyse dieser Daten gewonnen werden, für nachfolgende Nutzer zu dokumentieren. Auf diese Weise können erkannte spezielle Eigenschaften eines Datensatzes hinterlegt werden, sodass der nächste Verwender nicht den gleichen Erkenntnisgewinn durchmachen muss. Diese Dokumentation findet in der Regel in Textform statt.

5. Datenpreparation: Zwar sind die Datensätze im Data Lake durch Metadaten aufbereitet, doch der Datensatz erfährt keine Vorverarbeitung. Der Begriff Datenpräparation, auch “data grooming “ [TSRC15] genannt, bezeichnet den Prozess zur Transformation von Rohdaten zu analysierbaren Daten. Da bereits das Schema eines Datensatzes durch das Hinzufügen von schemantischen Metadaten (siehe Schritt 4) vorliegt, muss dies nicht mehr für die Transformation ermittelt werden. Nach der Anpassung eines Datensatzes in ein Format/Schema, erfolgt die Normalisierung. Dabei handelt es sich um den Prozess, gleiche Datenbestandteile zu erkennen und in eine einheitliche Form zu bringen. Als gängiges Beispiel sind hier

unterschiedliche Datumsformate anzuführen. Für den späteren Analyseprozess ist es von Vorteil ein einheitliches Datumsformat bei allen Datensätzen vorzufinden.
Während der Datenpräparation ist es wichtig, jeden Änderungsvorgang am Datensatz zu dokumentieren, damit eventuell ältere Versionen wiederhergestellt oder eine Änderungshistorie eingesehen werden kann. Dies ist vor allem bei der inhaltlichen Bearbeitung relevant, um die Integriät des Datensatzes sicherstellen zu können.

6. Datenbereitstellung: Nach der Datenpräparation ist der Datensatz für die Verwendung vorbereitet. Analysen werden nicht direkt auf dem Data Lake ausgeführt, sondern lediglich auf den relevante Daten. Zur Verwendung der Daten benötigt die zugreifende Instanz entsprechende Zugriffsrechte. Dazu führt der Data Wrangler eine Datenextraktion aus. Trotz allem sollte eine allgemeine Durchsicht und Erkundung der Daten direkt auf dem Data Lake möglich sein, damit Datenanalysten einen Überblick über den Data Lake erhalten.
Den Zugriff auf Daten zuzulassen stellt immer ein Risiko für die Datenintegrität dar. Die zugreifende Instanz muss für den Datenzugriff autorisiert sein. Aus diesem Grund wird in diesem Schritt eine Zugriffskontrolle benötigt. Dabei dürfen die usage guidelines für den Nutzungsgrund nicht missachtet werden. Verschiedene Zugriffskontrollmodelle sind in Kapitel 4 näher beschrieben. Ein ausgearbeitetes Zugriffskontrollmodell angepasst an die Bedürfnisse eines Data Lakes findet sich darunter auch.

7. Datenpflege und Datenerhaltung: Die Datenhaltung im Data Lake muss durch Prozesse geregelt werden. Nicht nur der Inhalt eines Data Lake untersteht einer ständiger Veränderung, auch die eingesetzten Technologien und die Hardware. Aus diesem Grund benötigt es ein Audit, der sich um die Pflege und Wartung des Data Lakes kümmert. Grundlegende Richtlinien und Maßnahmen sind in der “Data Governance “ festgelegt. Die Data Governance und die dazugehörigen Prozesse sind in Kapitel 3.2.3 näher ausgeführt.

Falls bei der Datenanalyse neue Daten anfallen, können diese wieder in Data Lake aufgenommen werden. Dies erfordert jedoch entsprechende Zugriffsrechte. Neue Daten durchlaufen den Data Wrangling Process ab Schritt 2 der Datenüberprüfung. Durch die Anreicherung des Data Lakes durch generierte Daten aus Analysen schließt sich der Data Wrangling Prozess. Die Erklärung des abstrakten Data Wrangling Prozess dient zur Herleitung und Erklärung einer Data Lake Architektur, die in Kapitel 5.1 beschrieben ist.

3.2.3 Data Governance

In einem Data Lake ist das Hauptziel einen Datensumpf zu vermeiden. Dies bedeutet, die Wertigkeit von Daten abzuschätzen und über die Lebenszeit eines Datensatzes zu entscheiden. Vieles hängt dabei von der Datenqualität ab. Die Wertigkeit von Daten korreliert mit deren Qualität und Vernetzung mit dem restlichen Datenbestand.
Unter Datenqualität versteht sich die Korrektheit, Verlässlichkeit und Verwendbarkeit von Daten. Unter Betrachtung, dass in einem Data Lake unbrauchbare Daten nicht versumpfen sollen, entscheidet die Datenqualität über den Lebenszyklus des Datensatzes. Die Datenqualität liegt der Entscheidungsqualität zugrunde und repräsentiert deshalb eine Teil des Datenwertes. Qualitätsmanagementprozesse stellen die Überprüfung von Datensätzen zur Qualitätssicherung dar. Diese ziehen sich durch das ganze System und die ganze Organisation. Somit besteht der Bedarf an Regularien,die die Maßnahmen, Zuständigkeiten, Regeln und Richtlinien innerhalb des Datenqualitätsmanagement

festlegen. Diese sind in den Data Governance definiert. Somit sind alle Aktionen in Bezug auf die Datenqualität konkretisiert, sodass die Qualität des Data Lakes ebenfalls sichergestellt werden kann. Während des Data Wrangling Prozesses regelt das Data Governance die Datenpflege. Es koordiniert alle Prozesse innerhalb des Data Lakes und legt die Verantwortlichkeiten zu diesen Prozessen fest. Darunter fallen zum Beispiel, Wartungsprozesse, Auditprozesse, Entscheidungsprozesse im Datenmanagement und Zugriffsprozesse. Das Data Governance setzt sich unter anderem aus den usage guidelines, den wrangling guidelines sowie aus weiteren allgemeinen Datenqualitätsrichtlinien und Prozessbeschreibungen zusammen. In einem Data Lake liegt dabei der Fokus vor allem auf der Datensicherheit, dem Life Cycle Management, der Datenqualität und der Verwendung von Metadaten.
Darüber hinaus kann das Data Governance durch die Zugriffskontrollrichtlinien erweitert werden. Die festgelegten Policies zur Zugriffskontrolle erweitern den Data Governance Richtlinienkatalog. Dabei enthält die Richtlinie zur Zugriffskontrolle die Einschränkungen aus den usage und wrangling guidelines. Die Umsetzung der Zugriffskontrolle basiert in der Modellierung auf den definierten Richtlinien innerhalb der Data Governance.

3.2.4 Metadatenmanagement

Metadaten dienen in einem Data Lake nicht nur der Anreicherung der Daten mit zusätzlichen Informationen, sondern auch als Sortierungs-, Filterungs- oder Kategoriesierungseigenschaften. Darüber hinaus dienen Metadaten der Systemverwaltung und der Systemadministration. In einem Zugriffskontrollmodell für Data Lakes können die Metadaten eines Objektes zur Evaluation über die Vergabe eines Zugriffsrecht herangezogen werden. Bei Verwendung eines attributbasierten Zugriffskontrollmodells enthalten Metadaten unter anderem die erforderten Objektattribute. Für die Verwendung der Metadaten in einem Zugriffskontrollmodell ist es wichtig, deren Ursprung, ihre Integrität und ihren Informationsgehalt zu kennen. Aus diesem Grund stellt dieser Abschnitt das Metadatenmanagement in einem Data Lake vor.
Wie bereits erwähnt, gibt es unterschiedliche Arten von Metadaten. Die schemantischen Metadaten beinhalten Informationen über das Schema des Datensatzes, die semantischen Metadaten die Bedeutung und de Kontext der Daten und die Konversationsmetadaten spezielle, charakteristische Zusatzinformationen. Auf der einen Seite versucht man Daten mit möglichst viel Informationen anzureichern, um einen größtmöglichen Mehrwert daraus schöpfen zu können. Auf der anderen Seite ist die Anreicherung aller Datensätze mit den gleichen Metadatenfeldern wichtig zur Speicherorganisation. Sobald gleiche Felder vorhanden sind, kann ein Vergleich stattfinden. Äquivalenzen bilden eine Verbindung zwischen Datensätzen. So kann der gleiche Datenursprung, der gleiche Anwendungszweck oder die Einordnung in dieselbe Kategorie eine Verbindung darstellen. Ein gutes Metadatenmanagement ermöglicht die Automatisierung vieler Verwaltungsaufgaben innerhalb des Data Lakes, wie z.B. das Lifecycle-Management, Versionierung oder die Strukturierung.

Das Metadatenmanagement definiert das Metadatenmodell, das den einzelnen Prozessen zu Grunde liegt. Basierend auf dem Metadatenmodell extrahieren die Komponenten die Metadaten und reichern den Datensatz damit an. Damit das Metadatenmanagement automatisiert ablaufen kann, müssen die Prozesse klar definiert sein. Abbildung 3.6 ist eine abstrakte Darstellung über die Prozesse im Metadatenmanagement. Zuerst durchläuft ein Datensatz im Data Wrangling Prozess die Datenaufbereitungsphase. Dazu kommt der Datensatz in die Staging Area des Datenmanagers und wird mit Metadaten angereichert. Der Schema Extractor zieht das Metadatenmodell hinzu und liest

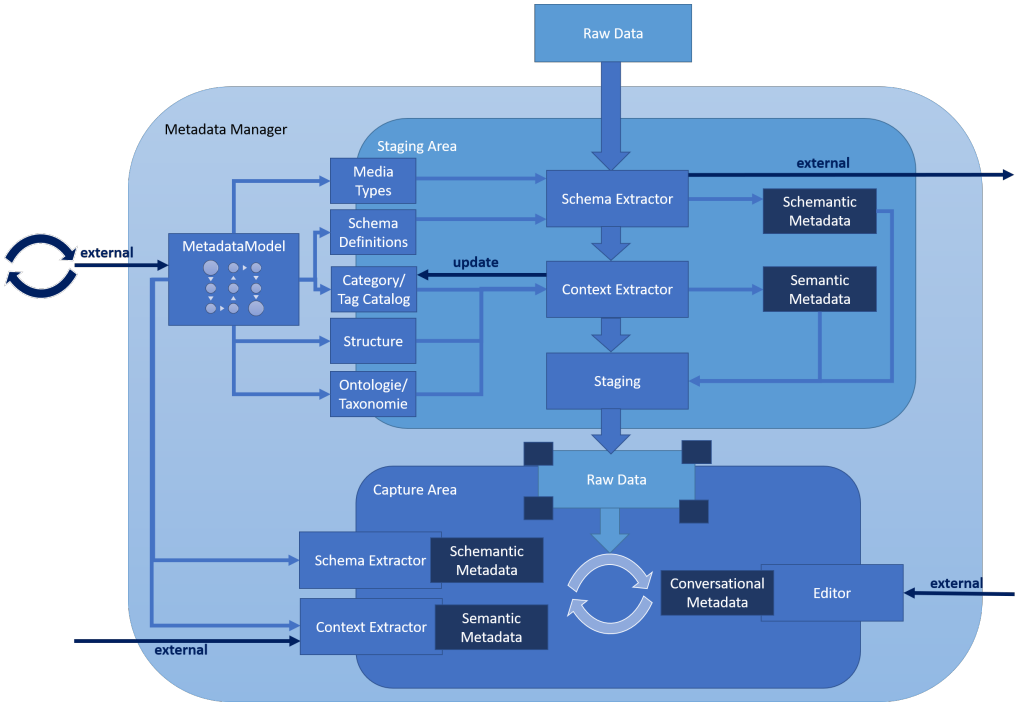


Abbildung 3.6: Metadatenmanagement

dort die bereits vorliegenden Schemadefinitionen und Medientypen aus. Enthält der Datensatz ein bereits vorliegendes Schema, so kann die Referenz auf das Schema in den schemantischen Metadaten hinterlegt werden. Bei einem nicht bekannten Schema kann ein externer Prozess (Automatisiert oder Datenanalyst) informiert werden, um das erkannte Schema dem Metadatenmodell hinzuzufügen. Bei Hinzukommen neuer Datenquellen oder bei Änderungen der vorliegenden Datenquellen muss das dazugehörige Metadatenmodell aktualisiert werden. Im Context Extractor wird der Inhalt des Datensatzes analysiert. Das Metadatenmodell legt eine Struktur für die semantischen Metadaten fest. Dabei sind einige Felder als obligatorisch, andere als optional festgelegt. Obligatorische Felder sind zum Beispiel das Erstellungsdatum oder die Datenquelle. Zu den optionalen Felder gehören dahingegen die Zuordnung zu einer Kategorie. Der angereicherte Datensatz befindet sich nun im Data Lake und in der Capture Area des Metadatenmanagers.
Metadaten ändern sich über den Lebenszyklus eines Datensatzes. Somit steht ein Datensatz im Data Lake unter Beobachtung des Metadatenmanagers. Die Capture Area erfasst alle Änderung am Datensatz und ermöglicht das Hinzufügen oder die Änderung der Metadaten. Dabei entstehen neben automatisch erfassten Metadaten, wie zum Beispiel die Versionshistorie, auch Metadaten aus Externen Quellen. Dies sind vor allem die Conversational Metadaten, die sich aus der Datenanalyse ergeben.

Es gibt verschiedene Frameworks und Tools die sich mit dem Metadatenmanagement in einem Data Lake auseinandersetzen. In der Regel liefert jede verwendete Architektur für Data Lakes ein Lösung für das Metadatenmanagement mit. Das Metadatenmanagement ist und bleibt ein Forschungsgebiet. So befasst sich das Paper “GEMMS: A Generic and Extensible Metadata Management System for Data Lakes“ [QHV16] mit erweiterbaren Metadatenmodellen. Ein anderer

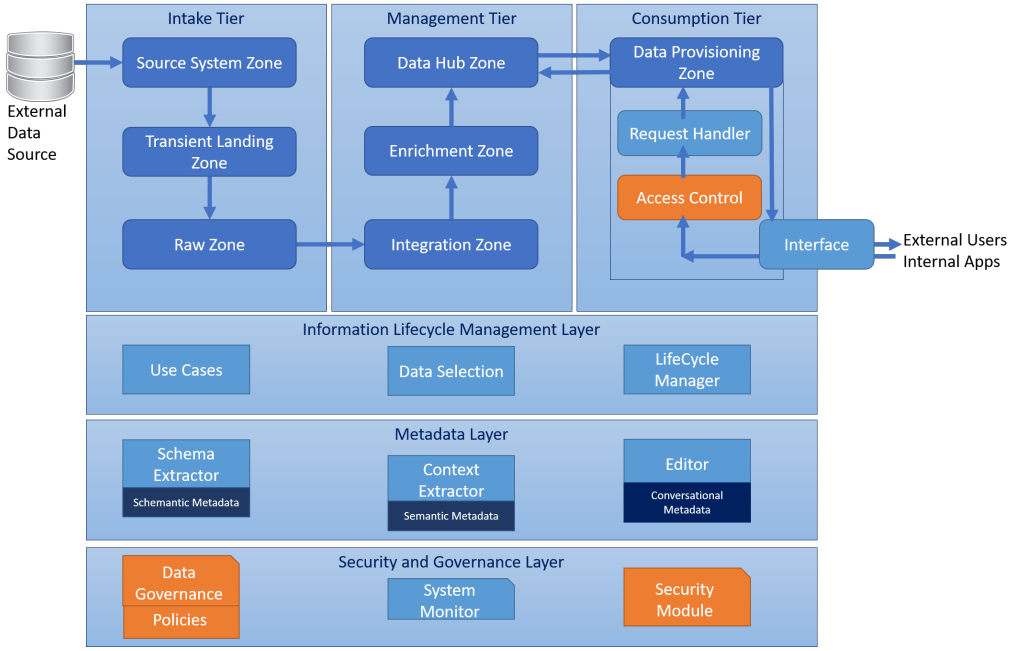


Abbildung 3.7: Referenzarchitektur mit hervorgehobenen Komponenten für die Zugriffskontrolle

Ansatz befasst sich mit der Modellierung eines Metadatenmodells als ein Data Vault [NRD18]. Es liegen bereits auch spezielle Lösungen für das Metadatenmanagement in einem Data Lake vor. So ist “Constance“ [HGQ16] ein Data Lake System, das Metadatenmanagement speziell für Data Lakes umsetzt und nicht auf bereits aus Data Warehouse Systemen bekannte Lösungen zurückgreift. Bei der Zugriffskontrolle innerhalb eines Data Lakes ist es möglich, die Metadaten bei der Rechtevergabe zu verwenden. Schemantische und kontextbezogenen Metadaten basieren auf einem definierten Metadatenmodell. In diesem Metadatenmodell kann beispielsweise die Kategorisierung innerhalb der kontextbezogenen Metadaten über die verfügbaren Zugriffsrechte entscheiden. Auf diese Weise dienen Metadaten als Entscheidungsgrundlagen in Zugriffskontrollmodellen.

3.2.5 Referenzarchitektur

Diese Kapitel beschäftigt sich mit der Vorstellung einer Referenzarchitektur für Data Lakes und dient im Folgenden als Modellmuster an dem sich Umsetzungen von Data Lakes orientieren. Darüber hinaus lässt sich anhand der Referenzarchitektur aufzeigen, an welchen Stellen die Zugriffskontrolle innerhalb eines Data Lakes umgesetzt werden kann.
Abbildung 3.7 zeigt die einzelnen Komponenten der Referenzarchitektur. Die Architektur besteht aus drei Verarbeitungsstufen (engl. tier) und drei Funktionsschichten (engl. layer). Jede Stufe stellt eine Abstraktion einer gemeinsamen Funktionalität dar. Daten durchlaufen zuerst den Intake Tier, danach den Management Tier und zuletzt den Consumption Tier. Ein Layer bündelt dahingegen eine gemeinsame Funktionalität die den Verarbeitungsstufen zu Grunde liegen. In jeder Stufe werden Komponenten oder Funktionalitäten von den darunterliegenden Funktionsschichten benötigt.

Layer

- Security and Data Governance Layer: Diese Schicht ist die Grundlage der gesamten Architektur. Sie beinhaltet alle wichtigen und sicherheitsrelevanten Dokumente und Prozesse. In dieser Schicht ist die Data Governance Policy hinterlegt und auch deren dazugehörigen Überwachungsprozesse. Des Weiteren liegt in dieser Schicht das Sicherheitsmodul. Dieses kümmert sich um die Absicherung des gesamten Systems nach Außen und führt regelmäßige Sicherheitsüberprüfungen durch. Zu den Aufgaben des Sicherheitsmoduls gehört unter anderem die Nutzerauthentifizierung, Quellauthentifizierung, Zugriffskontrolle und die Analyse von Sicherheitsrisiken und Schwachstellen. In regelmäßigen Abständen steht ein Monitoring des Gesamtsystems an, um mögliche Angriffe zu identifizieren und den Schutz des Systems aufrechtzuerhalten. Die in dieser Schicht definierten Sicherheitsmaßnahmen und -richtlinien gelten für alle anderen Komponenten im System. Zusammenfassend stellt diese Schicht die Konsistenz des Data Lakes sicher. Jegliche externe Aktion auf das Data Lake wird durch diese Schicht authorisiert, überwacht und dokumentiert.
- Metadata Layer: In der Metadatenschicht findet das Metadatenmanagement (siehe 3.2.4) statt. Das Metadatenmanagement greift beim Intake Tier bei der Übermittlung der Daten in die Raw Zone ein und reichert die Daten erstmals mit Metadaten an. Im Management Tier agiert die Metadatenschicht weiter als Modul zur Pflege und Organisation des Speichers. Im Consumption Tier dienen Metadaten zur Anfragenverarbeitung. Der Metadatenmanager greift auf die darunterliegende Sicherheitsschicht zu, um nach den Richtlinien der Data Governance zu agieren.
  Die Metadatenschicht durchzieht alle Verarbeitungsstufen. Einerseits dient es der Datenaufbereitung, sodass eine Einordnung des Datensatz in den Speicher erfolgen kann. Andererseits unterstützt es beim Life Cycle Management, der Vernetzung und der Analyse von Datensätzen.
- Information Lifecycle Management Layer (ILM Layer) Der ILM Layer dient vor allem zur Entscheidung des Lebenszyklus eines jedweden Datensatzes. In die Entscheidungsfindung bezieht es die definierten Anwendungsfälle, Metadaten und den Wert des Datensatzes mit ein. Die Bewertung eines Datensatzes basiert auf verschiedenen Kriterien, die von Datenanalysten definiert werden müssen. Diese Schicht führt dann basierend auf dem ermittelten Wert die Datenauswahl für den Data Lake durch. In Ergänzung mit der Metadatenschicht findet eine Klassifizierung und Einordnung der Daten statt. Die Schicht agiert als Schnittstelle zwischen den darunterliegenden Schichten und den darüberliegenden Verarbeitungsstufen. So durchläuft jeder Prozess diese Schicht, sodass der ILM Layer ständige Kontrolle über die Datensätze hat. Diese Schicht agiert oftmals als Monitoring-Prozess.

Die unterschiedlichen Schichten dienen hauptsächlich den organisatorischen Aufgaben innerhalb des Data Lakes. Die Regulierung der Zugriffskontrolle liegt im *Security and Governance Layer* und baut auf die verwendete Datenbank auf. Aus den darüber liegenden Schichten helfen Metadaten und weitere Informationen je nach verwendeten Zugriffskontrollmodells bei der Zugriffsrechtevergabe. Aufbauend auf den Schichten befassen sich die Verarbeitungsstufen (Tier) mit den Datensätzen. Der Data Lake Hauptspeicher befindet sich im Management Tier. Um in diesen zu gelangen, durchlaufen Datensätze zuvor den Intake Tier. Bei einem Datenzugriff über die vorliegende Schnittstelle durchläuft die Anfrage den Consumption Tier und führt im Zuge der Anfrage die Zugriffskontrolle auf dem *Security and Data Governance Layer* aus.

Tier

- Intake Tier: Der Intake Tier beinhaltet mehrere Zonen, die eng mit dem Zustand der Daten verknüpft sind. In der Source System Zone befinden sich alle Mechanismen für die Datenbeschaffung. Die Daten kommen in der Source System Zone an und werden direkt an die Transient Landing Zone transferiert. Die Transient Landing Zone dient als Zwischenspeicher bevor eine Aufbereitung der Daten für die Raw Zone stattfindet. Bei dem Transfer von der Transient Landing Zone zur Raw Zone findet eine Überprüfung der Daten, eine Gültigkeitsprüfung statt. Sobald die Daten in der Raw Zone liegen, ist der erste Schritt des Data Wrangling Prozesses abgeschlossen.
- Management Tier: Im Management Tier findet nun die Aufarbeitung der Daten als Vorbereitung auf die Verarbeitung im Consumption Tier statt. In der Integration Zone gelangen die Daten unverändert in ihrer ursprünglichen Form. Nachdem alle ungültigen Datensätze bereits herausgefiltert sind, finden nun weitere Überprüfungen statt. Der Datensatz befindet sich nun im zweiten Schritt des Data Wrangling Prozesses, der Datenüberprüfung. Die gewonnenen Ergebnisse gelangen in die darunterliegenden Schichten, sodass zum Beispiel die usage und wrangling guidelines für das Data Governance aktualisiert und Metadaten erstellt werden können. In der darauffolgenden Enrichment Zone findet eine weitere Anreicherung der Daten statt. Der dritte und vierte Schritt des Data Wrangling Prozesses laufen parallel ab. Während die Daten weiter angereichert werden, finden die weitere Datenbeschaffung statt. Der Datensatz verfügt nach der Enrichment Zone über alle automatisch erschließbaren Metadaten. Die aufbereiteten Daten sind nun bereit für die Speicherung in der Datenbank des Data Lake. Diese liegt in der Data Hub Zone und ist in der Regel eine Kombination aus mehreren Datenbanklösungen. Ein Data Hub stellt eine Datensammlung aus unterschiedlichsten Quellen dar und dient als Hauptspeicher des Data Lake.
- Consumption Tier: Das Consumption Tier besteht lediglich aus der Data Provisioning Zone, die die Schnittstelle zu zugreifenden Prozessen beinhaltet. Über die Schnittstellen gelangen die Anfragen nach der Überprüfung der Zugriffsrechte zum Request Handler. Dieser verarbeitet die Anfragen und sucht die angefragten Daten aus der Data Hup Zone heraus. Die Datensätze werden über die Schnittstelle an den Anfragenden zurückgeleitet.
  Der Consumption Tier arbeitet eng mit den darunter liegenden Schichten zusammen. Besonders zum Schutz des Data Lakes müssen externe Datenzugriffe ersteinmal authorisiert werden. Dies geschieht über die Zugriffskontrolle, die Bestandteil des Security Modules ist. Mit der Überprüfung der Data Governance muss außerdem festgestellt werden, ob ein Zugriff auf den Datensatz grundsätzlich überhaupt erlaubt ist. Die Anfragenverarbeitung geschieht in Zusammenhang mit der Metadatenschicht. Viele Anfragen basieren auf Metadaten, sodass die entsprechenden Datensätze mithilfe des Metadatenmanagers herausgesucht werden. Des weiteren kann die Metadatenschicht diesen Zugriff dokumentieren und direkt in den Metadaten des Datensatzes vermerken.

Die vorgestellte Referenzarchitektur beinhaltet alle nötigen Komponenten und spiegelt die Schritte des Data Wrangling Prozesses innerhalb der einzelnen Verarbeitungsstufen (Tiers) wieder. Organisatorische Prozesse sind vor allem in den Schichten widergespiegelt. Viele Realisierungen von Data Lakes basieren auf dem Softwareframework von Hadoop (Kapitel 7). Für die Erstellung eines passendes Zugriffskontrollkonzeptes für Data Lakes werden im folgenden Kapitel Zugriffskontrollmodelle näher betrachtet und erklärt.

4 Zugriffskontrolle

Zugriffskontrollmodelle dienen zur Absicherung eines Systems. Themen wie Privacy, Datenschutz und Datensicherheit spielen eine große Rolle bei der Systemarchitektur und sind durch etliche sicherheitskritische Ereignisse in den Fokus von vielen Anwender gerückt. Gerade die neue Datenschutz-Grundverordnung der Europäischen Union sorgte in letzter Zeit für viel Aufsehen und für ein Umdenken in der Datenverarbeitung. Doch diese Änderungen im Bereich Datenschutz bringen nicht viel, wenn das System nicht ausreichend gesichert ist. Ein Aspekt der Absicherung von Systemen ist das Rechtemanagement und die Zugriffskontrolle. Diese Komponenten erlauben einen ordnungsgemäßen Zugriff auf Daten und schützen das System vor unberechtigten Zugriffen. Eng im Zusammenhang mit den Zugriffskontrollmodellen und dem Rechtemanagement stehen die Authentifizierung und Autorisierung von Nutzern. Die Authentifizierung dient zur Identifizierung eines Nutzers, während die Autorisierung die Einräumung von Nutzungsrechten ermöglicht. Ein Zugriffskontrollmodell ist demnach ein Autorisationsmechanismus. In den folgenden Abschnitten ist die Entwicklung der Zugriffskontrolle anhand verschiedener Zugriffskontrollmodelle erklärt. Davor werden jedoch die für dieses Kapitel nötigen Grundbegriffe erläutert. Nach der Vorstellung verschiedener Zugriffskontrollmodelle leitet das Kapitel in einem Fazit zum Data Lake Kontext und dem nächsten Kapitel über.

4.1 Grundbegriffe

Bei der Zugriffskontrolle geht es um den Schutz von Dateien, Systemkomponenten, Informationen, etc. vor unberechtigten Zugriffen. Berechtigten soll über die Zugriffskontrolle der Zugriff gewährt werden. Für die Formalisierung der Konzepte dienen die in Abbildung 4.1 dargestellten Begrifflichkeiten. Bei einem Objekt handelt es sich um die zu schützenden Elemente eines Systems [Eck08, S. 230ff.]. Lediglich innerhalb des Zugriffskontrollmodells modellierte Objekte unterliegen der Zugriffskontrolle. Die für die Objekte gewählte Granularität legt die Feingliederung der Zugriffsrechte fest. Werden mehrere mögliche Objekte im System zu einem Objekt zusammengefasst, so erfolgt die Zugriffskontrolle lediglich auf dem übergeordneten Objekt. Regelungen für die eigentlichen Objekten können so nicht definiert werden. Die auf die Objekte zugreifenden Instanzen sind als Subjekte zu bezeichnen. Subjekte stellen zum Beispiel einzelnen Individuen, Benutzergruppen oder andere Systemprozesse dar.
Innerhalb eines Zugriffskontrollmodells erfragen Subjekte die Zugriffsrechte für ein oder mehrere Objekte. Da ein Subjekt ebenso ein Objekt darstellen kann, sobald auf es zugegriffen wird, gehören Subjekte und Objekte derselben Oberklasse Entität an. Eine Entität ist eine abstrakte Definition und erlaubt die Definition von Attributen und Funktionalitäten, die sowohl Objekte als auch Subjekte betreffen. Subjekte können darüber hinaus in Subjektgruppen zusammengefasst werden. Subjektgruppen bündeln Subjekte mit gleichen Privilegien.
Sowohl Subjekte als auch Subjektgruppen haben Privilegien, die ein Zugriffsrecht oder ein Attribut

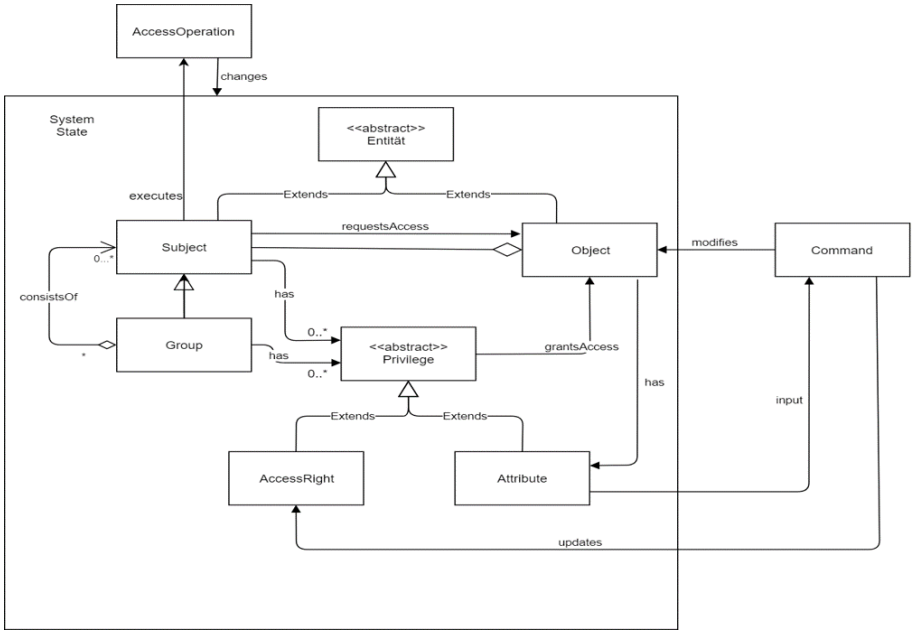


Abbildung 4.1: Begrifflichkeiten (eigene Abbildung, nach [Kep13])

darstellen [BZ12]. Ein Privileg erlaubt dem Subjekt oder der Subjektklasse den Zugriff auf ein Objekt. Die Objekte verfügen ebenso über Attribute.

Entitäten, Objekte, Subjekte, Subjektgruppen, Privilegien, Zugriffsrechte und Attribute bilden zusammen den Systemstatus. Auf dem Systemstatus werden unterschiedliche Operationen von außen durchgeführt. So kann ein Administratorbefehl ein Objekt basierend auf seinen Attributen modifizieren und die Zugriffsrechte anpassen. Die vom Subjekt angefragten Zugriffe erfolgen über eine Zugriffsoperation aus dem Systemstatus heraus. Die Zugriffsoperation ist abhängig von den erteilten Zugriffsrechten und kann zu Änderungen im Systemstatus führen.

4.1.1 Privilegien, Zugriffsrechte und Zugriffsbeschränkungen

Ein Privileg gibt die Möglichkeit, auf ein Objekt zuzugreifen oder bestimmte Aktionen innerhalb des Systems durchzuführen. Die Zugriffskontrolle erfolgt über die Erteilung oder den Entzug von Privilegien an Subjekte. Die Zugriffskontrolle sollte dabei immer dem “Prinzip des geringsten Privilegs“ [BZ12]. Dies bedeutet lediglich ein Minimum an Privilegien den Subjekten zu vergeben, sodass diese gerade ihre Aufgaben durchführen können. Innerhalb des Systems können Privilegien erteilt, verwehrt oder wieder entzogen werden.
Ein Zugriffsrecht ist ein Privileg, das die Zugriffsform auf das Objekt näher beschreibt[SB15]. Zugriffsrechte gefährden die Datenintegrität und sollten aus diesem Grund mit Bedacht vergeben werden. Universelle Zugriffsrechte beschreiben allgemeine Systemoperationen unabhängig von Objekten [Eck08]. Sie sind besonders gefährlich für das System, da sie uneingeschränkten Zugriff bieten. So erlauben systemweite Schreibrechte die Manipulation aller Dateien. Um diese universellen Zugriffsrechte zu vermeiden gibt es die objektspezifischen Zugriffsrechte [Eck08]. Diese sind pro

read	Der Lesezugriff wird lediglich zur Informationsbeschaffung genutzt. Dabei können die Inhalte des Objekts kopiert werden.
write	Der Schreibzugriff erlaubt mehrere Aktionen auf einmal. Neben dem Lesezugriff, dürfen Subjekte Objekte hinzufügen, bestehende manipulieren oder löschen. Der Schreibzugriff ist ein sehr mächtiges Zugriffsrecht und kann zu größeren Systemmanipulationen führen.
execute	Execute erlaubt die Ausführung eines Programmes des Systems. Die Programme können weiter Zugriffsrechte nutzen.
delete	Durch Löschen werden die Objekte vom System entfernt, sodass keine weiteren Zugriffe möglich sind.
create	Mittel create erstellt das Subjekt neue Objekte und ist somit Owner des Objekts.
search	Das Zugriffsrecht Search erlaubt das Durchsuchen des Objekt nach bestimmten Suchkriterien. Auf diese Weise kann die Ordnerstruktur zum Beispiel festgestellt werden.

Tabelle 4.1: Zugriffsrechte

Objekt beschrieben, was eine hohe Granularität des Zugriffsmodells zulässt. Objektspezifische Zugriffsrechte ziehen den funktionalen Kontext und das jeweilige Objekts mit ein. So sind die Zugriffsrechte entsprechend den Aufgaben der Subjekte definierbar. Wohingegen die universellen Zugriffsrechte in klassischen Betriebssystemen integriert sind, benötigt es für die Umsetzung von objektspezifischen Zugriffsrechten meist einen Middleware-Service. Die bekanntesten und weit verbreitetsten Zugriffsrechte listet Tabelle 4.1 auf.

Als Zugriffsbeschränkung bezeichnet man die Anwendung der Zugriffsrechte von einem Subjekt auf ein Objekt. Für das Subjekt sind lediglich die nach den Zugriffsrechten festgelegten Zugriffe erlaubt, somit ist das Subjekt in seinem Zugriff auf das Objekt beschränkt. Diese Zugriffsbeschränkungen nennt man einfach, da lediglich das Zugriffsrecht als Kriterium herangezogen wird [Eck08, S. 232]. Diese einfachen Zugriffsbeschränkungen sind in klassischen Betriebssystemen umgesetzt. Dahingegen gibt es auch komplexe Zugriffsbeschränkungen. Diese bedingen für einen zulässigen Zugriff neben dem Zugriffsrecht eines Subjekts andere Kriterien[Eck08, S. 232]. Als Bedingung für eine komplexe Zugriffsbeschränkung können sowohl systemglobale als auch objektlokale Eigenschaften dienen.

4.1.2 Zugriffskontrolle

Ziel der Zugriffskontrolle ist sowohl der Schutz der Daten vor unberechtigten Lesezugriffen als auch die Verhinderung von unsachgemäßer Modifikation [SV00]. Trotz des Schutzes soll gleichzeitig die Verfügbarkeit der Daten für berechtigte Nutzer gewährleistet sein. Um dies zu erreichen benötigt es einen Kontrollmechanismus, der jeglichen Zugriff auf das System und seine Systemressourcen kontrolliert und autorisiert. Die Zugriffskontrolle unterteilt sich zu diesem Zweck in mehrere Phasen (siehe Abbildung 4.2):

- Autorisierung: In der Autorisierungsphase wird festgelegt, welche Subjekte auf welche Objekte inwieweit zugreifen dürfen.

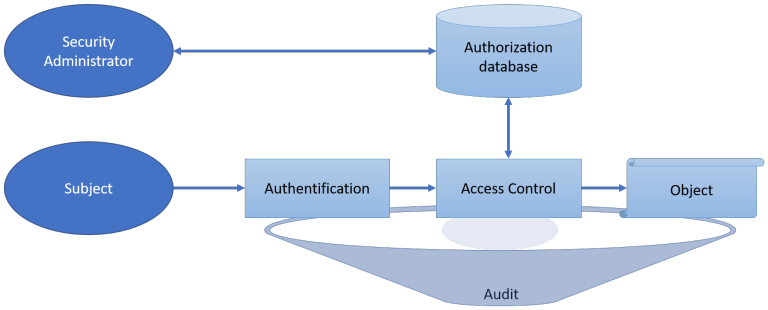


Abbildung 4.2: Zusammenhang der Sicherheitsmechanismen (angelehnt an [SB15, S. 135])

- Authentifizierung: Die Authentifizierung dient der Sicherstellung der Identität des Subjektes. Um sich zu authentifizieren, muss ein Subjekt einen Identifikator vorlegen und mittels eines Geheimnisses (Passwort, o.ä.) einen Nachweis über die Identität bringen [SB15, S.135].
- Zugriffskontrolle: Die Zugriffskontrolle schaut nun die Zugriffsrechte des authentifizierten Subjektes in der Autorisierungsdatenbank nach und erteilt basierend darauf anhand eines Zugriffskontrollmechanismus die Zugriffsrechte.
- Audit: Mit der Durchführung einer Systemprüfung wird die Einhaltung der definierten Sicherheitsstandards überprüft. Dabei ist der ganze Prozess sowie Aktionen unter ständiger Beobachtung.

Den einzelnen Phasen liegen unterschiedliche Konzepte zugrunde. Die Security Policy definiert auf einer abstrakten Ebene die Regeln, anhand deren die Zugriffskontrolle erfolgen soll [SV00]. Dabei sind die Richtlinien in textueller Form festgelegt. Es handelt sich bei der Security Policy mehr um ein Managementdokument, das es im technischen Umfeld zu realisieren gilt. Für die technische Umsetzung gibt es ein Sicherheitsmodel, das eine formale Repräsentation der Security Policy darstellt [SV00]. Das Sicherheitsmodel wird auf unterster Ebene in mehreren Sicherheitsmechanismen unterteilt. Ein Sicherheitsmechanismus beschreibt die zu implementierende Funktionalität für die Zugriffskontrolle. Zur Überprüfung, ob die Richtlinien aus der Security Policy eingehalten werden, findet das Systemaudit statt.

4.2 Zugriffskontrollmodelle

Die Security Policies decken sowohl gesetzliche Regelungen als auch unternehmerisch individuelle Regelungen ab. Die Umsetzung in einem Sicherheitsmodell ist meist sehr komplex. Aus diesem Grund gibt es verschieden Kategorien in die Security Policies eingeordnet werden können. Für jede Kategorie liegen Zugriffskontrollmodelle vor, die als Referenz für das eigene Modell herangezogen werden können. Diese Arbeit umfasst die vier Kategorien von Sicherheitsstrategien:

1. **Discretionary Access Control (DAC):** Die Zugriffskontrolle basiert auf der Identität des Subjekts und der dem Subjekt erteilten Zugriffsrechte [SB15, S. 136]. Diese Zugriffskontrollstrategie wird “discretionary “(dt. Ermessenspielraum) genannt, da ein Subjekt mit entsprechenden Rechten anderen Subjekten in eigenem Ermessen Zugriffsrechte einräumen kann.
2. **Mandatory Access Control (MAC):** Beim MAC basiert die Zugriffskontrolle auf Sicherheitsklassen von Subjekten und Objekten. Die Regelungen legen fest, welche Sicherheitsklassen von Subjekten und Objekten miteinander vereinbar sind. Die Sicherheitsklasse des Subjektes ist also ausschlaggebend für Zugriffe und somit verpflichtend (engl. mandatory).
3. **Role-based Access Control (RBAC):** RBAC nutzt die Einteilung von Subjekten in Rollen. Dabei erhält jede Rolle die Zugriffsrechte zugeteilt.
4. **Attribute-based Access Control (ABAC):** ABAC orientiert sich bei der Zugriffskontrolle an Objektattributen, Subjektattributen und Umgebungsvariablen. Eine Kombination von Attributen kann zur Erteilung oder zum Entzug von Zugriffsrechten führen.

In den folgenden Unterkapiteln werden nun die einzelnen Sicherheitsstrategien samt den Grundelementen der dazugehörigen Zugriffskontrollmodelle vorgestellt. Die *discretionary access control* beinhaltet das Zugriffsmatrixmodell samt möglicher Transformationen zur Implementierung. Das Zugriffsmatrixmodell wird in den folgenden Sicherheitsstrategien MAC und RBAC zur Umsetzung der Modelle verwendet. Als letztes ist das attribut-basierte Zugriffskontrollmodell vorgestellt. Es baut auf der Formulierung von Zugriffskontrollregeln und deren Verarbeitung in einem Zugriffskontrollmechanismus auf. Der letzte Abschnitt beschreibt kurz die unterschiedlichen Ebenen, auf denen ein Zugriffskontrollmodell angewandt wird.

4.2.1 Discretionary Access Control (DAC)

Die benutzerbestimmbare Zugirffskontrollstrategie, DAC genannt, verwendet das Eigentümer-Prinzip [Eck08, S.232]. Beim Eigentümer-Prinzip legt der Eigentümer (owner) die Zugriffsrechte auf das jeweilige Objekt fest. In einem System basierend auf DAC liegen somit nur individuelle Beschränkungen vor. Dies kann zu Inkonsistenzen im System führen. Zusammenhänge von Objekte werden bei der Zugriffsregelung nicht betrachtet, sodass sich Widersprüche in der Rechtevergabe ergeben können. Beispielsweise ist einem Subjekt die Ausführung einer Operation erlaubt, die einen Lesezugriff auf ein anderes Objekt o beinhaltet. Dabei liegt dem Subjekt jedoch explizit eine Verweigerung des Lesezugriffs auf das Objekt o vor.

Zugriffsmatrix-Modell In der Literatur werden verschiedene Modelle für die Zugriffskontrollstrategie DAC vorgestellt. Eines der ersten Modelle stellte Lampson vor [SV00]. Nachdem Graham und Denning das Modell von Lampson aufgriffen, formalisierten Harrison, Ruzzo und Ullmann dieses erstmals als HRU-Modell. Lampson bezeichnete sein ursprüngliches Modell als das Zugriffsmatrixmodell. Dies hat sich als Grundlage für weiteren Modelle durchgesetzt.
Die Matrix repräsentiert die einzelnen Zugriffsregeln, die für jedes Subjekt $S \in S_m$ zu einem Objekt $O \in O_n$ gelten. Subjekte oder Objekte, die nicht in dieser Matrix vorhanden sind, unterliegen keiner Zugriffskontrolle. Tabelle 4.2 zeigt eine beispielhafte Zugriffskontrollmatrix. Die Objekte stellen die Spalten und die Subjekte die Reihen dar. Jedes Feld innerhalb der Matrix, eine sogenannte

Objekte: / Subjekte:	Obj. 1	Obj. 2	Subj. 2
Subj. 1	read	own, write	-
Subj. 2	-	read	-
Subj. 3	write	read	own

Tabelle 4.2: Beispiel einer Zugriffskontrollmatrix

Listing 4.1 Kommandostruktur und Beispielkommando (aus [SV00])

```
1   command c(x_1,...,x_k)
2   if      r_1 in A[x_S1, x_O1] and
3   r_2 in A[x_S2, x_O2] and
4   ...
5   r_m in A[x_Sm, x_Om]
6   then    op_1
7   op_2
8   ...
9   op_n
10  end.
11
12  command ADDWRITE(owner,friend,file)
13  if OWN in A[owner,file]
14  then enter WRITE into A[friend, file]
15  end.
```

“authorization cell “ [Afy06], zeigt die Relation zwischen einem Subjekt S und einem Objekt O auf und definiert die erlaubten Zugriffe A von S auf O (A[S,O]). Das Zugriffsrecht von Subjekt 1 auf Objekt 3 ist in diesem Beispiel der Schreibzugriff ($(A[S_1, O_3] = write)$. Da Subjekte innerhalb eines Zugriffskontrollsystems ebenso Objekte sein können ($S \subseteq O$), müssen Zugriffsrechte auf das Subjekt ebenso in der Matrix definiert werden. In diesem Beispiel ist Subjekt 2 gleichzeitig ein Objekt, dessen Besitzer Subjekt 3 ist.

Eine Veränderung der Zugriffskontrollmatrix erfolgt mittels Kommandos (engl. command), die primitive Operationen abhängig von Bedingungen an einer authorization cell vornehmen [SV00]. Dadurch wird der Status der Matrix verändert. Listing 4.1 zeigt in Zeile 1-10 wie die Kommandostruktur aussieht. So bedingt die Durchführung von Operationen das bestimmte Aktionen r in authorization fields A[S,O] vorliegen. Das in Listing 4.1 vorkommende Beispielkommando Z.12-15 zeigt, wie der Besitzer (owner) eines Objektes (file) einem anderen Subjekt (friend) Schreibzugriff auf das Objekt erteilt. Die Formalisation von HRU definiert sechs primitive Operationen, die den Status der Matrix verändern können. Die Operationen enter oder delete fügen oder löschen ein Zugriffsrecht eines Subjektes auf ein Objekt. Mit den Befehlen create und destroy jeweils für Subjekt und Objekt, können neue Zeilen bzw. Reihen in der Matrix hinzugefügt und gelöscht werden. Da die Matrix ihrerseits ein zu schützendes Objekt darstellt, sind Berechtigungen der Kommandodurchführung zur Zustandsänderung der Matrix als Rechte zu modellieren[Eck08, S. 234]. Eine Matrix deren Zustand nach der Modellierung nicht mehr verändert werden kann, nennt man eine statische Matrix.

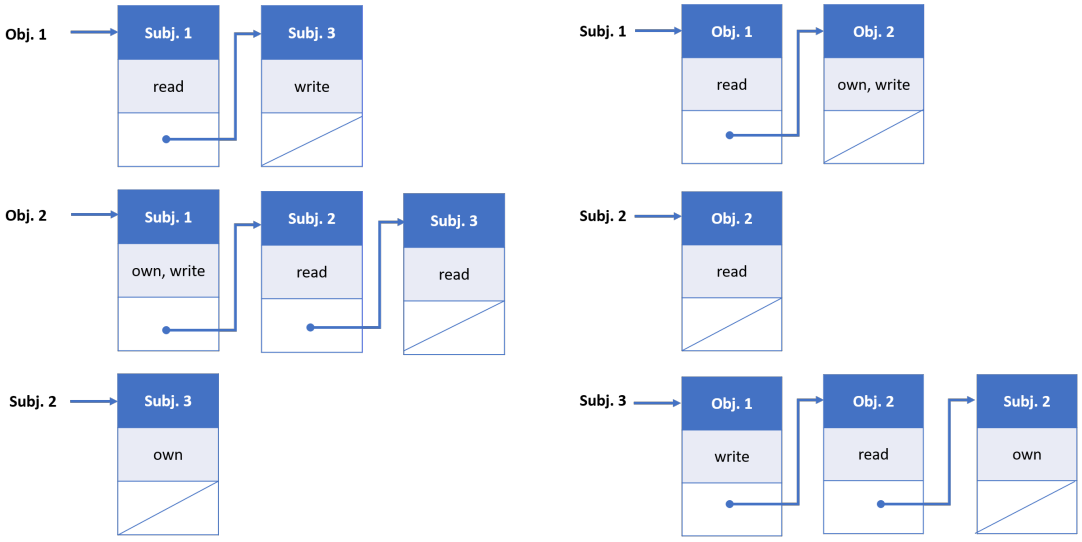
Subjekt	Zugriffsrecht	Objekt
Subj. 1	read	Obj. 1
Subj. 1	own	Obj. 2
Subj. 1	write	Obj. 2
Subj. 2	read	Obj. 2
Subj. 3	write	Obj. 1
Subj. 3	read	Obj. 2
Subj. 3	own	Subj. 2

Tabelle 4.3: Authorization Table am Beispiel von Tabelle 4.2

Obwohl die Zugriffsmatrix ein gutes und übersichtliches Konzept zur Modellierung darstellt, ist sie nicht zur direkten Implementierung geeignet. Für die Umsetzung transformiert man die Matrix in eines der folgenden Formate:

- Authorization Table: Diese Tabelle speichert alle Einträge aus der Matrix in einer Tabelle mit drei Spalten ab. Dabei repräsentiert jedes Tupel A,S,O in der Tabelle die Zugriffsrechte A eines Subjektes S auf ein Objekt O [SV00]. Dieser Ansatz ist in Datenmanagementsystemen umgesetzt.
- Access Control Lists (ACLs): Die Zugriffskontrollisten dröseln die Matrix anhand ihrer Spalten, also den Objekten, auf. Zu jedem Objekt entsteht eine verkettete Liste aus Subjekten und ihrer Zugriffsrechte auf das Objekt. Die Nutzer, die nicht über explizite Rechte auf das Objekt verfügen, können auf einen “default entry “ innerhalb der Liste zurückgreifen [SB15, S. 138]. Dieser beinhaltet die geringstmöglichen Zugriffsrechte, dem Prinzip des geringsten Privilegs folgend.
- Capability List: Die Capability List (capabilit = dt. Befähigung) zerlegt die Matrix anhand der Subjekte und listet pro Subjekt die Objekte samt der Zugriffsrechte darauf auf. Jedes Element innerhalb der Liste wird als “capability ticket “ [SB15, S. 138] bezeichnet und repräsentiert ein Objekt. Ein Subjekt kann sein *capability ticket* von einem Objekt an andere Subjekte weitergeben. Diese Weitergabe der Tickets ist ein großes Sicherheitsrisiko. Aus diesem Grund liegt die Verwaltung der Tickets in der Regel nicht beim Subjekt sondern bei einer höheren Instanz, wie zum Beispiel dem Betriebssystem.

Die Transformation der Zugriffsmatrix in eines der vorgestellten Formate führt zu unterschiedlichen Vorteilen und Nachteilen. Während die *authorization table* wie eine relationale Datenbank durchsucht werden kann, ist jedoch jede Zugriffsrecht als einzelne Zeile abgebildet. Dies lässt die Tabelle schnell wachsen. So führen bereits verschiedene Zugriffsrechte eines Subjekt auf ein bestimmtes Objekt zu mehreren Zeilen. Tabelle 4.3 zeigt die *authorization table* für das Beispiel der Zugriffsmatrix aus Tabelle 4.2, die bereits für das Minimalbeispiel sieben Tabelleneinträge benötigt. Die Aufgliederung der Matrix in eine *access control list* (Abbildung 4.3a) oder eine *capability list* (Abbildung 4.3b) ist dahingegen etwas kompakter. Bei den *ACLs* ist es einfach herauszufinden, welche Zugriffsrechte Subjekte auf ein bestimmtes Objekt haben. Dahingegen bieten die *capability lists* einen Überblick darüber, welche Zugriffsrechte ein Subjekt auf Objekte hat. Je nach Systemmodellierung sollte zwischen den beiden Listen gewählt werden. Zur Verwaltung eignet sich eher die *ACLs*, da diese Objektorientiert sind. Greift nun jedoch ein Subjekt auf ein Objekt zu, so muss die entsprechende Liste des Objektes nach dem Subjekt durchsucht werden.



(a) Access Control List **(b)** Capability List

Abbildung 4.3: ACL und Capability List am Beispiel von Tabelle 4.2

Das Zugriffsmatrix-Modell samt seiner Umsetzung als *authorization table, ACL* oder *capability list* ist das repräsentative Modell für *discretionary access control*. Es lässt sich sehr flexibel einsetzen, da sowohl die Objekt- als auch die Subjektgranularität festlegbar ist. Auf diese Weise lassen sich Zugriffsmatrizen universell für das System und objektspezifisch für einzelne Objekte modellieren. Ein Risiko der Zugriffsmatrizen ist die Gefahr von Inkonsistenzen der Zugriffsrechte. Durch die Festlegung auf objektspezifischer Ebene können zusammenhängende Zugriffe schwer erkannt werden. So bilden sich leicht Widersprüche innerhalb der Matrix. Diese müssen durch Systemanalysen festgestellt und behoben werden. Dynamische Zugriffsmatrizen lassen sich auch nach der Modellierung durch Kommandos verändern. Dies riskiert wiederum eine Systemmanipulation. Die Rechte zur Veränderung der Matrix müssen in einem separaten Zugriffskontrollmodell festgelegt werden. In der Literatur gibt es viele Variationen, die auf das Zugriffsmatrixmodell zurückgreifen. Diese vorzustellen würde den Rahmen der Arbeit sprengen. Für die Modellierung eines Zugriffkontrollmodells für Data Lake reicht das Grundverständnis zu Zugriffsmatrixmodellen aus.

UNIX Permission Model (POSIX) Zugriffsmatrizen bieten eine simple Modellierung von Zugriffsrechten. Die traditionelle Zugriffskontrolle von UNIX baut auf dem Konzept der Zugriffskontrollmatrizen auf. Das UNIX-Sytem weist jedem Nutzer eine eindeutige ID und eine “primary group “ zu [SB15, S. 145]. Darüber hinaus kann der Nutzer weiteren Gruppen, auch Rollen genannt, zugewiesen werden. Die Nutzer stellen im System die Subjekte dar. Bei der Erstellung von Objekten, bekommt das Objekt die Nuzter-ID (*user id*) als Owner, sowie eine Gruppe (*group id*) zugewiesen. Zu jedem Objekt gehören zwölf “protection bits “ [SB15, S. 145], die die Zugriffsrechte auf das Objekt festlegen. Diese *protection bits* gehen auf das POSIX (“Portable Operating System Interface for Unix “ [IEE17]), eine Standardisierung des UNIX Betriebssystems, zurück. So beinhalten die *protection bits* neben den *owner id* und der *group id* drei Berechtigungsblöcke a drei Bits (siehe Abbildung 4.4) [SB15, S. 145][Afy06, S. 39]. So gehört ein Block zum *owner* des Objektes, einer

Abbildung 4.4: Berechtigungsblöcke eines Objektes bei UNIX (eigene Abbildung)

zu der Gruppe des Objektes und einer für die Standardberechtigungen. Jeder Block besteht selbst aus drei Felder rwx. Hierbei steht r (read) für Lesezugriffe, w (write) für Schreibzugriffe und x (execute) für Ausführungszugriffe. Die drei Blöcke repräsentieren die Hierarchie(owner, group, defualt), sodass die Berechtigungen mit der höchsten übereinstimmenden Hierarchiestufe zwischen Subjekt und Objekt angewandt werden. Bei einem Objektzugriff erfolgen zwei Überprüfungsschritte bis das Zugriffsrecht erteilt oder verwehrt wird:

1. Feststellung der höchsten übereinstimmenden Hierarchiestufe
2. Abgleich des angefragten Zugriffes mit den festgelegten Zugriffsrechten der höchsten übereinstimmenden Hierarchiestufe.

Fragt zum Beispiel ein Subjekt, das gleichzeitig der *owner* des Objektes ist, einen Schreibzugriff an, so muss das Bit für den Schreibzugriff innerhalb des *owner* Zugriffsblockes gesetzt sein. Das letzte *protection bit* ist das “sticky bit “[SB15, S.145]. Ist dieses Bit gesetzt, so kann lediglich der *owner* das Objekt umbennen, verschieben oder löschen. Dies ist sehr praktisch für die Rechteverwaltung. Das Berechtigungsmodell von UNIX schließt durch die Verwendung von Nutzer- sowie Objektgruppen bereits Elemente der rollen-basierten Zugriffskontrolle (Kapitel 4.2.3) mit ein.

4.2.2 Mandatory Access Control (MAC)

Im Gegensatz zur benutzerbestimmbaren Zugriffskontrolle baut die systembestimmte Zugriffskontrolle (MAC) auf die von einer zentrale Autorität festgelegten Regeln auf [SV00]. MAC ist eine sehr restriktive Strategie und ist dadurch vor allem in sicherheitskritischen Systemen, wie z.B. beim Militär, eingesetzt [SB15, S.136]. Die zentrale Verwaltung der Zugriffsrechte stärkt die Konsistenz innerhalb des Systems. Außerdem bezieht MAC weitere Variablen in die Rechtevergabe mit ein. So wirken sich beispielsweise systemglobale Eigenschaften auf die benutzerbestimmte Rechtevergabe aus, kann diese sogar widerrufen[Eck08, S. 233].

MAC charakterisiert sich durch “multilevel security “ [SV00]. *Mulitlevel Security* kombiniert hierarchische Sicherheitsklassen mit nicht hierarchischen Sicherheitskategorien und baut daraus eine Zugriffskontrollstrategie auf [Eck08, S. 586]. So ist die MAC-Strategie von zwei Kernprinzipien geprägt: den Secrecy-Based Mandatory Policies, den Integrity-Based Mandatory Policies. Grundlage der Kernprinzipien ist die Sicherheitsklassifizierung.

Sicherheitsklassifizierung Die Sicherheitsklassifizierung erfolgt bei Subjekten sowie Objekten. Die Zuordnung zu Sicherheitsklassen nennt sich “labeling “ [Eck08, S. 586] und entscheidet über die Sensitivität von Objekten sowie der Vertraulichkeitsstufe des Subjektes. Sicherheitsklassen

verfügen über eine partielle Ordnung. Eine partielle Ordnung beschreibt in der Mengentheorie eine Relation zweier Elemente zueinander, die gewissen Regeln folgen. Für die weitere Arbeit definiert $A \geq B$, das A einer höheren Sicherheitsstufe als B angehört und somit B dominiert [SV00]. Eine Sicherheitsklasse stellt entweder ein Sicherheitslevel oder eine Sicherheitskategorie da:

- Classification: Das Sicherheitslevel, auch Classification genannt, beschreibt eine hierarchische Struktur von Sicherheitsklassen. So sind Klassifizierung, wie zum Beispiel Top Secret ≥ Secret ≥ Confidential ≥ Unclassified möglich. [SV00]
- Categorie: Die Sicherheitskategorie, clearance, untersteht dahingegen keiner Ordnung. Sie beschreibt eher die funktionalen Bereiche innerhalb eines Systems [SV00].

Die Zugriffskontrolle basiert auf dem Vergleich der Sicherheitsklassen zwischen Subjekt und Objekt. Verfügt das Subjekt über eine Sicherheitsklasse höher oder gleich der des Objektes, so ist der Zugriff gewährt. Die Verwendung von Sicherheitslevel in Kombination mit Sicherheitskategorien ist je nach Modell unterschiedlich.

Secrecy-based mandatory policy Diese Sicherheitsrichtlinie legt wert auf die Geheimhaltungspflicht von Informationen [SV00]. Lediglich autorisierten Subjekten ist der Zugriff auf Objekte erlaubt. Die Umsetzung der Sicherheitsklassen erfolgen folgendermaßen:
Die Einordnung des Objektes erfolgt anhand der Geheimhaltungsstufe [SS94]. Das Subjekt erhält eine Einordnung in eine Sicherheitskategorie (*clearance*) je nach Vertraulichkeitsstufe. Die Sicherheitskategorien werden zur feineren Unterscheidung von Subjekten und Objekten genutzt.

Die Zugriffskontrolle basiert auf der Sicherheitsklasse des Subjektes und des angefragten Objektes. Für die Feststellung der Zugriffsregeln überprüft der Zugriffskontrollmechanismus die aufgestellten Regeln. Diese Regeln stellen verschieden Sicherheitsklassen in Beziehung zueinander und definieren die Zugriffsrechte für jede Relation. Zwei Allgemeine Regeln sind “read down “ und “write up“ [SS94]. Beim *read down* ist ein Lesezugriff erlaubt, solange das Sicherheitsslevel des Subjektes höher oder gleich der Sicherheitsklasse des Objektes ist. Für das *write up* liegt das Sicherheitslevel des Subjektes unterhalb der Sicherheitsklasse des Objektes. Auf diese Weise können Informationen von Subjekten aus niedrigeren Sicherheitsstufen an Subjekte höherer Sicherheitsstufe übermittelt werden.
Um alle Möglichkeiten des Modells auszusetzen muss ein weiterer Gesichtspunkt betrachtet werden. Ein Nutzer verfügt über ein zugeordnetes Sicherheitslevel. Dabei ist es dem Nutzer überlassen sich als Subjekt gleichem oder niedrigerem Sicherheitslevel auf dem System anzumelden. Sich in einem niedrigerem Sicherheitslevel anzumelden ist nötig um die *write up* Regel zu umgehen. Einem Subjekt ist demnach nur der Schreibzugriff auf Objekte gleicher oder höherer Sicherheitsklasse erlaubt. Um Informationen an niedrigere Sicherheitsklassen weiterzuleiten muss das Subjekt sich somit in der niedrigeren Sicherheitsstufe anmelden können. Die Umständlichkeit der *write up* Regel erklärt sich durch das Vertrauen, das zwar Nutzern, jedoch nicht Subjekten entgegengebracht wird. Subjekte können durch Schadsoftware infiziert sein. Durch die *write up* Regel wird verhindert, dass sensible Informationen an niedrigere Sicherheitsklassen weitergegeben werden.[SS94]

Bell-LaPadula Modell Ein sehr bekanntes Modell als Beispiel für die Modellierung einer *secrecy-based mandatory policy* ist das Bell-LaPadula (BLP) Modell. Es wurde in den 70er Jahren von D.E. Bell und L.J. LaPadula als erstes vollständig formalisiertes Sicherheitsmodell entwickelt

[Eck08, S.255] [SB15, S.461]. Das BLP erweitert das Zugriffsmatrix-Modell Abschnitt 4.2.1 um Sicherheitsklassen für Subjekte und Objekte. Als Zugriffsrechte stehen fünf Aktionen zur Verfügung [Eck08, S.255]:

- read-only: reiner Lesezugriff
- append: Erweiterung eines vorhandenen Objektes (Daten hinzufügen)
- execute: Ausführung
- read-write: Lese- und Schreibzugriff
- control: Rechteweitergabe oder -rücknahme

So baut sich das Modell aus einer Menge von Subjekten S, Objekte O, Aktionen A und Sicherheitsklassen SC zusammen [Eck08, S.256]. Eine Sicherheitsklasse $SC(L, K) \in SC$ besteht aus dem Sicherheitslevel L und der Sicherheitskategorie K. Die Sicherheitsklassen $SC_s \in SC$ für Subjekte s werden *clearance* genannt. Dahingegen bezeichnet man die Sicherheitsklassen $SC_o \in SC$ für Objekte o als *classification*. Ein Subjekt darf jede *clearance* $SC_s(akt.)$ einnehmen, solange $SC_s(akt.) \leq SC_s \in SC$ gilt.[Eck08, S.255f.]
Der Vergleich von Sicherheitsklassen geschieht über die partielle Ordnung $(SC, \leq)$. Für $SC_x(L_x, K_x), SC_y(L_y, K_y) \in SC$ folgt:

$$SC_x(L_x, K_x) \leq SC_y(L_y, K_y) \Longleftrightarrow L_x \leq L_y \wedge K_x \subseteq K_y$$

Mithilfe dieser Relation können bei jeden Objektzugriff die Sicherheitsklassen von Objekten und Subjekten miteinander verglichen werden.

BLP modelliert die Regeln der *secrecy-based mandatory policy* zusammen mit der Zugriffsmatrix M_t für Subjekt s und Objekt o für die Erteilung eines Zugriffsrechts r folgendermaßen:

- Die *Simple-Security-Regel* setzt die *read down* Regel um und erlaubt keine Lesezugriffe nach oben. Dabei ist $r \in M_t(s, o) \wedge SC_o(L_o, K_o) \leq SC_s(L_s, K_s)$.
- Mithilfe der **-Eigenschaft* ist die *write up* Regel realisiert. Auf Objekte niedrigerer Sicherheitsstufe ist kein Schreibzugriff erlaubt, d.h. $r = append \in M_t(s, o) \wedge SC_s(L_s, K_s) \leq SC_o(L_o, K_o)$. Während ein append-Zugriff auch auf Objekte höherer Sicherheitsklassen angewandt werden darf, ist ein Lese-Schreib-Zugriff nur für Klassen gleicher Sicherheitsklasse erlaubt: $r = read - write \in M_t(s, o) \wedge SC_s(L_s, K_s) = SC_o(L_o, K_o)$

[SB15, S. 461ff.][Eck08, S. 255ff.]

Die Regeln des BLP überprüfen zuerst, ob das angefragte Zugriffsrecht in der *authorization cell* $M_t(s, o)$ der Zugriffsmatrix M_t liegt. Danach werden erst die eigentlichen zusätzlichen Regeln angewandt.Damit ist die Zugriffsmatrix aus DAC in ein MAC Modell integriert. Das BLP erweitert das Zugriffsmatrix-Modell um weitere Modellierungsregeln. Dabei sind jedoch die Anwendungsfälle auf die definierten Regeln beschränkt. So können nicht alle Sicherheitsanforderungen, zum Beispiel eine Kombination von Aktionen mit unterschiedlich benötigten Sicherheitsklassen, in BLP modelliert werden. Ein weiterer Nachteil im BLP Modell ist das blinde Schreiben[Eck08, S.261]. Dabei ist es dem Subjekt erlaubt ein bestehendes Objekt höherer Sicherheitsklasse zu erweitern, jedoch nicht seine Änderungen zu lesen. So können Objekte unabsichtlich fehlerhaft manipuliert und nicht mehr korrigiert werden.

Das BLP-Modell ist ein sehr restriktives Beispiel für MAC. Das Modell definiert Zugriffsrechte als universelle Rechte und erweitert die Zugriffsmatrix von DAC um systembestimmte Regeln. Die Implementierung ist einfach umzusetzen, da die Regeln einfach überprüfbar sind. Das Modell verfolgt ganz klar den Ansatz der Geheimhaltung. Jedoch vernachlässigt es die Datenintegrität, da zum Beispiel ein Schreiben nach oben erlaubt ist. Einen Fokus auf die Datenintegrität legt das Biba-Modell (siehe unten).

Integrity-Based Mandatory Policy MAC kann neben der Vertraulichkeit ebenso auf den Schutz der Datenintegrität ausgelegt sein. Dabei verfügt jedes Objekt über eine Integritätslevel von *Crucial (C)* über *Important (I)* zu *Unknown (U)*[SS94]. Das Integritätslevel eines Objektes gibt Auskunft darüber, inwieweit den enthaltenden Informationen vertraut werden kann und ob eine mögliche nicht-erlaubte Manipulation vorliegen könnte. Ein Integritätslevel des Subjektes spiegelt dessen Vertrauenswürdigkeit wider. Mithilfe der Integritätslevel von Subjekt und Objekt kann die Einhaltung folgender Regeln zum Erhalt der Datenintegrität genutzt werden. Die *read up*[SS94] Regel definiert, dass lediglich Subjekte niedrigerer Integritätsebene lesenden Zugriff auf das Objekt höherer Integritätsebene erhalten. Dadurch erhält das Subjekt keine möglicherweise manipulierten Objekte und verteilt diese nicht weiter. Ein Schreibzugriff auf niedrigerer Integritätsstufen ist durch Subjekte höherer Stufe durch die *write down*[SS94] Regel erlaubt. Der Informationsfluss bei der Verwendung von Integritätsklassen verläuft von oben nach unten, was dem genauen Gegenteil von der Verwendung von Sicherheitsklassen in *secrecy-based*-Ansatz entspricht[Eck08, S. 262].

Biba Modell Entsprechend dem Bell-LaPadula Modell teilt das Biba Modell ebenfalls in Subjekte und Objekte ein. Dabei bekommt jede Entität eine Integritätsklasse zugeordnet. Diese Integritätsklassen dienen dazu, lediglich eine kontrollierte Modifikation von Objekten unabhängig von Sicherheitsklassen zu garantieren[SB15, S. 471]. Die Integritätsklassen sind in einer strikten Hierarchie angeordnet und lassen sich somit vergleichen.

Das Biba Modell definiert vier unterschiedliche Zugriffsrechte. *Modify* entspricht dem aus BLP bekannten *append* und erlaubt die Veränderung eines bereits bestehenden Objektes. Ebenso entspricht *observe* dem *read-only*. Das *execute* Zugriffsrecht ist bei beiden Modellen gleich. Darüber hinaus ermöglicht *invoke* eine Kommunikation zwischen Subjekten. Aufbauend auf diesen Zugriffsaktionen ergeben sich folgende Regeln bei vorliegender Integritätsklasse I_s des Subjektes und I_o des Objektes:

- Die *Simple-Integrity-Regel* realisiert das *write down*, d.h. es gilt $I_s \geq I_o$.
- Bei der *Integrity- Confinement-Regel* handelt es sich um die Regulierung des *read up*, daraus folgt $I_s \leq I_o$.
- Für den Aufruf eines anderen Subjektes S_2 durch das Subjekt S_1 spezifiziert das Biba Modell eine separates Zugriffsrecht. Dieses muss ebenfalls durch eine Zugriffsregel abgedeckt sein. Mit der *Invocation-Eigenschaft* ist festgelegt, dass $I_{S_1} \geq I_{S_2}$ ist.

[SB15, S. 471]

Als Erweiterung der Integritätsklassen können ebenfalls Kategorien eingesetzt werden. Diese werden ähnlich dem Bell-LaPadula Modell gehalten und den bestehenden Regeln hinzugefügt.

Dazu analog zum Vergleich der Sicherheitsklassen die Integritätsklasse als eine partielle Ordnung behandelt, wobei die Kategorien einer niedrigeren eine Teilmenge der Kategorien der höheren Integritätsklasse darstellt.
Beide Modelle sind sehr restriktiv. Zwar ist die Datenintegrität durch den kontrollierten Informationsfluss nach unten besser geschützt, doch kommt nun die Geheimhaltung der Objekte zu kurz.

Weitere Modelle Sowohl Bell-LaPadula als auch Biba haben ihre Vor- und Nachteile. Sie lassen lediglich eine beschränkte Ausdrucksfähigkeit zu, sodass nicht alle Systemaspekte modelliert werden können. Aus diesem Grund gibt es viele Erweiterungen und Kombinationen beider Modelle mit anderen Ansätzen.

Das *Clark-Wilson Integrity Model* fokussiert sich auf die Durchführung von Aktionen [SB15, S. 473] . So erfragen Subjekte bei einem anderen Subjekt die Erlaubnis einer Aktion. Das Subjekt, das die Erlaubnis erteilt, darf dabei die Aktion nicht selbst ausführen. Auf diese Weise findet vor der Durchführung von Aktionen eine Überprüfung in Bezug auf die Beeinflussung der Datenintegrität statt. Das Modell unterscheidet zwischen geprüften und ungeprüften Daten. Dabei stehen die geprüften Daten unter besonderem Schutz. Prozeduren überprüfen die Veränderungen der geschützten Daten und gleichen sie mit festgelegten Regeln ab.

Ein weiteren kombinierten Ansatz aus DAC und MAC verfolgt das *Chinese Wall Model*. Dieses Modell verfolgt die Umsetzung von Vertraulichkeit sowie von Integrität und orientiert sich am Umgang mit Interessenskonflikten aus der Finanzwelt [SB15, S.475ff.] . Um Interessenskonflikte feststellen zu können, gibt es in diesem Modell drei Arten von Objekten. Das *Objekt* an sich gehört einer bestimmten Objektgesellschaft an. Das *DataSet (DS)* bündelt alle Objekte einer Objektgesellschaft. Darüber hinaus fügt die *Conflict of interest class (CI)* alle konkurrierenden DS zusammen. Jegliche Lese- und Schreibzugriffe werden über zwei Regelungen realisiert: Mit der *Simple-Security-Regel* darf ein Subjekt S lediglich ein Objekt O lesen, wenn S bereits ein anderes Objekt dergleichen DS von O genutzt hat. Eine weitere Bedingung ist, dass O zu einer DI gehört auf die S noch gar nicht zugegriffen hat. Die zweite Regel (**-Eigenschaft*) erlaubt Schreibzugriffe sobald die *Simple-Security-Regel* für S und O zutrifft und alle Objekte die S lesen darf in derselben DS von O liegen. Ein Schreibzugriff von einem Subjekt ist also auf ein Dataset beschränkt. Lesezugriffe sind solange erlaubt, wie sie nicht auf konkurrierende Objekte zugreifen.

Das *Chinese Wall Modell* zeigt einen völlig anderen Ansatz unabhängig von vordefinierten Sicherheitsklassen auf. Dabei orientiert sich die Zugriffsregelung an bereits getätigten Zugriffen, baut also auf die Historie auf. Das *Clark-Wilson Integrity Model* dahingegen greift auf eine Aufgabentrennung zurück. Diese Aufgabentrennung erlaubt eine Überprüfung der Auswirkungen auf das System bevor die Aktion durchgeführt wird.

4.2.3 Role-based Access Control (RBAC)

Mit der Einführung von Sicherheitsklassen im Bell-LaPadula Modell zeigte sich schon eine Gruppierung von Subjekten zur Modellierungen von Zugriffsrechten für diese Gruppe. Diesem Prinzip folgt ebenso die rollenbasierten Zugriffskontrolle (RBAC).

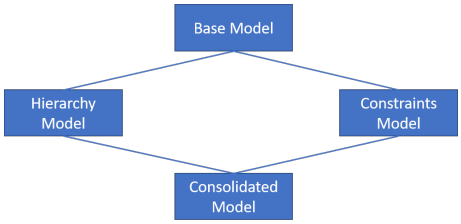


Abbildung 4.5: Zusammenhang der RBAC Modelle (eigene Abbildung, nach [SB15, S. 151]))

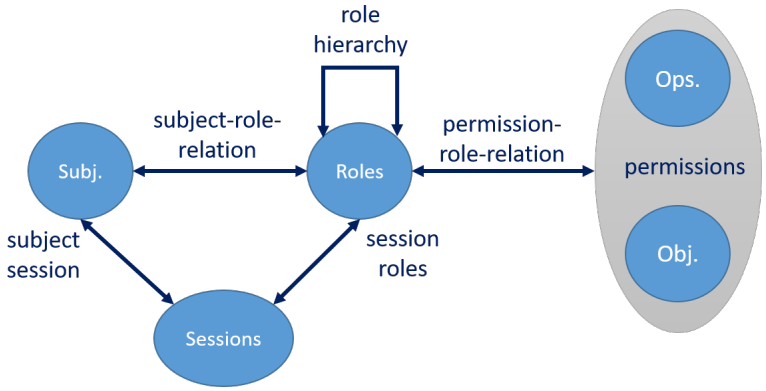


Abbildung 4.6: Elemente des RBAC (eigene Abbildung, angelehnt an [SB15, S. 151, Fig. 4.8(b)])

Eine Rolle vereint mehrere zusammengehörigen Privilegien und ermöglicht deren zentralen Zugriffsverwaltung [BZ12, S. 189]. Das Zusammenfassen von Privilegien zu Rollen vereinfacht die Administration von Zugriffsrechten. Vorwiegend wird die Zuordnung zu Rollen bei Subjekten vorgenommen. Rollen von Subjekten werden auch als Gruppen bezeichnet. Objekte können ebenso anhand ihrer Privilegien zusammengefasst werden. Diese Rollen werden als Objektklassen bezeichnet. Subjekte als auch Objekte können mehreren Rollen angehören, sodass die Zugriffskontrolle sehr feingranular gestaltet werden kann. Die ursprüngliche Idee der rollenbasierten Zugriffskontrolle bezog sich lediglich auf Subjekte. Dieser ursprüngliche Ansatz wird nun anhand der unterschiedlichen Modelle von RBAC (siehe Abbildung 4.5) vorgestellt.

RBAC Base Model Das RBAC Basismodel stellt das Grundgerüst da. Es beinhaltet die Basiskomponenten und Basisfunktionen (siehe Abbildung 4.6), die für eine minimale Umsetzung von RBAC benötigt werden:

- Subjekt Das Subjekt ist wie bei den bereits vorgestellten Modellen die anfragende Instanz, die auf das Objekt zugreifen möchte.
- Rolle Eine Rolle bündelt die Privilegien, die für eine Aufgabenerfüllung notwendig sind. Über ihre Rollen erhalten Subjekte Zugriffsrechte auf Objekte.
- Zugriffsrecht Das Zugriffsrecht erlaubt einen bestimmten Zugriff auf ein oder mehrere Objekte. In diesem Modell werden Zugriffsrechte als *permissions* bezeichnet.

(a) Subject-Role-Object **(b)** Capability List und ACL für RBAC

Abbildung 4.7: Umsetzung des RBAC (eigene Abbildung)

- Session Ein Subjekt kann mehreren Rollen angehören. Eine Session beschreibt in welchen Rollen das Subjekt gerade das System aktiv nutzt.

Daraus lässt sich folgende formale Definition des rollenbasierten Sicherheitsmodells durch ein Tupel

$$RBAC = (S, O, R, P, sr, pr, session)$$

aufstellen, wobei S die Menge an Subjekten, O die Menge an Objekten, R die Menge aller Rollen, P die Menge aller Zugriffsberechtigung und sr,pr,session Relationen sind [Eck08, S. 244]. sr repräsentiert die Zuordnung von Subjekten zu Rollen, pr die Zuordnung der Rollen zu Zugriffsberechtigungen und die session beschreibt die Zuordnung der aktiven Rollen eines Subjektes zu einem bestimmten Zeitpunkt.

Es ist erlaubt, dass ein Subjekt über mehrere Rollen verfügt. Um ein Zugriffsrecht zu erhalten, muss ein Subjekt innerhalb einer Session aktiv in einer oder mehreren Rollen angemeldet sein. Diese aktiven Rollen werden als *session roles* bezeichnet.
Durch Rollen erhält das rollenbasierte Modell einen hohen Grad an Flexibilität und Granularität. Für jede zu tätigende Aufgabe können die Zugriffsrechte in einer Rolle gebündelt werden. Dadurch erhält ein Subjekt lediglich die benötigten Rechte, die zur Aufgabenerfüllung notwendig sind. Sobald ein Subjekt eine Aufgabe nicht mehr durchführen muss, wird ihm die Zuordnung zur Rolle entzogen.

Zur Umsetzung des RBAC Basismodells verwendet man die Transformationen von Zugriffsmatrizen. Abbildung 4.7 zeigt eine mögliche Realisierung. Dabei sind Subjekte mehreren Rollen zugeordnet. Jede Rolle verfügt über Zugriffsrechte auf ein oder mehrere Objekte. Die Zuordnung von Subjekten zu Rollen ist in einer *capability list* realisierbar. Die Umsetzung der *permission-role-relation* spiegelt sich in der ACL wider. Dabei verweist das Objekt auf die Rollen, die ein Zugriffsrecht auf das Objekt besitzen.

RBAC Role Hierarchy Model Rollenhierarchien spiegeln die Hierarchie innerhalb einer Organisation wider [SV00][SB15, S.152]. Dies vereinfacht die Administration von Zugriffsrechten weiter. So können Jobpositionen und Abteilungen die Rolle eines Mitarbeiters im System spezifizieren. Der Chef einer Abteilung hat Beispielsweise Zugriffsrechte auf alle Vorgänge und Objekte innerhalb der Abteilung und steht höher in der Hierarchie als ein Sachbearbeiter der Abteilung. Doch auch Spezialisierungen sind möglich. Während der Chef beispielsweise lediglich einen Überblick über alle Projekte erhalten möchte, braucht ein Mitarbeiter speziell Zugriffsrechte für sein Projekt. Mit

der Rollenhierarchie lassen sich die Organisationsstrukturen abbilden.
Auf die Menge der Rollen wird zur Erzeugung einer Hierarchie wiederum eine partielle Ordnung angewandt [Eck08, S. 246f.]. Über diese Rollenhierarchie sind hierarchische Berechtigungsstrukturen und das Vererben von Rechten möglich. Bei der Implementierung dienen Invarianten zur Validierung der Rollenhierarchie. So darf ein Subjekt nur in seinen zugewiesenen Rollen aktiv sein und nur deren Rechte einnehmen. Bei Vererbung von Rechten innerhalb der Rollenhierarchie erhält das Subjekt zusätzlich die Rechte der höheren Rollen zugewiesen.
Um die Vererbungshierarchie ordentlich umzusetzen, sollte diese Anhand der Abstraktion von Rollen vorgenommen werden. Die hierarchische Abbildung der Organisationsstrukturen stellt eine separate Rollenhierarchie da. Über die Verwendung der Rollenhierarchie entscheidet der Einzelfall.

RBAC Constraint Model Neben dem hierarchischen Modell braucht es ein Einschränkungsmodell, das die Zugriffsrechte neben den Hierarchien weiter definiert. Auf diese Weise können spezielle Sicherheitsrichtlinien realisiert und Rechte von Rollen bedingt bzw. beschränkt werden [Eck08, S.247]. Eine Beschränkung ist eine definierte Rollenbeziehung oder eine zusätzliche Bedingung [SB15, S.152]. Die Regeln der Aufgabentrennung (*separation of duty*) sind anhand von Beschränkungen realisierbar.

Eine Einschränkung ist der “wechselseitige Ausschluss von Rollenmitgliedschaften “[Eck08, S.248]. Diese *mutual exclusive role* erlaubt sowohl eine statische als auch eine dynamische Einschränkung, dass ein Subjekt lediglich einer Rolle dieses gegenseitigen Ausschlusses angehören darf [SB15, S. 153]. Die dynamische Beschränkung wirkt sich lediglich auf die Session eines Subjektes aus und ist in anderen Sessions wieder aufgehoben. Eine Erweiterung ist durch die Rechtebeschränkung möglich. Daraus ergibt sich:

- Ein Subjekt darf lediglich einer Rolle aus dem *mutual exclusive* Set angehören.
- Zugriffsrechte dürfen ausschließlich einer Rolle zugewiesen werden. [SB15, S. 152f.]

Auf diese Weise verfügen Rollen, die sich gegenseitig ausschließen, nicht über gleiche Zugriffsberechtigungen. Dadurch können Objekte, die lediglich über *mutual exclusive* Rollen erreichbar sind, gesondert geschützt werden.
Eine andere Art die Rollenzuteilung von Subjekten zu beschränken, ist das Setzen einer Voraussetzung. So dienen bestimmte Subjekte oder das Angehören zu einer “prerequisite role “ [SB15, S. 153] als Kriterium. Auf diese Weise kann zum Beispiel die Rollenhierarchie umgesetzt werden.

Neben dem gegenseitigen Ausschluss und den Voraussetzungen stellen Kardinalitäten eine weitere Einschränkung da. So setzt beispielsweise die Kardinalität einer Rolle fest, wie viele Subjekte dieser Rolle angehören dürfen [SB15, S.153]. Ebenso kann es die Anzahl an Rollen eines Subjektes oder einer Session einschränken. Mächtige Zugriffsrechte sind in der Regel ebenso durch die Kardinalität der bevollmächtigten Rollen geschützt.

RBAC Consolidated model Alle drei Modelle sind im *consolidated model* zusammengeführt und bilden zusammen das RBAC Zugriffskontrollmodell. Rollen ermöglichen eine aufgabenorientierte Vergabe von Zugriffsrechten. Mithilfe von Rollenhierarchien und Einschränkungen lässt sich die Zugriffskontrolle anwendungsspezifisch modellieren. Durch die Verwendung von Rollen wird die starke Kopplung von Objekten und Subjekten aufgehoben. Rollen sind ein statischer Bestandteil des

Systems, ihre Anforderungen ändern sich selten. Änderungen der Subjektmenge wirken sich nicht mehr auf die Zugriffskontrolle aus und führen nicht zu einer Notwendigkeit von Neumodellierungen. Ein Systemadministrator übt die Administration über die Rollen aus, die sich in der Regel auf das Hinzufügen und Entfernen von Rollen beschränkt.
Das RBAC ist durch seine anwendungsspezifische Modellierbarkeit vielseitig einsetzbar. Zugriffsrechte sind durch das Modell nicht vorgegeben und lassen dadurch viel Freiraum im Design und der Realisierung des Zugriffskontrollmodells.

4.2.4 Attribute-based Access Control (ABAC)

Das attribut-basierte Zugriffsmodell basiert auf komplexen Regeln basierend auf Attributen [Hu14]. Dabei schafft das ABAC eine Möglichkeit, Zugriffsrechte ohne den Zusammenhang von Subjekt und Objekten zu spezifizieren. Die Zugriffskontrolle basiert auf den vorliegenden Attributen von anfragenden Subjekten, angefragten Objekte und Umgebungsvariablen. ABAC kombinieriert Vorteile aus DAC, MAC und RBAC. So dienen *access control lists*, Rollen oder Sicherheitsklassen als Attribute und werden in die Entscheidungsfindung miteinbezogen. ABAC definiert als zentrale Instanz zur Zugriffsrechtevergabe einen Zugriffskontrollmechanism, den "access control mechanism "[Hu14]. In diesen Mechanismus fließen alle Attribute zur Entscheidungsfindung ein. Das ABAC besteht aus drei Grundelementen, den Attributen, der logischen Architektur und den Policies.

Attribute Attribute stellen jegliche Charakteristika eines Systemelements in Form eines Key-Value-Pairs da [Hu14][SB15, S.154]. Das attributbasierte Zugriffskontrollmodell nutzt die Attribute zur Evaluierung, ob ein Zugriffsrecht einem Subjekt erteilt wird. Das ABAC unterscheidet zwischen drei Attributen:

- Subjektattribute: Subjekte sind im ABAC eine Systementität, die auf ein Objekt zugreifen oder den Systemstatus verändern möchte. Jedes Subjekt verfügt über Attribute, die dessen Identität und Charakteristika beschreiben [SB15, S.154]. Subjektattribute sind zum Beispiel der Identifier, der Name und die Zugriffsrolle.
- Objektattribute: Die Definition des Objektes aus anderen Zugriffskontrollmodellen bleibt erhalten. Ein Objekt ist eine durch ein Subjekt angefragte Systemressource. Objektattribute lassen sich aus den Metadaten von Objekten extrahieren [SB15, S.154]. Die Metadaten eines Objektes enthalten etliche Informationen, wie z.B. den Ersteller, das Erstellungsdatum, den Kontext.
- Umgebungsattribute (*environmental attribut*): Umgebungsvariablen beschreiben den Systemstatus zum Zeitpunkt der Zugriffsanfrage. Diese Attribute sind unabhängig von der Anfrage und beinhalten technische, situationsbedingte oder kontextbezogenen Informationen [Hu14]. Beispiele für Umgebungsattribute sind die aktuelle Zeit oder der Systemstatus.

Logische Architektur Der Grundgedanke von ABAC basiert auf der Evaluierung von Attributen in Kombination mit vorgelegten Regeln. Abbildung 4.8 veranschaulicht den logischen Ablauf. Mit der Zugriffsanfrage eines Subjektes auf ein Objekt, startet der Zugriffskontrollmechanismus. Der

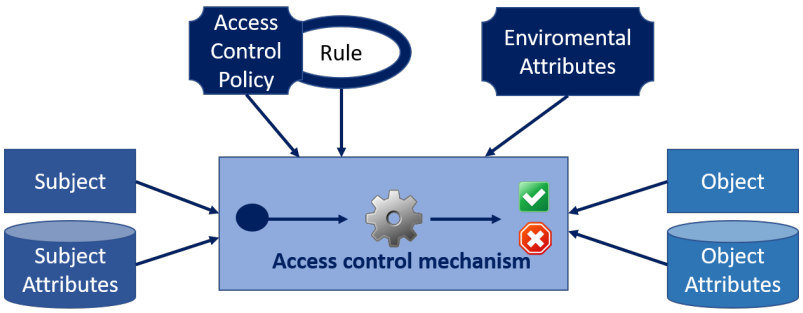


Abbildung 4.8: Elemente des ABAC (eigene Abbildung, angelehnt an [SB15, S. 155, Fig. 4.10])

Mechanismus zieht nun die Regeln aus der vordefinierten *access control policy* heran. Aus diesen Regeln ergibt sich, welche Attribute für die Evaluierung herangezogen werden müssen. Nach der Auswertung erlaubt oder verweigert der Kontrollmechanismus den angefragten Zugriff.

ABAC Policies Zu jedem Objekt im Systems sollte zumindest eine Zugriffsregel in der Policy definiert sein, damit das Objekt unter der Zugriffskontrolle steht. Regeln innerhalb der Policy orientieren sich am Objekt und legen die benötigten Attribute zur Einschränkung des Zugriffs fest. Die Privilegien eines Subjektes sind bei ABAC nicht fest definiert, sondern hängen von der Auswertung der vorliegenden Regeln innerhalb der Policy ab.
Eine Regel, die über den Zugriff $can_access(s, o, e)$ eines Subjektes s auf ein Objekt o in einem System e entscheidet, ist mithilfe einer boolschen Funktion f folgendermaßen aufzustellen:

$$Rule : can_access(s, o, e) \leftarrow f(ATTR(s), ATTR(o), ATTR(e))$$

[SB15, S. 158f.]
Mit der Aufstellung von boolschen Ausdrücken, lassen sich vielerlei Regeln definieren. Für eine Übersichtlichkeit und zur Realisierung Komplexer Zugriffsregeln lassen sich Regeln auch miteinander zu einer neuen Regel kombinieren. Dadurch das die Regeln lediglich von Attributen und nicht von den Instanzen selbst abhängig sind, lassen sich eine Vielzahl von Zugriffsrechten abstrakt für mehrere Entitäten modellieren [.] Vor allem für die sich oftmals ändernde Subjektmenge ist dies Loslösung von Identitäten ein Vorteil. Aus der Abstraktion folgt jedoch eine gewisse Komplexität der Aufstellung der Regeln. Regeln beziehen sich auf das Objekt und definieren vor allem die erlaubten Zugriffe. Auf der anderen Seite können Regeln explizit Zugriffe verweigern, indem der boolsche Ausdruck basierend auf den Attributen falsch ergibt.

Access Control Languages Während in anderen Sicherheitsstrategien die Policies vor allem zur Definition von abstrakten Regeln für die Modellierung in Zugriffsmatrizen oder Zugriffslisten dienen, nutzt ABAC die Regeln der Policy direkt im Modell. Die Verwendung der Policies im ABAC basiert auf der Formulierung in einer “access control language“[AAA+18]. Diese sollte eine hohe Ausdruckskraft nahe der natürlichen Sprache bei gleichzeitiger Einfachheit bieten. Die Definition von eigenen Sprachen ist möglich, die Verwendung von bereits existierenden Sprachen bietet sich jedoch an.

- XACML ist eine der viel verwendete Sprache, da sie mit unterschiedlichen Sicherheitsmodellen, wie ACL,RBAC oder ABAC, kompatibel ist. Diese Sprache ist vom OASIS-Konsortium standardisiert basiert auf XML. [AAA+18]
- XACL basiert ebenfalls auf XML und wurde im Jahr 2000 von IBM definiert. Die Sprache ist für XML Dokumente festgelegt und erlaubt die vier Standardoperationen read, write, create, delete [AAA+18].
- APPEL kommt ursprünglich aus der Telefonanrufsteuerung. Aufgrund seiner Erweiterbarkeit ist diese Sprache in unterschiedlichen Domänen, so zum Beispiel der Zugriffskontrolle, anwendbar. Die Sprache bietet eine sehr ausdrucksstarke Syntax und modelliert Regeln anhand von Auslösern, Bedingungen und Aktionen [AAA+18].
- EPAL wurde zur Einhaltung von Data Governance definiert und ist aus diesem Grund ebenfalls in der Zugriffskontrolle einsetzbar. Es erlaubt die Definition von Zugriffsbeschränkungen (Zugriffsvergabe, Zugriffsverweigerung). Neben Bedinungen können Objekthierarchien oder Verwendungszwecke abgebildet und verwendet werden [AAA+18].

Diese exemplarisch und abstrakt vorgestellten Richtliniensprachen sind ein Auszug aus einem weitreichenden Katalog. Jede Sprache legt seine Schwerpunkte auf unterschiedliche Charakteristika und Funktionen. Vor allem XACML findet im Bereich der Zugriffskontrolle häufig Verwendung. Die Architektur von XACML spiegelt die Hauptelemente des ABAC wieder und eignet sich somit für die Realisierung von ABAC.

4.2.5 Application vs. Database Security Models

Die zuvor vorgestellten Sicherheitsstrategien zeigen die unterschiedlichen Möglichkeiten der Zugriffskontrolle auf. Unabhängig von der verwendeten Sicherheitsstrategie entscheidet die Verwendungsebene über die Auswirkungen der Zugriffskontrolle. Die Zugriffskontrolle kann auf unterschiedlichen Ebenen erfolgen, der Anwendungsebene, der Datenbankebene oder der Systemebene. Die Zugriffskontrolle des Betriebssystems findet auf der Systemebene statt. Da diese in der Regel sehr allgemein gehalten ist, verfügen Anwendungen und Datenbanken über eigene Sicherheitsmechanismen zur weiteren Absicherung und Modellierung eigener Regeln.
Mit der Zugriffskontrolle direkt an der Datenbank sind die Datenobjekte gesondert geschützt. Keine unberechtigten Subjekte erhalten damit Zugriff auf die Daten. Anwendungen müssen sich für einen Datenbankzugriff authentifizieren und eine separate Zugriffskontrolle für ihre Nutzer bzw. Subjekte durchführen. Bei der anwendungsspezifischen Zugriffskontrolle führt die Anwendung eine eigene Subjektverwaltung und die damit einhergehende Zugriffskontrolle durch. Eine Anwendung kann als Erweiterung der Zugriffskontrolle der Datenbank verwendet werden, um weitere Kontrollmechanismen unabhängig der Datenbankimplementation umzusetzen.

4.3 Fazit

In den vorangegangenen Abschnitten sind unterschiedliche Sicherheitsstrategien und deren Zugriffskontrollmodelle vorgestellt. Die benutzerbestimmbare Zugriffskontrolle (DAC) nutzt das Eigentümer-Prinzip zur Festlegung der Zugriffsrechte in einer Zugriffsmatrix. Dieses benutzerbestimmbare Prinzip birgt das Risiko von Inkonsistenzen und gegenseitigen Aufhebungen von

Zugriffsregeln im System. Zugriffsmatrizen werden unter anderem in Betriebssystemen verwendet. Ein Beispiel dafür ist POSIX-Modell, das bei UNIX-Betriebssystemen genutzt wird.

Bei der systembestimmten Zugriffskontrolle (MAC) liegt die Administration von Zugriffsrechten bei einer zentralen Autorität. Die Rechtevergabe erfolgt mittels zusätzlicher Kriterien, der Sicherheitsklassifizierung von Subjekten und Objekten. Die Zugriffskontrollmodelle von MAC dienen entweder der Einhaltung der Geheimhaltung (Bell-LaPadula) oder Integrität (Biba) von Objekten. Wenige Modelle, wie zum Beispiel das Chinese Wall Modell, vereinen Geheimhaltung und Integrität. Dabei ist das Chinese Wall Modell jedoch sehr restriktiv und vor allem für die Vermeidung von Interessenkonflikten sinnvoll. Das *Clark Wilson Modell* realisiert eine Aufgabentrennung im Bezug auf die Vergabe von Zugriffsrechten. Autorisierte Subjekte führen eine Überprüfung durch, bevor sie Subjekten Zugriffsrechte gewähren. Diese Aufgabentrennung der Zugriffsvergabe ist für dynamische Systeme sehr interessant. MAC-Zugriffskontrollmodelle sind alle sehr restriktiv definiert und sind in der Praxis mithilfe Erweiterungen und Kombinationen dynamischer modelliert.

Für größere Systeme mit vielen Objekten und Subjekten bietet sich eine Gruppierung der Elemente und eine darauf basierende Zugriffskontrolle an. Die rollenbasierte Zugriffskontrolle verwirklicht die Umsetzung der Rechtevergabe anhand von Rollen. Subjekte erhalten über ihre zugewiesen aktiven Rolle ihre Zugriffsrechte zugeteilt. Über die Verwaltung der Zugriffsrechte anhand von Rollen ist die bei den anderen Modellen vorliegende enge Kopplung der Zugriffsrechte an Subjekte und Objekte aufgehoben. Des Weiteren lassen sich mit Rollenhierarchien Organisationsstrukturen und Vererbungen von Zugriffsrechten abbilden. Mithilfe von Rollenbeschränkungen sind Ausnahmen und zusätzliche Bedingungen modellierbar.

Als letzte Sicherheitsstrategie wurde die attributbasierte Zugriffskontrolle vorgestellt. Sie nutzt vorliegende Attribute um in einem Entscheidungsfindungsprozess über die Vergabe von Zugriffsrechten abzustimmen. Innerhalb von Zugriffskontrollrichtlinien sind Regeln für den Entscheidungsprozess definiert. Der *access control mechanism* führt bei einer Anfrage den Entscheidungsfindungsprozess mithilfe der definierten Regeln, der Subjektattribute, der Objektattribute und der Umgebungsattribute durch. Für die Definition der Sicherheitsrichtlinien und ihren Regeln können bei der attribut-basierten Zugriffskontrolle verschiedene Sprachen herangezogen werden. Vor allem XACML bietet sich bei ABAC an.

Die Anwendung von Zugriffskontrollmechanismen geschieht auf mehreren Ebenen. Während die Zugriffskontrolle von Betriebssystemen auf dem Zugriffsmatrix-Modell aufbauen, nutzen Anwendungs- und Datenbanksysteme weit komplexere Zugriffskontrollmethoden. Das Data Lake Konzept ist unabhängig von Technologien. Aus diesem Grund eignet sich zur Zugriffkontrolle kein Zugriffskontrollkonzept auf Datenbankebene sondern auf Anwendungsebene. Das folgende Kapitel beschäftigt sich mit der Zugriffskontrolle für Data Lakes.

5 Erweitertes Zugriffskontrollmodell für Data Lakes

Zugriffskontrollmodelle für Data Lakes bauen bisher auf den verwendeten Technologien auf. Ein allgemein auf Data Lakes abgestimmtes Konzept liegt noch nicht vor. Dieses Kapitel behandelt die benötigten Eigenschaften eines für Data Lakes optimierten Zugriffskontrollmodells. Mit der Evaluation der aufgestellten Kriterien anhand bestehender Zugriffskontrollmodelle ist die Ausarbeitung eines kombinierten Modells für Data Lakes motiviert.

5.1 Anforderungen

Die in Kapitel 4 vorgestellten Sicherheitsstrategien zeigen die unterschiedlichen Schwerpunkte und Eigenschaften der vorhandenen Zugriffskontrollmodelle auf. In der in vorgestellten Referenzarchitektur für Data Lakes sind die Ansatzpunkte für eine Zugriffskontrolle bereits hervorgehoben. Anhand der Referenzarchitektur, des allgemeinen Konzeptes von Data Lakes und der bekannten Eigenschaften von bestehen Zugriffskontrollmodellen ergeben sich bestimmte Anforderungen für ein Zugriffskontrollkonzept für Data Lakes. Diese sind im Folgendem aufgelistet. Die zur Feststellung über die Erfüllung einer Anforderung benötigten Kriterien sind dabei direkt vermerkt.

A1 Anwendungsbasiertes Zugriffskontrollmodell: Data Lake ist ein Konzept unabhängig von Technologien. Entsprechend ist das Zugriffskontrollkonzept ebenfalls unabhängig der verwendeten Speichertechnologie zu halten. Zur Umsetzung muss ein Zusammenspiel mit der datenbank-basierten Zugriffskontrolle möglich sein.

- **K1** Das Zugriffskontrollmodell ist auf der Anwendungsebene umsetzbar.
- **K2** Das Zugriffskontrollmodell kann die Zugriffskontrolle darunterliegender Ebenen (System- oder Datenbankebene) erweitern.

A2 Feingranulare, objektspezifische Zugriffsrechte: Laut der Definition des Data Lake Konzeptes in Abschnitt 3.1 kommen Daten aus unterschiedlichen Datenquellen zusammen. Über die während des Data Wrangling Prozesses erhaltenen Richtlinien zur Datenverwendung ergeben sich objektspezifische Zugriffsrechte. Kapitel 4 zeigte verschiedene mögliche Zugriffsrechte der Sicherheitsstrategien. Die Differenzierung zwischen verschiedenen Zugriffsrechten unterstützt die für Data Lakes benötigte aufgabenspezifischen Zugriffsvergabe.

- **K1** Die Zugriffsrechtevergabe erfolgt objektspezifisch.
- **K2** Das Zugriffskontrollmodell definiert grundlegenden Zugriffsrechte.
- **K3** Es können weitere individuelle Zugriffsrechte definiert werden.

A3 Bündelung zusammengehöriger Privilegien für Gruppen: Mit der Bündelung von Privilegien für Subjekte und Objekte vereinfacht sich die Administration und Vergabe von Zugriffsrechten. In einem Data Lake bietet sich dies bei der Vielzahl von Subjekten und Objekten an.

- **K1** Zugriffsrechte können für Subjektrollen definiert werden.
- **K2** Subjekte ist es möglich, mehreren Rollen anzugehören und so unterschiedliche Zugriffsrechte zu erhalten.
- **K3** Ein Subjekt kann zu einem Zeitpunkt ausschließlich in einer Rolle aktiv sein.
- **K4** Zugriffsrechte können für Objektklassen definiert werden.

A4 Sicherstellung der Geheimhaltung (Confidentiality): Die Zugriffskontrolle kann zur Erfüllung der Geheimhaltung von Objekten dienen. In einem Data Lake ist die Absicherung vor unberechtigten Zugriffen ebenso wichtig wie bei anderen Speicherkonzepten.

- **K1** Es existiert eine Einordnung von Subjekten und Objekten in Sicherheitsklassen.
- **K2** Es existieren Regeln, die zur Zugriffsrechtevergabe oder Zugriffsbeschränkung basierend auf den Sicherheitsklassen des zugreifenden Subjekte auf ein Objekt führen.

A5 Sicherstellung der Objektintegrität (Integrity): Die Integrität eines Objektes ergibt sich aus der Absicherung vor Manipulation. Ein Data Lake erlaubt die Speicherung aus unterschiedlichen Datenquellen und Zugriffe von unterschiedlichen Instanzen. Zur Erhaltung der Objektintegrität benötigt es eine Zugriffskontrolle mit entsprechender Sicherstellung.

- **K1** Es existieren Integritätsklassen, die einerseits die Objektintegrität andererseits die Vertrauenswürdigkeit eines Subjektes widerspiegeln.
- **K2** Mithilfe von Regelungen können Zugriffsrechte zur Objektmanipulation (z.B. write, append,...) anhand der Integritätsklassen von Subjekt und Objekt bestimmt werden.

A6 Regelung der initialen Rechtevergabe: Ein Data Lake stellt eine sehr dynamische Speicherlösung da, deren Administration sehr aufwendig ist. Mit der Regelung einer initialen Rechtevergabe innerhalb der Datenbeschaffungsphase während des Data Wrangling Prozesses unterliegt das Objekt einem grundlegenden Schutz.

- **K1** Das Zugriffskontrollmodell beschreibt einen initialen Zustand der Zugriffsrechte auf das Objekt.
- **K2** Es bestehen Regelungen zur weiteren Festlegung der Zugriffsrechte eines Subjektes auf das Objekt.

A7 Dynamische Anpassbarkeit der Rechtevergabe: Das Zugriffskontrollmodell benötigt eine gleiche Dynamik wie das Data Lake an sich. Dies bewirkt eine Anpassbarkeit an Veränderungen des Datenbestands und der Anwendungsszenarien.

- **K1** Die Zugriffsrechtevergabe erfolgt dynamisch zur Anfragezeit des Subjektes auf ein Objekt und ist nicht statisch im System hinterlegt.
- **K2** Bei der Zugriffsrechtevergabe spielen neben der Subjekt- und Objektidentität mehrere Aspekte, wie zum Beispiel Attribute eine Rolle.

K3 Eine Änderung in der Zugriffsrechtevergabe kann jederzeit eingespielt werden und ist direkt wirksam.

K4 Es ist möglich, eine temporäre Zugriffsfreigabe zu erteilen.

A8 Definition komplexer Zugriffsbeschränkungen: Aufgrund unterschiedlicher Anwendungsszenarien und vorliegenden Nutzungsrichtlinien von Objekten ergeben sich komplexe Einschränkungen der Zugriffsrechte. Diese bilden sogenannte Zugriffsbeschränkungen.

K1 Ausnahmefälle bei der Zugriffsrechtevergabe sind abweichend der restlichen Zugriffsrechtevergabe modellierbar.

K2 Es ist möglich, Zugriffsrechte aufgabenorientiert festzulegen.

Die Anforderungen fassen die gewünschten Eigenschaften eines Zugriffskontrollmodells für Data Lakes zusammen. Anhand der aufgestellten Kriterien ist eine Evaluation über die Erfüllung der Anforderungen möglich. Der nächste Abschnitt evaluiert die bereits vorhandenen Zugriffskontrollmodelle anhand der definierten Kriterien.

5.2 Vergleich bestehender Zugriffskontrollmodelle

Die aufgestellten Anforderungen sind aus der Analyse der vorgestellten Zugriffskontrollmodelle in Kapitel 4 abstrahiert. Kapitel 4 beinhaltet mehrere verschiedene Sicherheitsstrategien samt einiger Zugriffskontrollmodelle. Diese Zugriffskontrollmodelle sind anhand der aufgestellten Kriterien für ein Zugriffskontrollmodell für Data Lakes evaluiert. Aus der Evaluation und dem Vergleich der Modelle begründet sich ein hybrider Ansatz zur Abdeckung aller Anforderungen. In Tabelle 5.1 sind die Ergebnisse der Evaluation zusammengefasst. Dabei sind zur Übersichtlichkeit die Kriterien innerhalb der Anforderungen nicht separat aufgelistet. Innerhalb einer Zelle sind alle Kriterien einer Anforderung evaluiert. Ein Häckchen beschreibt die Erfüllung eines Kriteriums, ein Kreuz, dass das Kriterium nicht erfüllt ist. Zum Beispiel ist für das Zugriffsmatrix-Modell K1 von A1 erfüllt und K2 von A1 nicht erfüllt.

Das Zugriffsmatrix-Modell wird in vielen Systemen zur Definition der Zugriffsrechte von Subjekten auf Objekte verwendet. Es dient als Grundlage für andere Modelle. Die Granularität der Zugriffsmatrix ist nicht im Modell definiert, sodass dies durch die Umsetzung bestimmt wird. Anforderung A3, A4, A5 sind nicht durch Vorgaben im Modell spezifiziert, sodass sie als nicht erfüllt gekennzeichnet sind. Mithilfe des Eigentümer-Prinzips und der Anpassung der Zugriffsrechte durch Kommandos ist die initiale Rechtevergabe (A6) gewährleistet. Eine Zugriffsmatrix ist nicht sehr dynamisch und lässt die keine Modellierung komplexer Zugriffsbeschränkungen zu.

Zur benutzerbestimmbaren Zugriffskontrolle (MAC) sind im Kapitel 4 vier unterschiedliche Modelle vorgestellt, die im Folgenden evaluiert sind. Das Bell-LaPadula Modell gehört zu den Modellen deren Fokus auf der Geheimhaltung von Informationen liegt. Das Modell definiert fünf Zugriffsrechte, die an bestimmte Regeln gebunden sind. Diese decken die benötigte Funktionalität ab. Eine Erweiterung des Modells durch eigene Zugriffsrechte ist nicht vorgesehen (K3 von A2 nicht erfüllt). Die Einstufung der Subjekte in Sicherheitsklassen und die strikten Zugriffsregeln schränken Subjekte höherer Sicherheitsstufen ein. Zur Ausführung eines Zugriffsrechtes ist es Subjekten möglich, sich in einer niedrigeren Sicherheitsstufe einzuloggen. Daraus ergibt sich, dass

	A1	A2	A3	A4	A5	A6	A7	A8
Zugriffsmatrix	✓✓	✓✓✗	✗✗✗✗	✗✗	✗✗	✓✓	✗✗✓✗	✗✗
Bell-LaPadula	✓✓	✓✓✗	✓✓✓✓	✓✓	✗✗	✓✓	✓✗✗✗	✗✓
Biba	✓✓	✓✓✗	✓✗✓✓	✗✗	✓✓	✓✓	✓✗✗✗	✗✗
Clark-Wilson	✓✓	✓✗✗	✓✗✓✓	✗✗	✗✗	✓✓	✓✗✗✗	✗✗
Chinese Wall	✓✓	✓✓✗	✗✗✗✓	✗✓	✓✓	✓✓	✓✓✗✗	✗✗
RBAC	✓✓	✓✓✓	✓✓✓✓	✗✗	✗✗	✓✗	✗✗✓✓	✗✓
ABAC	✓✓	✓✓✓	✓✗✗✓	✓✗	✓✗	✓✓	✓✓✓✓	✓✓

Tabelle 5.1: Evaluation bestehender Zugriffskontrollmodelle

die Anforderung A3 samt aller Kriterien erfüllt ist. Da es sich bei Bell-LaPadula um ein Modell zur Sicherstellung der Geheimhaltung und nicht der Objektintegrität handelt ist A4 erfüllt und A5 nicht erfüllt. Eine dynamische Anpassbarkeit in der Rechtevergabe sieht Bell-LaPadula nicht vor. Komplexe Zugriffsbeschränkungen sind über die zwei Basisregeln im Bell-LaPadula nicht möglich. Das Biba Modell unterscheidet sich bei der Erfüllung der Anforderungen wenig von dem Bell-Lapadula Modell. Der wesentliche Unterschied liegt in der Sicherstellung der Objektintegrität statt der Geheimhaltung.

Das Clark-Wilson Integrity Modell basiert auf der Definition erlaubter atomarer Transaktionen. Das Modell definiert erlaubte Zugriffsrechte innerhalb Transaktionen. Dabei sind keine Zugriffsrechte definiert, sondern lediglich ob ein Zugriff erfolgen darf. Somit sind K2 und K3 von A2 nicht erfüllt. Die Bündelung der gemeinsamen Privilegien findet in gewisserweise durch die Zuordnung von Subjekten zu zertifizierten Instanzen statt (K1, K3 von A3 erfüllt). Objekte sind in geschützte und nicht-geschützte Objektklassen kategorisiert(K4 von A3 erfüllt). Bei dem Modell handelt es sich um ein Zugriffskontrollmodell zur Sicherstellung der Objektintegrität und nicht Geheimhaltung. Aspekte zur komplexen Zugriffsbeschränkung und zur dynamischen Anpassbarkeit sind in diesem Modell nicht berücksichtigt.

Das Chinese Wall Modell erlaubt die Sicherstellung der Geheimhaltung sowie der Integrität von Objekten. Hinter dem Modell verbirgt sich die Idee, Interessenskonflikte zu vermeiden. Die Zugriffshistorie eines Subjektes beeinflusst die zukünftige Zugriffsrechtevergabe. Das Modell unterscheidet lediglich zwischen zwei grundlegenden Rechten (read, write), sodass K2 von A2 erfüllt, K3 von A2 jedoch nicht erfüllt ist. Die Bündelung von Objekten in sogenannte DataSets ist mit einer Objektklassifizierung vergleichbar (K4 von A3 erfüllt). Zwar existieren beim Chinese Wall Modell keine Sicherheitsklassen, es bestehen jedoch Regelungen basieren auf der Zugriffshistorie eines Subjektes. Diese definieren den erlaubten Informationsfluss. Auf diese Weise ist K1 von A4 nicht erfüllt, K2 von A4 erfüllt. Die Unterscheidung zwischen Data Sets und Konfliktklassen schützen die Objektintegrität und ermöglichen die Festlegung von Regelungen zur Objektmanipulation (A5 erfüllt). Ausgehend von allumfassendem Leserecht, führen Zugriffe auf ein DataSet zum Auschluss der Zugriffsrechte auf Konfliktklassen des DataSets. Somit sind A6 und K1,K2 von A7 erfüllt.

Das rollenbasierte Zugriffskontrollmodell ist sehr abstrakt definiert. Das Modell bündelt die Zugriffsrechtevergabe auf Objekte für Subjekte in Rollen (A2 und A3 erfüllt). Inwieweit die Rollen eine Sicherheitsklasse oder eine Integritätsklasse darstellen können ist im Modell nicht ausgeführt, sodass A4 und A5 nicht erfüllt sind. In einer Zugriffsmatrix sind die Zugriffsrechte der Rollen auf

Objekte festgelegt. Vergleichbar mit dem Zugriffsmatrixmodell ist A6 und K3 von A7 somit erfüllt. Während die Zugriffsrechte an Rollen gebunden sind und somit nicht zur Anfragezeit entschieden werden (K1,K2 von A7 nicht erfüllt), kann die Zugriffsfreigabe temporär durch eine dynamische Zuordnung von Subjekten zu Rollen durchgeführt werden (K4 von A7 erfüllt). Die Definition der Rollen und ihren Zugriffsrechten ist in der Regel aufgabenorientiert, sodass K2 von A8 erfüllt ist.

Als letztes Sicherheitsstrategie ist die attribut-basierte Zugriffskontrolle evaluiert. Mit einer Zugriffskontrolle basierend auf Attributen sind komplexe Zugriffsregeln und die Modellierung von Klassen für Objekte und Subjekte modellierbar(A3-K1, A3-K4, A4-K1, A5-K1 erfüllt). Darüber hinaus definiert die attribut-basierte Zugriffskontrolle keine standardisierten Regeln zur Zugriffsrechtevergabe bei vorliegenden Klassen (A3-K2, A3-K3, A4-K2, A5-K2 nicht erfüllt). Die attribut-basierte Zugriffskontrolle ist dynamisch modellierbar und erlaubt komplexe Zugriffsbeschränkungen durch die Definition von Regeln innerhalb der Richtlinien (A7 und A8 erfüllt).

Die Evaluation zeigt, dass keines der behandelten Modelle alle Anforderungen direkt erfüllt. Als bestes schneidet die attribut-basierte Zugriffskontrolle ab. Diese erlaubt viel Modellierungsfreiraum, sodass Elemente anderer Zugriffskontrollen eingebaut werden können. Des Weiteren ist die Umsetzung eines Modells zur Sicherstellung der Geheimhaltung und der Datenintegrität bei der attribut-basierten Zugriffskontrolle durch die Festlegung von Regeln in den Richtlinien möglich.

5.3 Konzept des erweiterten Zugriffskontrollmodells für Data Lakes

Die Evaluation der Zugriffskontrollmodelle zeigt auf, dass kein Modell allen Anforderungen entspricht. Ein Data Lake ermöglicht eine Datenvielfalt mit an Technologien gebundenen Einschränkungen hinsichtlich der Datenspeicherung. Die Flexibilität eines Data Lakes ist ein wesentliches Merkmal, das sich auch auf die Zugriffskontrolle auswirkt. Weitere Anforderungen entstehen durch die Aufbereitung der Daten zum Anfragezeitpunkt, den verschiedenen Nutzergruppen und den unterschiedlichen Datenquellen. Im Folgenden ist ein erweitertes Zugriffskontrollmodell für Data Lakes aufgestellt, das den Herausforderungen gerecht wird und der einfacheren Administration von Zugriffsrechten dient.

5.3.1 Logische Architektur

Das erweiterte Zugriffskontrollmodell setzt sich aus mehreren Zugriffskontrollmodellen zusammen. Grundlage für diesen hybriden Ansatz bietet das attribut-basierte Zugriffskontrollmodell. In dieses fließen durch spezielle Subjekt- und Objektattribute mithilfe der Verwendung bestehender Regeln Aspekte aus dem Bell-LaPadula, dem Biba und dem rollen-basierten Zugriffskontrollmodell ein. Das Modell fokussiert sich auf der Kombination von Sicherheits- und Integritätsklassen auf eine Mehrfaktorzugriffsregelung. Abbildung 5.1 zeigt das erweiterte Zugriffskontrollmodell, wobei die neuen Elemente Orange hervorgehoben sind. Es setzt sich aus mehreren Komponenten entsprechend dem attribut-basierte Zugriffskontrollmodell zusammen:

- Security Artifacts: Die Sicherheitsartefakte enthalten alle Definitionen für das Zugriffskontrollmodell. Aus den Artifakten leiten sich die Regeln für den Zugriffskontrollmechanismus ab. Unter den Sicherheitsartefakten befinden sich die Auflistung erlaubter Zugriffsrechte, die

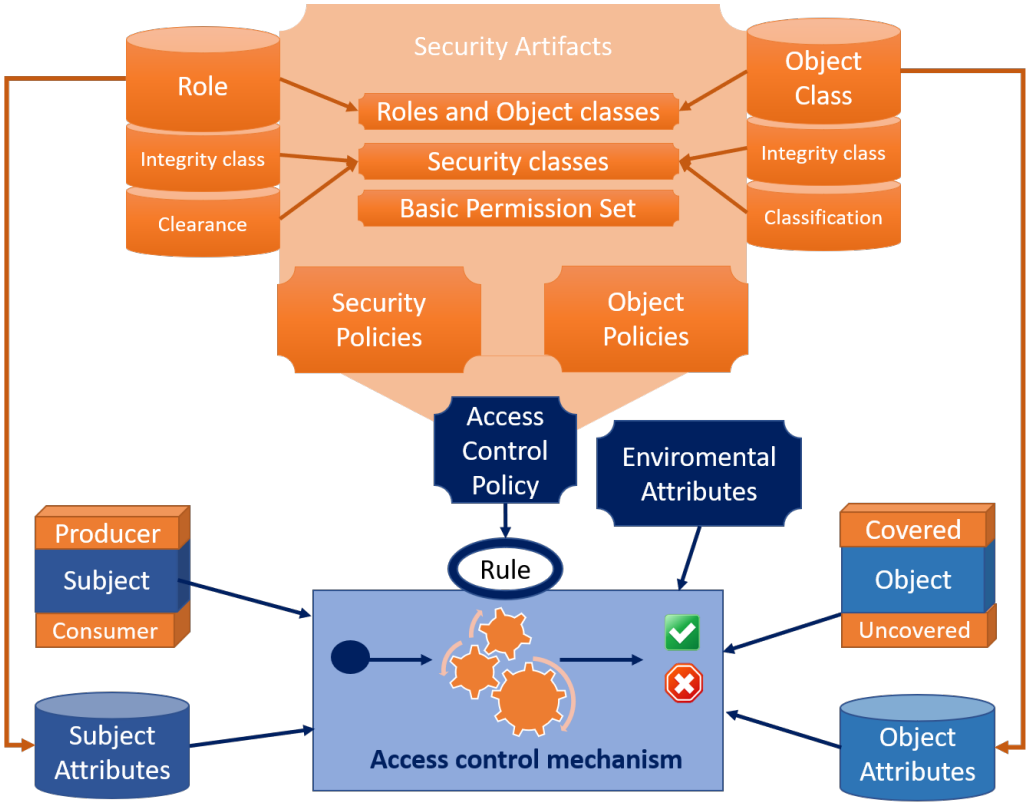


Abbildung 5.1: Erweiterung der attribut-basierten Zugriffskontrolle (ursprüngliche Elemente blau, Erweiterungen orange dargestellt)

Festlegung der Sicherheits- und Integritätsklassen sowie die Definitionen der Subjektrollen und Objektklassen. Zu den Rollen und Objektklassen sind Integritäts- und Sicherheitsklassen zugeordnet. Innerhalb der Sicherheitsrichtlinie sind die partiellen Ordnungen der Klassifizierungen sowie deren speziellen Regeln zur Zugriffsrechtevergabe ausformuliert. In den Richtlinien der Objekte sind deren Nutzungsrichtlinien hinterlegt. Darüber hinaus ist es dort möglich, weitere objekt-bezogenene Regelungen zur Zugriffskontrolle zu definieren.

- Access Control Policies: Unter den Zugriffskontrollrichtlinien sind alle Inhalte aus den Sicherheitsartefakten zusammengefasst, sodass sich daraus die Regeln für die entsprechende Zugriffsanfrage ableiten lassen.
- Subjekte: In einem Data Lake liegen im wesentlichen zwei Arten von Subjekten vor: auf der einen Seite Subjekte, die Objekte vor allem anlegen (Producer), auf der anderen Seite Subjekte, die auf Objekte zugreifen und diese manipulieren (Consumer).
- Subjektattribute: Ein Subjekt verfügt über unterschiedliche Subjektattribute. Dieses Zugriffskontrollmodell setzt Subjektrollen als ein Subjektattribut voraus. Ein Subjekt darf zum Anfragezeitpunkt lediglich einer aktiven Rollen angehören. Anhand dieser erfolgt die Integritäts- und Sicherheitsüberprüfung. Weitere Subjektattribute können als Bedingung für ein Zugriffsrecht in der weiteren Überprüfung dienen.

- Objekte: Für die Zugriffskontrolle spielen die Objektinhalte keine Rolle. Aus diesem Grund wirkt sich die Datenvielfalt und Datenstrukturen nicht auf die Zugriffskontrolle im Data Lake. Objekte bleiben weiterhin die Instanzen, auf das ein Subjekte zugreifen möchte. Nicht alle Daten im Data Lake sind zwangsweise durch die Zugriffskontrolle und dessen Integritätsschutz abgedeckt. Aus diesem Grund unterscheidet das Modell zwischen geschützten und ungeschützten Daten (uncovered und covered).
- Objektattribute: Objekt-attribute stellen im Data Lake die Metadaten da. Sie enthalten zusätzliche Informationen zu dem Objekt. Das erweiterte Zugriffskontrollmodell benötigt zur Abgleichung der Integritäts- und Sicherheitsklassen mit dem Subjekt eine Zuordnung jedes Objektes zu einer Objektklasse innerhalb der Objektattribute.
- Umgebungsattribute: Neben Subjekt- und Objektattributen können Zugriffsrechte an Umgebungsattribute gebunden sein. Damit lassen sich zum Beispiel Zugriffe auf Bürozeiten oder an den Systemstatus binden.
- Zugriffskontrollmechanismus: Der Zugriffskontrollmechanismus der attribut-basierten Zugriffskontrolle durchläuft dazu mehrere Überprüfungen. In einem ersten Schritt findet eine Integritätsprüfung statt. Ist diese erfolgreich durchlaufen, erfolgt die Überprüfung der Sicherheitsstufen und danach der Abgleich mit zusätzlichen Regelungen. Auf diese Weise kann eine Anfrage frühzeitig abgelehnt werden, wenn die Voraussetzungen der Integritätsklasse und der Sicherheitsklasse nicht erfüllt sind.

Das vorgestellte Zugriffskontrollmodell ist eine Abwandlung der attribut-basierten Zugriffskontrolle unter dem Einfluss anderer Zugriffskontrollmodelle. Mit der Aufstellung der Sicherheitsartefakte ist die Zugriffskontrolle genauer definiert. Folgende Abschnitte beinhaltet eine genauere Beschreibung der Sicherheitsartefakte und des Zugriffskontrollmechanismus.

5.3.2 Sicherheitsartefakte

Die Sicherheitsartefakte dienen zur Modellierung der Zugriffskontrolle für ein System. Das *Basic Permission Set* legt die Menge der erlaubten Zugriffsrechte P = { create, read-only, read-write, append, execute, control } fest:

- create: Diese Zugriffsrecht erlaubt dem Subjekt neue Objekte zu erstellen. Um das create-Zugriffsrecht ausüben zu dürfen, muss das Subjekt als Producer in den Zugriffskontrollrichtlinien registriert sein. Innerhalb der Zugriffskontrollrichtlinien sind die festzulegenden Attribute eines vom Subjekt erstellten Objektes hinterlegt.
- read-only: Der reine Lesezugriff auf ein Objekt hat keine Auswirkungen auf die Objektintegrität, jedoch kann es Integrität des Subjektes beeinflussen. Zur Sicherstellung der Vertraulichkeit während eines Lesezugriffes dienen die Sicherheitsklassen.
- read-write: Der Schreibzugriff beinhaltet einen Lesezugriff. Aus diesem Grund müssen Sicherheits- und Integritätsklassen die Sicherstellung der Integrität und der Vertraulichkeit gewährleisten.
- append: Der verändernde Zugriff erlaubt es, das Objekt zu erweitern ohne die Inhalte des Objektes zu lesen.

- execute: Dieses Zugriffsrecht erlaubt es, Objekte bzw. andere Subjekte auszurufen und auszuführen.
- control: Um restriktive Zugriffsbeschränkungen zeitweise aufheben zu können, erlaubt dieses Zugriffsrecht die einmalige Rechteweitergabe an ein anderes Subjekt.

Zu den Sicherheitsartefakten gehören ebenfalls eine Menge an Rollen R_s und Objektklassen K_o um die Verwaltung der Objekte und Subjekte einfacher zu gestalten. Jedes Objekt im Data Lake ist genau einer Objektklasse zugeordnet. Ein Subjekt kann über mehrere Rollen verfügen, jedoch zu einem Zeitpunkt ausschließlich in einer Rolle aktiv sein. Jeder Rolle und Objektklasse sind eine Integritäts- und eine Sicherheitsklasse zugeordnet. Über diese Zuordnung sind die höchsten Sicherheitsstufen definiert, die die Entität einnehmen kann. Einer Entität ist es möglich, niedrigere Sicherheitsstufen einzunehmen.

Die einzelnen Sicherheits- und Integritätsklassen sind in den Sicherheitsrichtlinien über partielle Ordnungen definiert. Die Integritätsklassen definieren inwieweit einer Entität vertraut werden kann. Sie orientieren sich an dem Biba Modell und sind in einer strikten Hierarchie anzuordnen. Über die Hierarchie lassen sich die Integritätsklassen miteinander vergleichen. Es gelten gemäß dem Biba Modell folgende Regeln einer Zugriffsaktion $p \in P$ eines Subjektes s der Integritätsklasse $I_s \in I$ auf ein Objekt der Integritätsklasse $I_o \in I$:

- Simple-Integrity-Regel (write down): $(p = read - write \vee p = append) \wedge I_s \geq I_o$
- Integrity-Confinement-Regel (read up): $p = read - only \wedge I_s \leq I_o$
- Invocation-Eigenschaft: $(p = execute \vee p = control) \wedge I_{S_1} \geq I_{S_2}$
- Integrity-Creator-Regel: $p = create \wedge I_s \geq I_o$

Die Sicherheitsklassen sind vergleichbar mit Geheimhaltungsstufen. Eine Sicherheitsklasse $SC(L, K) \in SC$ besteht aus einer Kategorie K und einem Sicherheitslevel L entsprechend dem Bell-LaPadula Modell. Sie lassen sich anhand

$$SC_x(L_x, K_x) \leq SC_y(L_y, K_y) \Longleftrightarrow L_x \leq L_y \wedge K_x \subseteq K_y$$

miteinander vergleichen. Während der Sicherheitsüberprüfung des Zugriffsrechtes $p \in P$ eines Subjektes s mit der Clearance SC_s auf eine Objekt o mit der Classification SC_o innerhalb des Zugriffskontrollmechanismus gilt:

- Simple-Security-Regel (read down): $p = read - only \wedge SC_s \geq SC_o$
- *-Eigenschaft (write up): $(p = read - write \vee p = append) \wedge SC_s \leq SC_o$
- Confidentiality-Confinement-Regel: $(p = execute \vee p = control) \wedge SC_{S_1} \geq SC_{S_2}$
- Confidentiality-Creator-Regel: $p = create \wedge SC_s \geq SC_o$

Sowie die Regeln für die Integritätsklassen als auch die Regeln für die Sicherheitsklassen sind in den Sicherheitsrichtlinien definiert. Weitere objektspezifische Regeln sind in den Objektrichtlinien definierbar. Dort fließen unter anderem Regelungen basierend auf den Nutzungsrichtlinien und der Data Governance ein. Die Zugriffskontrollrichtlinie ist die Schnittstelle zwischen dem Zugriffskontrollmechanismus und den Sicherheitsartfakten. Über diese Schnittstelle werden die entsprechenden Regeln für die gestellte Zugriffsanfrage zusammengestellt und an den Zugriffskontrollmechanismus übergeben.

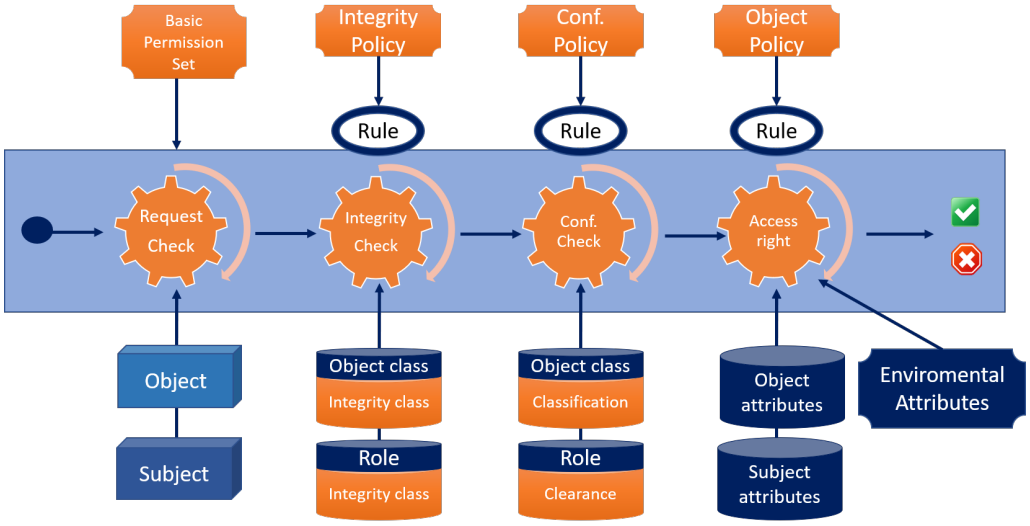


Abbildung 5.2: Mehrfaktorzugriffsregelung im Zugriffskontrollmechanismus (Conf. = Confidentiality)

5.3.3 Zugriffskontrollmechanismus

Abbildung 5.2 zeigt die vier Stufen, die eine Zugriffsanfrage innerhalb des Zugriffskontrollmechanismus erfolgreich durchlaufen muss. Zuerst erfolgt eine Überprüfung der Anfrage und ihrer Parameter. Neben der Identität des Subjektes und der Existenz des Objektes wird das angefragte Zugriffsrecht mit den erlaubten Zugriffsrechten abgeglichen. Danach gelangt die Anfrage zur Integritätsüberprüfung. Dabei liefern die Integritätsrichtlinien die dem angefragten Zugriffsrecht entsprechenden Regeln zur Überprüfung der Integritätsklassen. Neben den Regeln liest der Mechanismus auch die benötigten Attribute aus. Im Falle des Integritätschecks benötigt der Mechanismus die Integritätsklassen des Objektes und des Subjektes, die an die Objektklasse bzw. die Subjektrolle gebunden sind. Entsprechend dem Integritäts-Check läuft ebenfalls der Confidentiality-Check anhand der Vertraulichkeitsrichtlinien und der Classification des Objektes und der Clearance des Subjektes. Die dritte Überprüfung zieht objektspezifische Regeln hinzu. Basierend auf den erhaltenen Regeln liest der Zugriffskontrollmechanismus die benötigten Attribute aus. Dies können Subjekt-, Objekt- sowie Umgebungsattribute sein.

5.4 Evaluation der Anforderungen

Die Erweiterung der attribut-basierten Zugriffskontrolle mit dem Bell-LaPadula, dem Biba und dem rollen-basierten Modell zielte auf die Erfüllung aller aufgestellten Anforderungen für ein Zugriffskontrollmodell für Data Lakes ab. In diesem Abschnitt ist das erweiterte Zugriffskontrollmodell gegen die gegebenen Anforderungen evaluiert.

A1 Anwendungsbasiertes Zugriffskontrollmodell:

✓ [$K1$] Das Zugriffskontrollmodell ist hinsichtlich Data Lakes zur Umsetzung als ein Service konzipiert. Anfragen an das Data Lake laufen über diesen Zugriffskontrollservice.

- ✓ [$K2$] Durch die Einbeziehung unterschiedlicher Attribute bei der Zugriffsrechtevergabe stellt die Verwendung des erweiterten Zugriffskontrollmodells eine genauere und vielfältig modellierbare Zugriffskontrolle im Vergleich zu der vorhandenen Zugriffskontrolle auf darunterliegender Ebenen da.

A2 Feingranulare, objektspezifische Zugriffsrechte:

- ✓ [$K1$] Die Zugriffsrechtevergabe kann feingranular auf Objektebene erfolgen. Die aufgestellten Regeln innerhalb der vorliegenden Richtlinien beziehen sich auf Attribute anhand derer die Zugriffsrechte eines Subjektes auf ein Objekt ermittelt werden können.
- ✓ [$K2$] Unter den Sicherheitsartefakten befindet sich eine Auflistung der erlaubten Zugriffsrechte auf das System. Das erweiterte Zugriffskontrollkonzept definiert grundlegende Zugriffsrechte, die in der Regel innerhalb eines Data Lakes benötigt werden.
- ✓ [$K3$] Die Definition von individuellen Zugriffsrechten kann auf zweierlei Arten erfolgen. Einerseits ist es möglich durch Regeln eine Kombination von der festgelegten Zugriffsrechte zu bewirken. Auf der anderen Seite kann das Zugriffskontrollkonzept durch eigene Zugriffsregeln im *Basic Permission Set* erweitert werden. Bei der Definition eigener Regeln ist die Anpassung der einzelnen Richtlinien zum Integritäts- und Sicherheits-Check sowie der objekt-spezifischen Richtlinien notwendig.

A3 Bündelung zusammengehöriger Privilegien für Gruppen:

- ✓ [$K1$] Das Zugriffskontrollmodell setzt voraus, das jedes Subjekt mindestens einer Subjektrolle zugeordnet und dort aktiv sein muss. Ohne eine vorhandene Subjektrolle erhält das Subjekt über die Zugriffskontrolle keine Zugriffsrechte.
- ✓ [$K2$] Subjekten ist es erlaubt, mehreren Rollen zugeordnet zu sein. Dies ist in den Subjektattributen hinterlegt.
- ✓ [$K3$] Zum Zugriffszeitpunkt prüft der Zugriffskontrollmechanismus die Eigenschaften der aktiven Rolle des Subjektes ab. Ein Subjekt darf zwar mehreren Rollen zugeordnet, jedoch nur in einer Rolle aktiv sein.
- ✓ [$K4$] Jedes Objekt wird einer Objektklasse automatisch bei der Erstellung über den create-Befehl zugeordnet. Ebenso geschieht die Objektklassenzuordnung mit der Anpassung des Metadatenmodells innerhalb des Data Wrangling Prozesses.

A4 Sicherstellung der Geheimhaltung (Confidentiality):

- ✓ [$K1$] Über die Angehörigkeit zu Subjektrollen bzw. Objektklassen gehören sowohl Subjekte als auch Objekte Sicherheitsklassen an.
- ✓ [$K2$] Die Regeln zur Zugriffsrechtevergabe bzw. -beschränkung sind aus dem Bell-LaPadula-Modell übernommen und an das erweiterte Zugriffskontrollmodell angepasst. Innerhalb des Zugriffskontrollmechanismus setzt der Confidentialiy-Check die Einhaltung der Regeln durch.

A5 Sicherstellung der Objektintegrität (Integrity):

- ✓ [$K1$] Neben den Sicherheitsklassen sind alle Rollen und Objektklassen einer Integritätsklasse zugeordnet. Auf diese Weise gehört jedes Objekt und Subjekt einer Integritätsklasse an.

- ✓ [$K2$] Innerhalb der Integritätsrichtlinien sind die vom Zugriffskontrollmechanismus einzuhaltenden Regeln für den Integritäts-Check definiert.

A6 Regelung der initialen Rechtevergabe:

- ✓ [$K1$] Durch die Vorgabe, dass ein Objekt einer Objektklasse angehören muss, sind durch den Integritäts- und den Sicherheits-Check initiale Zugriffsrechte definiert. Darüber hinaus erlauben objekt-spezifische Regeln, die an bestimmte Objektattribute gebunden sind, eine Definition von initialen Zugriffsrechten auf Objekte.
- ✓ [$K2$] Über die in den Sicherheitsartefakten vorliegenden Richtlinien können individuelle Zugriffsregeln definiert sein. Regeln bestehen aus zu erfüllenden Bedingungen, die an unterschiedliche Attribute geknüpft sein können. So erfolgt die Zugriffsrechtevergabe nicht statisch über eine Zugriffskontrollmatrix sondern zur Anfragezeit über die Überprüfung der vorhandenen Regeln.

A7 Dynamische Anpassbarkeit der Rechtevergabe:

- ✓ [$K1$] Basierend auf dem attribut-basierten Zugriffskontrollmodell hängen die Zugriffsrechte von der Auswertung definierter Bedingungen innerhalb der Regeln aus den Richtlinien ab. Somit sind keine Zugriffsrechte statisch im System hinterlegt.
- ✓ [$K2$] Bei der Zugriffsrechtevergabe spielen unterschiedliche Attribute innerhalb der verschiedenen Überprüfungen des Zugriffskontrollmechanismus eine Rolle. Innerhalb den Regeln der Richtlinien sind die Abhängigkeiten von Zugriffen eines Subjektes auf ein Objekt an Attribute definiert.
- ✓ [$K3$] Die Zugriffsrechtevergabe ist dynamisch über die Richtlinien der Sicherheitsartefakte anpassbar. Des weiteren erlaubt das Zugriffsrecht control eine Rechteweitergabe an andere Subjekte.
- ✓ [$K4$] Eine temporäre Zugriffsfreigabe ist über die Zuordnung einer Rolle, mithilfe einer an die Zeit geknüpfte Regel innerhalb der Richtlinie oder über die Rechteweitergabe per control möglich.

A8 Definition komplexer Zugriffsbeschränkungen:

- ✓ [$K1$] Über die Zugriffsrichtlinien können sowohl die Vergabe von Zugriffsrechten, als auch eine Beschränkung der Zugriffsrechte modelliert sein.
- ✓ [$K2$] Über die Definition von Rollen ist es möglich, Zugriffsrechte aufgabenorientiert an eine Rolle zu binden. Dazu sind die Bedingungen der Zugriffsrechtevergabe innerhalb der Richtlinien an die Mitgliedschaft eines Subjektes zu einer Rolle gebunden.

Das aufgestellte, hybride Zugriffskontrollmodell deckt alle Anforderungen ab. Durch die fehlende Festlegung der Zugriffsrechte innerhalb einer Zugriffskontrollmatrix ist das Zugriffskontrollmodell dynamisch und flexibel modellierbar. Durch die Evaluation der bestehende Zugriffskontrollmodelle konnte deren Vorteile und Anforderungserfüllung herausgearbeitet und in einem hybriden Modell kombiniert werden. Auf diese Weise sind alle Anforderungen erfüllbar.

5.5 Fazit

Ein Data Lake stellt durch seine Flexibilität und Datenvielfalt gesonderte Anforderungen an ein Zugriffkontrollkonzept. Neben der Sicherstellung der Geheimhaltung und der Objektintegrität ist vor allem eine dynamische Rechtevergabe und die Definition komplexer Zugriffsbeschränkungen für ein Data Lake wichtig. Basierend auf den Charakteristika eines Data Lakes und der Analyse bestehender Zugriffskontrollmodelle in den vorigen Kapiteln konnten Anforderungen für das Data Lake ausformuliert werden. Die in Kapitel 4 vorgestellten Zugriffskontrollmodelle erfüllten lediglich einige Anforderungen. Keines der Modelle konnte allen Anforderungen gerecht werden. Aus der Evaluation der bestehenden Modelle lies sich ein hybrider Ansatz entwickeln. Dieser basierte auf der logischen Architektur des attribut-basierten Zugriffskontrollmodells. Das ausgearbeitete Zugriffskontrollmodell verwendet Sicherheitsartefakte als Basis für die Zugriffskontrollrichtlinie. Innerhalb der Sicherheitsartefakte sind die Einflüsse anderer Zugriffskontrollmodelle sichtbar. Das Bell-LaPadula-Model und das Biba Modell spiegeln sich in den Integritäts- und Sicherheitsklassen wieder. Diese werden mithilfe von Rollen und Objektklassen an Subjekte und Objekte gebunden. Die Evaluation der gegebenen Anforderungen ergab eine vollständige Abdeckung aller Anforderungen durch die Erfüllung aller aufgestellten Kriterien. Eine exemplarische Anwendung des erweiterten Zugriffskontrollmodells ist im nächsten Kapitel motiviert.

6 Anwendungsfall: Bereich E-Health

Neben den bekannten Big Data Beispielen von IoT (Internet of Things) und Industrie 4.0, ist die Verwendung von Big Data Lösungen auch für den E-Health-Bereich interessant. Im Rahmen seiner Digitalisierungspläne plant beispielsweise der aktuelle Gesundheitsministers Jens Spahn Patientenakten elektronisch (ePA) für Ärzte und Patienten bereitzustellen [May18]. Die ePA soll elektronische Fallakten (eFA) oder elektrische Medikationspläne (eMP) enthalten und jederzeit von Patienten, Apothekern oder Ärzten eingesehen werden können [Gem18; Wil18]. So scheint der Gesundheitsminister alle analogen Artefakte abschaffen zu wollen und einen digitalen Austausch zwischen Patienten und Heilberuflern zu fördern. Da Daten für das Gesundheitswesen weit umfangreicher als Patientenakten sind, seien es Forschungsdaten, Pharmadaten oder persönliche Daten des Patienten, fällt eine massive Datenmenge unter die Digitalisierungspläne des Ministers. Eine Idee wäre, dabei auf das Konzept des Data Lake umzustellen und die Datensilos einzelner Parteien zusammenzulegen.

Gerade bei der Feststellung eines Krankheitsbildes eines Patienten und der Ursachenforschung sind viele unterschiedliche Akteure beteiligt. Da zu den Akteuren nicht nur Ärzte sondern zum Beispiel auch Pflegepersonal, Angehörige, etc. zählen, besteht unterschiedlicher Informationsbedarf. Aufgrund der Masse unterschiedlicher Interessen der Beteiligten und der unterschiedlichen Rechte zum Datenzugriff eignet sich ein Anwendungsfall aus dem Bereich E-Health hervorragend zur Anwendung und Modellierung des erweiterten Zugriffskontrollmodells.

Die Notwendigkeit eines modellierbaren Zugriffskontrollmodells auf ein Data Lake ist nun anhand eines realistischen Anwendungsfalls in diesem Kapitel motiviert. Zuerst beschäftigt sich das Kapitel mit der Heranführung der Problematik von getrennten Speicherlösungen. Nachdem ein Problemlösungsprozess eine Verwendung eines Data Lakes im Bereich E-Health motiviert, sind die Kernpunkte des Anwendungsfalls für das erweiterte Zugriffskontrollmodell herausgearbeitet und modelliert. Dabei helfen die Ergebnisse des Problemlösungsprozess zur Feststellung benötigter Zugriffsregeln.

6.1 Problemlösungsprozesse

Probleme treten in allen Bereichen auf. Aus diesem Grund liegen im Qualitätsmanagement unterschiedliche Strategien zur systematischen Problemlösung vor. Diese Problemlösungsprozesse beschreiben einen schrittweisen, dokumentierbaren Ablauf zur Ursachenanalyse und Problembehebung. Die Basis aller Vorgehensmodelle innerhalb des Qualitätsmanagement ist der PDCA-Zyklus (Abbildung 6.1) nach W.E.Deming [JSW17, S.8]. Der PDCA-Zyklus ist ein kontinuierlicher Prozess zur ständigen Verbesserung [HM11, S.81]. Nach der Planungsphase erfolgt die Umsetzungsphase. Diese wird danach beurteilt (check) und mögliche Verbesserungspunkte umgesetzts (act). Danach beginnt der Zyklus von neuem.

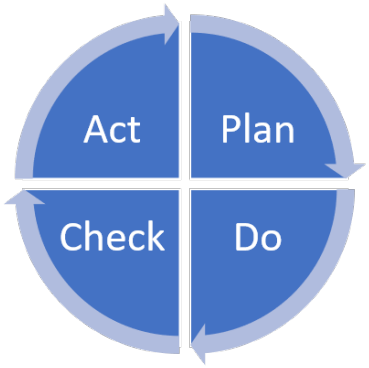


Abbildung 6.1: PDCA-Zyklus

Dieser sehr allgemein gefasste Zyklus zieht sich durch das ganze Qualitätsmanagement bis hin zum Reklamationsmanagement. Das Reklamationsmanagement befasst sich mit dem Umgang von Reklamationen eines Produktes oder Prozesses. Es nutzt Problemlösungsprozesse um gezielt Probleme zu analysieren und zu lösen. Unter den Problemlösungsprozessen hat sich in der Industrie der 8D-Prozess durchgesetzt. Der Verband der Automobilindustrie (VDA) erläutert in einem separaten Kapitel die einzelnen Schritte des 8D-Prozesse zur Anwendung im Automobilbereich [Aut10]. Ein ähnliches Vorgehensmodell, das 7STEP, findet in der Industrie ebenfalls Anwendung. Dieses unterscheidet sich inhaltlich kaum von dem 8D-Prozess [JSW17, S.13], sodass im Folgenden lediglich der 8D-Prozess erklärt und angewandt wird.

8D stehen für acht Disziplinen die im Prozess durchlaufen werden [JSW17, S. 9]. Die acht Schritte des Vorgehensmodells dienen als Leitfaden zur nachhaltigen und systematischen Problembeseitigung. Der 8D-Prozess ist ein sehr ausführlicher Prozess, der über einen längeren Zeitraum umgesetzt wird. Aus diesem Grund sieht einer der Schritte Sofortmaßnahmen zur kurzzeitigen Behebung vor. Für kleinere, interne Probleme kann statt des 8D-Prozesses der 4D-Prozess angeweandt werden. Abbildung 6.2 zeigt die Abfolge der einzelnen Disziplinen.

Disziplin 1: Zusammenstellen des Teams Im ersten Schritt geht es um die Bestimmung eines bestmöglichen Teams den Problemlösungsprozess zusammenzustellen. Dabei sollten alle vom Problem und der späteren Problemlösung betroffenen Parteien eine Interessensvertretung im Team haben. Jedes Teammitglied muss über ausreichend Kenntnisse verfügen [JSW17, S.19] und bekommt eine eindeutige Rolle zugewiesen [Aut10]. Der Teamleiter ist für die korrekte Durchführung des Prozesses verantwortlich. In diesem Schritt erfolgt ebenfalls die Einigung auf Teamregeln um eine gute Zusammenarbeit zu ermöglichen.

Disziplin 2: Problembeschreibung In diesem Schritt setzt sich das Team zusammen, um das Problem vollständig zu definieren und abzugrenzen. Zur Erfassung des Problems können andere Methodiken, wie zum Beispiel die 5-Why-Methode aus der Fehler-Ursachen-Analyse herangezogen werden [JSW17, S.20]. Ziel ist es alle Teammitglieder auf den gleichen Wissenstand zu bringen und

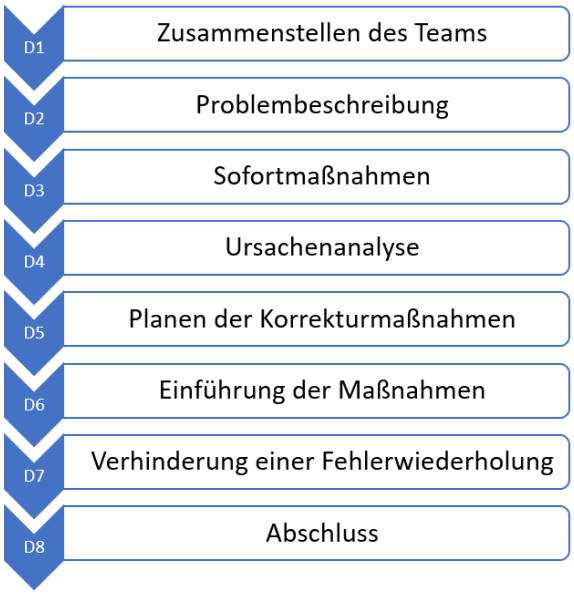


Abbildung 6.2: 8D-Prozess

eine einheitliches Problemverständnis zu erschaffen. Ergebnis dieses Schrittes ist die Dokumentation des Unterschieds zwischen dem vorhandenen Ist- und dem gewünschten Soll-Zustandes. Darüber hinaus sollen die Auswirkungen des Problems sowie die betroffenen Teile analysiert werden.

Disziplin 3: Sofortmaßnahmen Eine Fehlerursachenanalyse kann sich langwierig gestalten, sodass zuerst Sofortmaßnahmen zur Problembehebung ergriffen werden. Die Sofortmaßnahmen sollen den Kunden vor Fehlern und Schaden schützen. Neben Rückrufaktionen bzw. Austausch von Produkten, sollten die weitere Auslieferung gestoppt werden. Ergebnis dieses Schrittes ist die Entfernung des fehlerhaften Teils aus dem gesamten Umlauf und Betroffenen einen Ausgleich anzubieten [Aut10].

Disziplin 4: Ursachenanalyse Nachdem durch die Sofortmaßnahmen das Problem erstmal behoben ist, erfolgt die Ursachenanalyse. Mit der Ermittlung und der darauffolgenden Behebung der Ursachen soll eine langfristige und nachhaltige Lösung zur Problemvermeidung gefunden werden. In diesem Schritt stellt das Team mögliche Problemursachen fest. Durch die Aufstellung der Ursachen-Wirkung-Zusammenhänge kann die Kernursache aus den möglichen Ursachen herausgefiltert werden.

Disziplin 5: Planen der Korrekturmaßnahmen Im fünften Schritt geht es nun um die Festlegung der Maßnahmen zur nachhaltigen Beseitigung der Kernursache [JSW17, S.28f.]. Das Team sammelt alle möglichen Korrekturmaßnahmen. Mithilfe einer Nutzwertanalyse oder der Fehlermöglichkeits- und Einflussanalyse (FMEA) können die besten Korrekturmaßnahmen ermittelt werden. Nach der Auswahl erfolgt eine Wirksamkeitsprüfung. Am Ende des Schrittes sollte sich das Team über die nachhaltige Problemursachenbehebung mithilfe der Korrekturmaßnahmen sicher sein.

Disziplin 6: Einführen der Maßnahmen Nachdem Schritt 5 nachweist, dass die Korrekturmaßnahmen die Ursachen erfolgreich beheben, leitet dieser Schritt die Umsetzung der Maßnahmen ein. Die Maßnahmen sollen organisatorisch im Prozessablauf verankert werden. Dadurch sind die Korrekturmaßnahmen nachhaltig und dienen zur Verhinderung der Fehlerwiederholung. Mit der Einführung der Korrekturmaßnahmen können die Sofortmaßnahmen zur Problembehebung aufgehoben werden. Ergebnis dieser Disziplin ist die Umsetzung der Korrekturmaßnahmen und ihre langfristige Beobachtung [Aut10].

Disziplin 7: Verhinderung der Fehlerwiederholung Dieser Schritt soll ein Wiederauftreten des Problems vorbeugen. Dazu zählt vor allem die Aufklärung der Prozessbeteiligten über die Fehlerursache sowie die Ausstellung von Empfehlungen und Treffen von Vorbeugungsmaßnahmen für andere Systeme oder Produkte [JSW17, S.32]. Es geht in diesem Schritt vor allem um Wissensaustausch über die gewonnen Erkenntnisse.

Disziplin 8: Abschluss Zum Schluss des Problemlösungsprozesses überprüft der Teamleiter die Umsetzung der Maßnahmen um einen formalen Abschluss zu finden. Nach der Komplettierung aller Dokumente ist es in diesem Schritt wichtig, den Teamerfolgs als Annerkennung der erbrachten Leistungen zu würdigen [Aut10]. Auf diese Weise bleibt ein positiver Eindruck bei den Teammitgliedern hängen, der sich auf zukünftige Problemlösungsprozesse auswirkt.

Der 8D-Prozess bietet eine schrittweise Anleitung zur Problemanalyse und -behebung. Innerhalb der einzelnen Disziplinen werden weitere Methoden zum Ergebnisgewinn eingesetzt. Mit erfolgreichem Durchlaufen aller Schritte ist das Problem nachhaltig behoben. Der Problemlösungsprozess ist vielseitig einsetzbar. Innerhalb des E-Health Anwendungsfalls dient er beispielsweise zur Feststellung der Notwendigkeit eines Data Lakes mit einem Zugriffskontrollmodell. Der nächste Abschnitt beschäftigt sich mit einem Problem aus dem medizinischen Bereich, das als Grundlage für die Durchführung des 8D-Prozesses herangezogen wird.

6.2 Vorstellung des Anwendungsfalls anhand des 8D-Prozesses

Der 2001 bekanntgewordene Lipobay-Skandal kostete mindestens 104 Menschen das Leben [Alb02]. Das Medikament Lipobay ist ein Cholesterinsenker und sollte bei betroffene Patienten mit einem hohen Cholesterinspiegel zur Vorbeugung von Durchblutungsstörungen und Herz-Kreislauf-Erkrangungen eingesetzt werden. Cholesterinsenker aus der Wirkstoffgruppe der Statine sind dafür bekannt in seltenen Fällen Muskelschwäche zu verursachen. Im Lipobay-Skandal kam es nun zu einem gehäuften Auftreten von Muskelschwächen bei Patienten, einige starben infolgedessen an akutem Nierenversagen. Nach genaueren Untersuchungen stellte sich im Nachhinein heraus, dass das Medikament vor allem in Kombination mit anderen Medikamenten zur Zerstörung von Muskelgewebe führte. Dieser Skandal zeigt, dass das Zusammenwirken von unterschiedlichen Medikamenten zu unerwünschten Nebenwirkungen mit Todesfolge führen kann. Um solche Fälle besser untersuchen zu können und schnellst möglich die Ursache herauszufinden, müssen Patientendaten ausgewertet werden.

Auf diesem Lipobay-Skandal aufbauend kann ein Anwendungsfall für einen Pharmakonzern

hergeleitet werden, der die Auswirkungen des Medikamentes bei Einnahme in Kombination mit einem anderen Präparat eines anderen Konzerns untersuchen möchte. Dabei verwendet der Konzern die 8D-Methode um eine strukturierte Untersuchung durchzuführen. Innerhalb des Prozesses wird die Notwendigkeit eines Zugriffskontrollmodells für Data Lakes deutlich:

D1 Das Team für den Problemlösungsprozess setzt sich aus den Fachvertretern der Pharmakonzerne, aus Fachärzten der zu behandlenden Krankheiten und einem Vertreter des Bundesinstituts für Arzneimittel und Medizinprodukte (BfArM).

D2 Die Problembeschreibung fasst die Nebenwirkungen bei kombinierter Einnahmen zweier Medikamente, deren Auftretenswahrscheinlichkeit und deren Verbreitung zusammen.

D3 Bei der Sofortmaßnahme handelt es sich um die Information von der Ärzte über die Nebenwirkung bei der kombinierten Einnahme der Medikamente. Schlimmstenfalls sollte über eine Rückrufaktion nachgedacht werden.

D4 In diesem Schritt erfolgt die Feststellung der Fehlerursache. Zur Untersuchung auf Gemeinsamkeiten innerhalb der Patientengruppen, benötigt das Team Zugriff auf alle relevanten Daten. Im medizinischen Bereich legt jeder Arzt die Patientendaten in einem eigenen System ab. Dieses ist vor externen Zugriffen geschützt. Für die Analyse der Ursache benötigt das Team einen zusammengeführten Datensatz zu den Patienten. Dies ist bei den vorhandenen Datensilos nicht möglich.

D5 Angenommen die Fehlerursache konnte trotz fehlender vollständiger Patientenunterlagen ermittelt werden, beginnt in diesem Schritt die Planung der Abstellmaßnahmen. Für die Pharmakonzerne bedeutet dies, die Zusammenstellung der Medikamente zu ändern und Medikamentenstudien zu planen.

D6 Das Einführen der Abstellmaßnahmen ist in diesem Anwendungsfall die Herstellung und Markteinführung der neuen Medikamente. Des Weiteren wird die Produktion der alten Medikamente gestoppt.

D7 Die Fehlerwiederholung kann durch Überwachung der Nebenwirkungen und Medikamententest verhindert werden. Für die Überwachung der Patienten fehlt widerrum der legitime Zugriff auf die Patientenakte.

D7 Pro Forma erfolgt zum Abschluss des Problemlösungsprozesses die Würdigung der Teamleistung.

Innerhalb des vierten und siebten Schrittes benötigte das Team Daten zur Feststellung der Ursache und der Überwachung zur Vermeidung von Fehlerwiederholungen. Diese Daten befinden sich lokal in den Speichersystemen der behandelnden Ärzte, sind also nicht global erreichbar. Die auftretenden Zugriffsprobleme motivieren zur Verwendung eines Data Lakes im medizinischen Bereich. Ein Data Lake würde alle medizinischen Daten eines Patienten bündeln und einen leichteren Datenaustausch ermöglichen. Zur Absicherung vor unberechtigten Zugriffe muss das Data Lake durch ein Zugriffskontrollmodell geschützt werden. Der nächste Abschnitt beschreibt die Modellierung des aufgestellten Zugriffskontrollmodells für Data Lakes anhand des Anwendungsfalls.

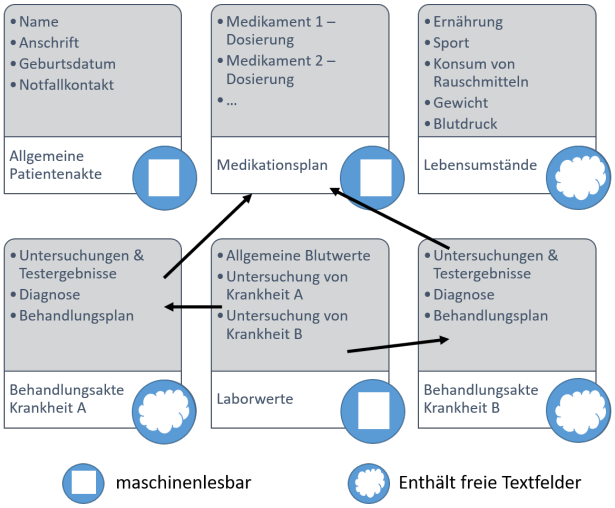


Abbildung 6.3: Objektklassen im Data Lake

6.3 Modellierung des erweiterten Zugriffskontrollmodells

Zur Veranschaulichung der Umsetzung des erarbeiteten Zugriffskontrollmodells für Data Lakes dient ein Anwendungsfall aus dem E-Health-Bereich.

Objekte und Objektklassen Zur Modellierung wird auf einen begrenzten Datensatz pro Patient zurückgegriffen. Abbildung 6.3 zeigt die im Data Lake vorliegenden Objektklassen anhand derer die Datensätze (Objekte) eingeordnet werden. Die Allgemeine Patientenakte, der Medikamentationsplan, sowie die Laborwerte liegen in maschinenlesbarer Form vor. Sie enthalten lediglich definierte Felder die miteinander verglichen werden können. Dahingegen verfügen die Lebensumstände und die unterschiedlichen Behandlungsakten freie Textfelder. Inhalte aus diesen Objekten können nicht automatisch verarbeitet werden. Die unterschiedlichen Datenobjekte zeigen die Datenvielfalt auf, die in einem Data Lake vorkommen kann.

Subjekte und Subjektrollen Auf das Data Lake greifen unterschiedliche Instanzen mit unterschiedlichen Interessen zu. Da es vor allem um den Schutz der Patientendaten vor unberechtigten Zugriffen geht, müssen die erlaubten Rechte der einzelnen Subjekte definiert werden. Zur Modellierung unterschiedlicher Sicherheits- und Integritätsstufen werden für den Anwendungsfall folgende Subjektrollen mit ihren Interessen verwendet:

- Patient: eigene Daten einsehen, persönliche Daten ändern, anonyme Vergleichspersonen einsehen
- Ärzte: Patientendaten für die Behandlung einsehen, Medikamentenpläne ändern, Testergebnisse eintragen (Laborwerte)
- Pflegekräfte: Medikationspläne auslesen, Gesundheitszustand beobachten/eintragen

Objekte: / Subjekte:	PA	LU	M	BA A	LB	BA B
Patient	write	write	read	read	read	read
Pflegekräfte	read	append	read	-	-	-
Apotheke	-	-	read	-	-	
Hausarzt	write	write	write	write	append	append
Facharzt	read	read	write	-	read	write
Pharmakonzern	stat.	stat.	stat.	stat.	stat.	stat.
Krankenversicherung	read	-	read	part.	-	part.
Labor	-	-	-	-	create	-

Tabelle 6.1: Zugriffsrechte im Anwendungsfall
PA=Patientenakte, LU = Lebensumstände, M=Medikationsplan,
BA A= Behandlungsakte A, LB=Laborwerte, BA B = Behandlungsakte B
stat. = statistical read, part. = partial read

- Krankenversicherung: Patientendaten für Bezahlung einsehen, Statistiken erstellen, Versicherungskonditionen vorhersagen
- Pharmaindustrie: Wirkung der Medikamente, Nebenwirkungen ableiten, Bedarf an neuartigen Medikamenten feststellen, Verbreitung der Medikamente verfolgen, Statistiken erheben
- Labor: Laborwerte eintragen

Zugriffsrechte Zur Erfüllung ihrer Aufgaben benötigten die unterschiedlichen Subjekte verschiedene Zugriffsrechte auf die Objekte. Die Zugriffskmatrix in Tabelle 6.1 listet die einzelnen benötigten Zugriffsrechte entsprechend den Subjektrollen und Objektklassen übersichtlich auf. Neben Lese- und Schreibrechten, werden einige individuelle Rechte benötigt. Die Pharmakonzerne brauchen beispielsweise lediglich statistische Daten und keinerlei Identitätsdaten. Die Daten des Patienten dürfen für Pharmakonzerne nur in anonymisierter Form zur Verfügung stehen. Des Weiteren benötigt die Krankenversicherung eines Patienten keinen kompletten, sondern einen partiellen Zugriff auf die Behandlungsakten. Lediglich die kostenpflichtigen Behandlungen samt ihren Diagnosen sind relevant für die Krankenkasse. Über diese speziellen Zugriffsrechte wird das unbegrenzte Datensammeln von externen Institutionen innerhalb des Data Lakes verhindert und die Daten des Patienten geschützt. Subjekte erhalten lediglich die zur Aufgabenerfüllung notwendigen Objektzugriffe. Die Zugriffrechte des erweiterten Zugriffskontrollmodells bleiben erhalten, sodass $P = \{create, read-only, read-write, append, execute, control\}$. Die Modellierung der speziellen Zugriffsrechte partial read und statistical read erfolgt über die Objektrichtlinien. Da beide Zugriffsrechte im Wesentlichen einen eingeschränkten Lesezugriff darstellen, orientiert sich der Integriäts-Check und Vertraulichkeits-Check an den Regeln für Lesezugriffe.

Integritätsklassen Der Anwendungsfall enthält sowohl vertrauenswürdige als auch nichtvertrauenswürdige Subjekte. Im wesentlichen lassen sie sich in drei Kategorien, den Fachpersonal, Beteiligte und Externe unterteilen. Das Fachpersonal bündelt die Ärzte und die Labore.

Zu den Beteiligten gehören die Patienten und Pflegekräfte, zu den Externen zählen Apothekenmitarbeiter, Krankenversicherung und die Pharmaindustrie. Neben der Vertraulichkeitsstufe der Subjekte werden auch Objekte in Integritätsklassen eingeteilt. Der Anwedungsfall beinhaltete sensible medizinische Informationen sowie Zusatzinformationen zu dem Patienten. Aus den Objekt- und Subjektunterscheidungen in Bezug auf die Integrität ergeben sich drei Integritätsklassen $I = \{Crucial(C) \geq Essential(E) \geq Unknown(U)\}$.

Sicherheitsklassen Bei den Geheimhaltungsstufen verhält es sich ähnlich den Integritätsklassen. Medizinische Informationen wie die Laborwerte und Behandlungsakten unterstehend der höchsten Geheimhaltungsstufen. Danach folgen die persönlichen Daten und die Lebensumstände eines Patienten. Durch den Zugriff der Pflegekräfte und der Apotheke gehören die Laborwerte einer anderen Geheimhaltungsstufe an. Daraus ergeben sich die Sicherheitslevel $L = \{Sensitive \geq Private \geq Trusted \geq Public\}$. Bei der Sicherheitskategorie handelt es sich für den gesamten Anwendungsfall um dieselbe, da alle Daten im Data Lake zum Anwendungsfall gehören. Aus diesem Grund werden die Sicherheitskategorien bei der folgenden Modellierung außer acht gelassen.

Objektrichtlinien Die Integritäts- und Vertraulichkeitsrichtlinien können aus dem erweiterten Zugriffskontrollmodell ohne Modifikation übernommen werden. Zur Modellierung des Zugriffskontrollmodells für den Anwendungsfall benötigt es lediglich die Definition der Objektrichtlinien. Dieser Abschnitt stellt exemplarisch ein paar der modellierte Objektrichtlinien für die Objektklassen Lebensumstände (LU) und Behandlungsakten (BA). Die Regeln bestehen aus der Formulierung der grundlegenden Bedingungen und der erlaubten Zugriffsrechte die bei Erfüllung der Bedingungen gewährt werden. Zur Erklärung, R2.1 setzt die Objektklasse BA also Behandlungsakte sowie die Rolle eines Haus- oder Facharztes für eine Append-Zugriffsrecht voraus. Erfüllt die Anfrage darüber hinaus R2.2 so erhält das zugreifende Subjekte volle Schreibrechte auf das Objekt. Auf diese Weise können mehrere Regeln miteinander kombiniert und so komplexe Zugriffsbeschränkungen realisiert werden. In R1.4 wird das Objektattribut statistics vorrausgesetzt und somit ein statistischer Lesezugriff erlaubt. In R1.2 ist eine Umgebungsvariable zur Zugriffsrechtevergabe vorausgesetzt.

R1.1 $objectclass = LU \land role = (Patient \lor Hausarzt) : p = write$

R1.2 $objectclass = LU \land role = Pflegekraft \land subject\_attr.location = obj\_attr.home : p = append$

R1.3 $objectclass = LU \land role = Facharzt : p = read$

R1.4 $objectclass = LU \land role = Pharma \land object_{a}ttr.statistics = true : p = read(statistical)$

R2.1 $objectclass = BA \land role = (Hausarzt \lor Facharzt) : p = append$

R2.2 $R2.1 \land object\_attr.owner = subj\_attr.id : p = write$

R2.3 $objectclass = BA \land role = Krankenversicherung \land object\_attr.selectableFields.count > 0 : p = read$

Die vorgestellten Regeln für die Objektrichtlinien sind lediglich ein Auszug aus den nötigen Regeln, die für das Zugriffskontrollmodell aufgestellt werden müssen. Sie zeigen exemplarisch, wie Regeln zur Zugriffskontrolle definiert werden können. Nach der Zusammenstellung der Subjektrollen, Objektklassen, Sicherheits- und Integritätsklassen sowie der Objektrichtlinien ist eine Implementierung des Zugriffskontrollmodells für den Anwendungsfall möglich.

7 Implementierung

Im vorigen Kapitel wurde ein Anwendungsfall für die Modellierung eines Zugriffskontrollmodells für Data Lakes vorgestellt. Die Implementierung des Zugriffskontrollmodells erfolgte basierend auf der vorgestellten Modellierung des Anwendungsfalls. Dieses Kapitel dokumentiert das implementierte Zugriffskontrollmodell. Zuerst befasst sich das Kapitel mit der Architektur des verwendeten Data Lakes für das ein Zugriffskontrollservice entwickelt wurde. Danach werden die wesentlichen Komponenten des Services samt Codebeispielen vorgestellt.

7.1 Architektur des Data Lakes

Oftmals ist das Konzept des Data Lake in Zusammenhang mit Hadoop erklärt. Aus diesem Grund ist für die Implementierung des Anwendungsfalls ein Hadoop Data Lake verwendet worden. Hadoop ist eine Softwarebibliothek von Apache rund um den MapReduce Algorithmus zur Verarbeitung von Big Data [Fou18a]. Seit 2008 startete das OpenSource-Projekt von Doug Cutting als Top-Level-Projekt von Apache richtig durch und wird mittlerweile von Softwaregiganten wie Google und Facebook verwendet [Lit16].Das auf Java basierende Software Framework ist für skalierbare, verteilte Systeme vorgesehen und setzt sich aus vier zentralen Komponenten zusammen [Fou18a; Lit16]:

- Das Hadoop Common bündelt die Grundfunktionalitäten und Tools für das restliche Softwareframework. Es beinhaltet die Basisauthentifizierung über HTTP oder eine native Bibliothek für die Checksummen, Kompressionen oder das Cache-Management. Darüber hinaus bietet es Schnittstellen für die anderen Softwarekomponenten.[Fou18b; Lit16]
- Das Hadoop Distributed File System, kurz HDFS, ist ein verteiltes Dateisystem zur Speicherung von großen Datenmengen [Lit16].
- Aus dem MapReduce-Algorithmus, einem Algorithmus zur verteilten Verarbeitung großer Datenmengen, entstand das Hadoop-Projekt. Der Algorithmus wurde 2003 von Google veröffentlicht und nutzt vor allem die Parallelisierung von Aufgaben innerhalb verschiedener Rechenmaschinen zur schnellen Bearbeitung. MapReduce kann in vielen Gebieten rund um Big Data zum Einsatz kommen, so zum Beispiel beim Data Mining oder Finanzanalysen.[Lit16]
- Die letzte Grundkomponent ist der Yet Another Resource Negotiatior, eher bekannt als YARN. Als Ergänzung zum MapReduce-Algorithmus dient YARN zum Ressourcenmanagement und zur Aufgabenverwaltung. Ein globaler Ressource Manager behält den Überblick während pro Anwendung ein Application Master die Aufgaben koordiniert und mit dem Ressource Manager zusammenarbeitet. [Fou18a]

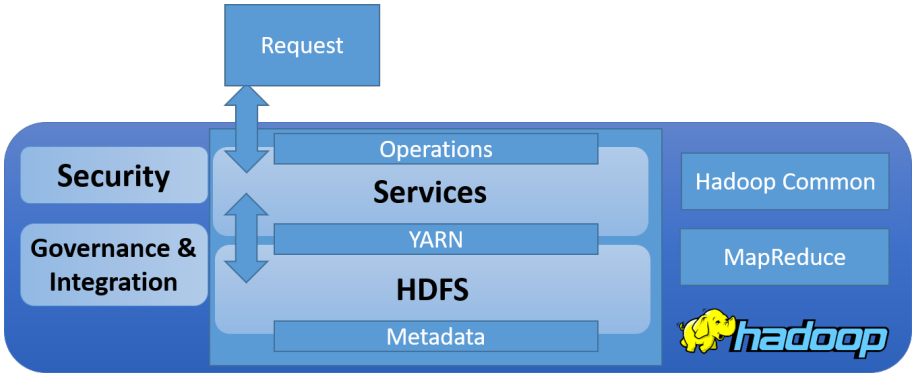


Abbildung 7.1: Beispielarchitektur (Hadoop Logo von [Fou18a])

So besteht Hadoop aus einem verteilten Dateisystem, das durch das Hadoop Common verwaltet, durch den MapReduce für die Verarbeitung verwendet und durch YARN als Resourcen Manager verwaltet wird (siehe Abbildung 7.1). Dabei ist ein Hadoop System in mehrere Cluster aufgeteilt, wobei das Master-Slave-Prinzip zur Anwendung kommt. Ein Cluster nimmt die Rolle des Masters ein und die restlichen Cluster dienen als Sklaven (Slaves). Der Masternode ist für die Verwaltung und das Management der Slaves, sowie für die Verwaltung der Metadaten zuständig. Auf den Slaves werden die Daten repliziert abgelegt, sodass Datenblöcke auf mehreren Slaves zur Verfügung stehen.[Lit16]

Ergänzend zu den Hadoop Basiskomponenten können andere Frameworks, wie zum Beispiel HBase, Hive, Spark innerhalb des Hadoop Sytems genutzt werden. Hadoop ist also eine Basiskomponente die beliebig erweitert werden kann.

Hadoop Distributed File System (HDFS) Ein wichtiger Grundbestandteil von Hadoop ist die Partitionierung von Daten und die parallele Berechnungen über viele Hosts. Dabei nutzt es eine verteiltes Dateisystem, das Hadoop Distributed File System (HDFS). Das HDFS speichert die Daten separat von ihren Metadaten auf sogenannten DataNodes. Die Metadaten liegen auf einem dedizierten Server, dem NameNode. Zur Vermeidung von Datenverlust, werden die Daten auf mehreren DataNodes repliziert. Die Kommunikation zwischen den unterschiedlichen Nodes findet über das TCP-Protokoll statt [SKRC10]. Das HDFS ist plattformunabhängig und kann auf jeder Standardhardware aufgesetzte werden. In der Regel ist die Hardware mit einem Linux Betriebssystem ausgestattet, auf der die Java-Software für die NameNodes und DataNodes problemlos ausgeführt werden kann.

Abbildung 7.2 zeigt die Architektur innerhalb des HDFS. Ein HDFS Cluster besteht aus einem zentralen NameNode und mehreren DataNodes. Der NameNode verwaltet die DataNodes und reguliert die Zugriffe auf Dateien von Clients. Auf den DataNodes liegen die Dateien in Blocks aufgeteilt. Diese Blocks liegen auf mehreren DataNodes. Über den NameNode können Dateizugriffe, Umbenennungen, eine Einteilung in eine Ordnerstruktur erfolgen. Der NameNode behält den Überblick über die Blöcke einer Datei auf den verschiedenen DataNodes. Die DataNodes dahingegen dienen der Speicherung der Dateiblöcke und zur Ausführung der angestoßenen Lese- und Schreibzugriffe. Diese Aktionen koordiniert der NameNode als zentrale Instanz innerhalb des Dateisystems. Dabei durchlaufen keine Nutzerdaten den NameNode. Dies vereinfacht die

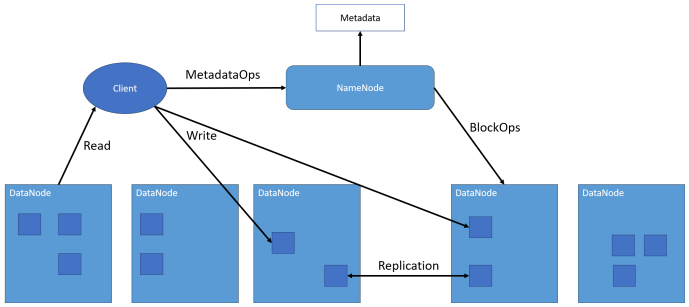


Abbildung 7.2: HDFS Architektur (angelehnt an [Fou18c])

Systemarchitektur erheblich. [Fou18c]
Die Dateiorganisation erfolgt wie bei traditionellen Dateisystemen über hierarchische Ordnerstrukturen. Dabei erstellen Anwendungen Ordner und legen die Dateien darin ab. Innerhalb des HDFS ist die Festlegung der erlaubten Speichernutzung pro Datei oder Ordner per “user quotas“ [Fou18c] möglich. Für Dateizugriffe implementiert das HDFS ein Zugriffsmodel. Dieses ist an das POSIX-Model (siehe Kapitel 4.2.1) angelehnt [Fou18d]. Im folgenden steht das Wort Objekt für Ordner oder Dateien. Jedes Objekt hat drei verschiedene Berechtigungsgruppen. Der Ersteller des Objekt ist der Owner und verfügt über die Administratorrechte. Neben dem Owner gibt es noch die Berechtigungen für Nutzer derselben Gruppe und alle anderen Nutzer. Die Berechtigungen sind mittels r für Lesezugriffe, w für Schreibzugriffe und x für Ausführungszugriffe definiert. Pro Berechtigungsgruppe findet eine Festlegung der Berechtigungen statt. Über Access Control Lists (ACL), siehe 4.2.1, kann eine feinere und komplexere Festlegung der Zugriffsrechte pro Objekt erfolgen [Fou18d].

Komplexer Zugriffskontrollmodelle lassen sich nicht direkt auf dem HDFS umsetzen. Die Implementierung ist innerhalb eines Java-Services möglich, der auf das HDFS aufgesetzt ist. Der Zugriffskontrollservice dient dabei als Schnittstelle zum HDFS und kann mit weiteren Services kombiniert werden. Folgender Abschnitt zeigt die Implementierung des erweiterten Zugriffskontrollmodells als Java-Service, der direkt auf das HDFS zurückgreift.

7.2 Zugriffskontrollservice

Der Zugriffskontrollservice ist als eine Java-Services implementiert, der auf das HDFS in Hadoop zugreift. Abbildung 7.3 zeigt eine abstrakte Übersicht über das Projektes. Der Service setzt eine vorherige Authentifizierung des Subjektes voraus, sodass die Identität, Subjektattribute und Subjektrolle bekannt sind. Der Input setzt sich aus dem angefragten Objekt, der Subjektrolle und aus dem angefragten Zugriffsrechtes zusammen. Für alle schreibenden Zugriffe können des weiteren der zu schreibende Inhalt als Eingabeparameter hinzugefügt werden. Für einige Regeln ist werden spezielle Subjektattribute benötigt. Diese gehören der Anfrage an. Die Anfrage gelangt zuerst in den Requestchecker. Dieser überprüft ob alle Eingabeparameter valide sind. Danach beginnt der Zugriffskontrollmechanismus mit den Überprüfungen der Richtlinien. Zur Durchführung der einzelnen Überprüfungen greift der Mechanismus auf die Sicherheitsartefakte zu. Außerdem baut der HDFSConnector eine Verbindung zum HDFS auf, damit die Objektattribute ausgelesen und bei der

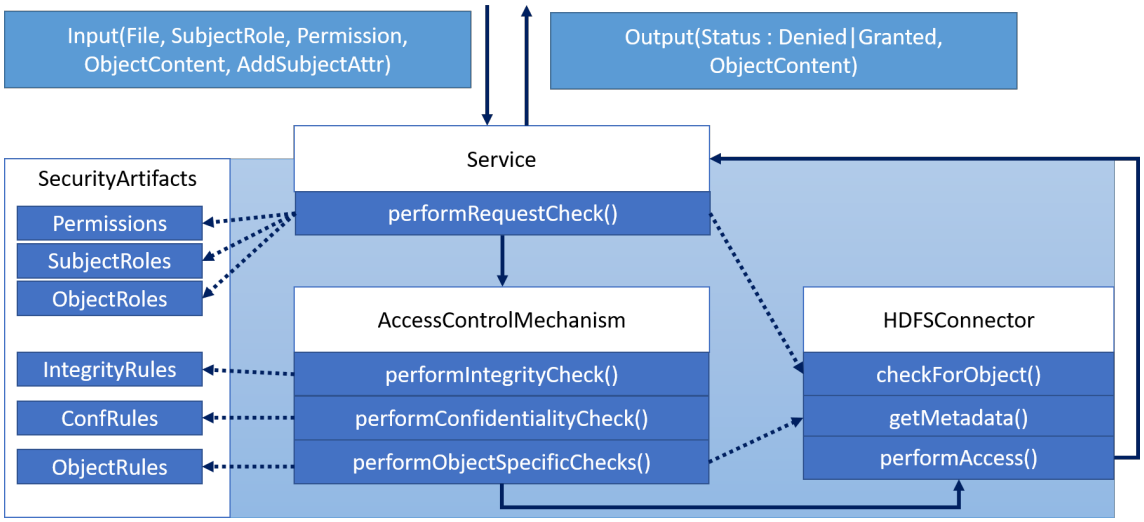


Abbildung 7.3: Projektübersicht

Listing 7.1 HDFSConnector: Liste aller Dateien des HDFS

```
1  Configuration conf = new Configuration();
2  FileSystem fs = FileSystem.get(conf);
3  FileStatus[] fileStatus = fs.listStatus(new Path(HDFS_SUBDIRECTORY));
4
5  for (FileStatus status : fileStatus) {
6     System.out.println(status.getPath().toString());
7  }
```

Anwendung der objektspezifischen Regeln genutzt werden können. Nachdem alle Überprüfungen nacheinander erfolgreich durchlaufen sind, wird der HDFSConnector zur Durchführung des Zugriffsrechtes aufgerufen. Dieser liefert das Ergebnis an den Service zurück. Der Service liefert den Output über den Status der Anfrage und bei erlaubten Zugriff die gewünschten Inhalte.

HDFSConnector Über den HDFSConnector verbindet sich der Zugriffskontrollservice mit dem HDFS. Dies ist für den Erhalt der Objektattribute und die Durchführung des Zugriffsrechtes nötig. Listing 7.1 zeigt den Code zur Auflistung aller Dateien in einem Unterordner des HDFS. Dazu navigiert das FileSystems zum Unterordner und liest den FileStatus aus. Über den HDFSConnector können ebenfalls Dateien ausgelesen oder geschrieben werden. Listing 7.2 zeigt das Auslesen einer Datei im HDFS. Mithilfe eines InputStreams und eines JAXB-Converters zeigt das Codebeispiel das Lesen einer Datei, die die Lebensumstände beinhaltet. Da es sich um eine xml-Datei mit bekannten Schema handelt, erstellt der Unmarshaller ein den Inhalten der xml-Datei entsprechendes Java-Objekt, das im Zugriffskontrollservice weiter verwendet werden. Äquivalent zum Lesen einer Datei, erfolgt das Schreiben in eine existierende Datei über ein OutputStream und einen Writer. Dazu muss die XML-Datei zuvor aus den Java-Objekt über den Marshaller von JAXB erstellt werden.

Listing 7.2 HDFSConnector: Auslesen einer Datei

```
Path inFile = new Path(requestedFileURL);
if (!fs.exists(inFile)) {
   System.out.println("Input file not found");
   return;
}

inputStream = fs.open(inFile);
JAXBContext context = JAXBContext.newInstance(LivingCondition.class);
Unmarshaller un = context.createUnmarshaller();
LivingCondition testConditions = (LivingCondition) un.unmarshal(inp);
IOUtils.closeStream(inputStream);
```

Listing 7.3 Definition der Integritätsklassen

```
        package SecurityArtifacts;

        public enum IntegrityClass {
        CRUCIAL(2), ESSENTIAL(1), UNKNOWN(0);

        private int hierarchyLevel;

        public int getHierarchyLevel(){
        return this.hierarchyLevel;
        }
        }
```

Zugriffsrichtlinien Alle Sicherheitsartefakte sind in einem Separaten Package untergebracht. Neben den einzelnen Richtlinien beinhaltet das Package die Definition der Subjektrollen, Objektklassen, Zugriffsrechte, Integritäts- und Sicherheitsklassen. Listing 7.3 zeigt die Definition der Integritätsklassen als eine Enum mit zugeordneten Werten. Die numerischen Werte bilden die Hierarchie der Integritätsklassen ab. Innerhalb der Definition der Subjektrollen und Objektklasse findet die Zuordnung zu den Integritäts- und Sicherheitsklassen statt. Der Service initialisiert alle Sicherheitsartefakte. Die einzelnen Richtlinien sind über statische Methoden implementiert. Listing 7.4 zeigt die Implementation der Integritätsregeln. Der Vergleich der Integritätsklassen findet über das Hierarchielevel der einzelnen Klassen statt. Ebenso erfolgt die Definition der Objektregeln. Listing 7.5 zeigt die Implementierung der Objektregeln R2.1 und R2.2.

Zugriffskontrollmechanismus Ausgehend vom Zugriffskontrollmechanismus werden die Regeln der Richtlinien durchgesetzt. Der Mechanismus enthählt weniger die Logik der einzelnen Überprüfung sondern Koordiniert den Überprüfungsablauf. Er ist die Schnittstelle zwischen dem Service und den Richtlinien und liefert dem Service das endgültige Ergebnis der Überprüfungen zurück. Dabei bricht der Zugriffskontrollmechanismus beim ersten Fehlschlagen einer Überprüfung direkt ab.

Der Aufbau des Projektes orientiert sich an der logischen Architektur des Zugriffskontrollmodells. Ein zentraler Service dient als Schnittstelle zwischen dem Anfragesteller und dem Zugriffskontrollmechanismus. Basierend auf den Sicherheitsartefakten werden die einzelnen Überprüfungen durchgeführt. Nach dem Integritäts- und Sicherheits-Check, die für alle Anfragen durchlaufen

Listing 7.4 Regeln der Integritätsrichtlinie

```
1          private static boolean applySimpleIntegrityRule(IntegrityClass oi , IntegrityClass si){
2             return si.getHierarchyLevel() >= oi.getHierarchyLevel();
3          }
4          private static boolean applyIntegrityConfinementRule(IntegrityClass oi , IntegrityClass si){
5             return si.getHierarchyLevel() <= oi.getHierarchyLevel();
6          }
7          private static boolean applyInvocationProperty(IntegrityClass i1 , IntegrityClass i2){
8             return i2.getHierarchyLevel() >= i1.getHierarchyLevel();
9          }
10         private static boolean applyIntegriyCreatorRule(IntegrityClass oi , IntegrityClass si){
11            return applySimpleIntegrityRule(oi, si);
12         }
```

Listing 7.5 Regeln der Objektrichtlinie

```
1          private static boolean applyAppendCareRecordRule(HDFSObject o, Subject s, PermissionSet p){
2            return (o.getObjectClass() == ObjectSecurityClass.CASERECORD
3                     && s.getSubjectRole ==
                              (SubjectSecurityRoles.DOCTOR||SubjectSecurityRoles.CONSULTANT)
4                     && p == PersmissionSet.APPEND);
5          }
6          private static boolean applyWriteCareRecordRule(HDFSObject o, Subject s, PermissionSet p){
7             return ( applyAppendCareRecordRule(o,s,p)
8             && o.getOwnerId() == s.getId()
9             && p == PermissionSet.WRITE );
10         }
```

werden, erfolgt die Überprüfung der Objektregeln. Diese sind bei den Sicherheitsartefakten abgelegt. Manche Regeln benötigen Objektattribute zur Überprüfung. Dazu greift der HDFSConnector auf das HDFS-FileSystem zu und liest das angefragte Objekt aus. Im Falle einer bekannten Objektstruktur kann das Objekt zu einem Java-Objekt umgewandelt werden. Im Anwendungsfall handelt es sich ausschließlich um XML-Dateien, die mithilfe von JAXB verarbeitet werden können. Sobald das Zugriffsrecht vom AccessControlMechanism gewährt wurde, kann über den HDFSConnector das Zugriffsrecht ausgeführt werden. Der HDFSConnector verfügt über uneingeschränkte Rechte auf das HDFS. Eine Zugriffsbeschränkung und Zugriffskontrolle muss durch den Zugriffskontrollmechanismus erfolgen, sodass keine unberechtigten Zugriffe direkt über den HDFSConnector auf das HDFS laufen.

7.2.1 Evaluation

Der Service implementiert das für den Anwendungsfall aufgestellte Zugriffskontrollmodell. Alle benötigten Komponenten sind als Java-Objekte definiert. Im Falle der Richtlinien erfolgt die Implementierung der Regeln jeweils als eigene statische Methoden. Eine Lösung der Umsetzung der Richtlinien und Regeln wäre die Verwendung von Richtliniensprachen. Für die Implementierung wurde die Richtliniensprache XACML in Betracht gezogen. Es liegen einige Bibliotheken zur Implementierung von Richtlinien in Java mithilfe von XACML vor. Die Einbindung von XACML in das Projekt konnte im Rahmen der Implementierung nicht realisiert werden. Die Verwendung

von Richtliniensprachen für einen Zugriffskontrollservice basierend auf Java ist ein offener Punkt, der in weiteren Arbeiten aufgegriffen werden kann.

Bei der Implementierung des Zugriffskontrollmodells lag der Fokus vor allem auf der exemplarischen Umsetzung des Zugriffskontrollmechanismus und der damit verbundenen Sicherheitsartefakte. Zur Verbesserung der Zugriffe auf das HDFS gibt es unterschiedliche Hadoop-Erweiterungen, wie zum Beispiel YARN. Über YARN können Anfragen an das HDFS gestellt werden. Die Einbettung des Zugriffskontrollmechanismus in ein komplettes System zur Untersuchung des Zusammenspiels des Zugriffskontrollservices mit anderen Komponenten ist ebenfalls ein interessanter Aspekt für weiterführende Arbeiten.

8 Zusammenfassung und Ausblick

Das Data Lake Konzept ermöglicht eine Datenspeicherung im Rohformat. Im Gegensatz zu dem Data Warehouse Konzept setzt ein Data Lake eine Vorverarbeitung der Daten über den ETL-Prozess nicht voraus. Die Daten durchlaufen lediglich einen Überprüfungs- und Anreicherungsprozess, den Data Wrangling Prozess, bevor sie in den Data Lake gelangen. Während des Prozesses erfolgt eine Anreicherung der Daten mit Metadaten. Des Weiteren bezieht der Prozess die zur Definition der Zugriffskontrollregeln herangezogenen Nutzungsrichtlinien zu den einzelnen Datenobjekten. Ein Zugriffskontrollmechanismus für Data Lakes stützt sich auf den im Data Wrangling Prozess angereicherten Daten und deren zugehörigen Sicherheitsartefakte. Die einzelnen Schritte des Data Wrangling Prozesses spiegeln sich der logischen Architektur eines Data Lakes wieder. Ein Data Lake besteht aus drei Verarbeitungsstufen und drei darunter liegenden Funktionsschichten. Während die Verarbeitungsstufen spiegeln den Datenbeschaffungsprozess, den Datenspeicherungsprozess und den Datenzugriffsprozess wieder. Innerhalb des Datenzugriffsprozesses ist eine Zugriffskontrolle zum Schutz des Data Lakes vor unberechtigten Zugriffen nötig. Das Zugriffskontrollmodul greift auf das in der Sicherheitsschicht vorliegende Sicherheitsmodul und auf die vorliegenden Richtlinien zurück.

In der Literatur liegen viele Arbeiten zu Big Data und Data Warehouse Zugriffskontrollmodellen vor. Aus den vorgestellten verwandten Arbeiten zeigt sich, dass Lösungen eines Zugriffskontrollmodells aus der Kombination zweier existierender Zugriffskontrollmodelle unterschiedlicher Sicherheitsstrategien bestehen. Zur Erklärung der bekanntestens Zugriffskontrollmodelle der einzelnen Sicherheitsstrategien erfolgte eine Definition der benötigten Grundbegriffe. Ein Subjekt erfragt ein Zugriffsrecht auf ein Objekt. Sowohl Subjekte als auch Objekte können mithilfe von Rollen und Objektklassen gruppiert werden. Mithilfe von Kommandos kann der Objektstatus und somit Objektattribute manipuliert werden, sodass eine Änderung der Zugriffsrechte möglich ist. Zugriffskontrollmodelle verwenden in der Regel fünf Zugriffsrechte. Neben einem einfachen Lese- und Schreibzugriff gibt es Rechte zur Erstellung, Ausführung oder Lösung eines Objektes. Zugriffskontrollmodelle können darüber hinaus weitere Zugriffsrechte definieren und betreffende Regelungen definieren. Das Zugriffsmatrix-Modell der Sicherheitsstrategie DAC bildet die Zugriffsrechte von Subjekten auf Objekten in einer Matrix ab. Modelle der system-bestimmten Zugriffskontrolle verwenden das Zugriffsmatrix-Modell und erweitern es durch eigene Regeln für die einzelnen Zugriffsrechte. Das Bell-LaPadula definiert Regeln zu Einhaltung der Vertraulichkeit von Informationen. Das Biba Modell fokussiert sich dahingegen auf Regeln zur Einhaltung der Objektintegrität. Sowohl in Bell-LaPadula als auch bei Biba erfolgt die Definition der Regeln basierend auf einer Klassifizierung von Objekt und Subjekt. Dies nutzen ebenfalls rollen-basierte Zugriffskontrollmodelle. Dabei bündeln Rollen die Zugriffsrechte, die ein Subjekt zur Aufgabenerfüllung benötigt. Zur Umsetzung komplexer und dynamischer Zugriffsbeschränkungen wird das attribut-basierte Zugriffskontrollmodell verwendet. Dieses Modell erlaubt die Formulierung von Bedingungen gebunden an Attribute zur Erteilung von Zugriffsrechten. Die verschiedenen Modelle verfügen über Vor- und Nachteile.

Zur Evaluation der benötigten Zugriffskontrollmodelle für einen hybriden Ansatz zugeschnitten auf das Data Lake Konzept wurden Anforderungen un Kriterien herausgearbeitet. Dies Anforderungen beziehen sich aus der Überlegung, welche Eigenschaften aus den vorgestellten Zugriffskontrollmodellen für ein Data Lake sinnvoll sind. Neben einer notwendigen Dynamik, steht der Schutz des gesamten Data Lakes im Vordergrund. Durch die fehlende Vorverarbeitung der Daten kann eine Manipulation nicht ausgeschlossen werden. Aus diesem Grund benötigt es Integritätsprüfungen um Daten anhand ihres Datenursprungs einzuordnen. Darüber hinaus dienen Geheimhaltungsstufen zum Schutz der Dateninhalte vor unberechtigten Zugriffen. Bei der Evaluation zeigte sich, dass das attribut-basierte Zugriffskontrollmodell die meisten Kriterien erfüllt. In Kombination mit dem Bell-LaPadula, dem Biba und dem rollen-basierten Zugriffskontrolle können alle Kriterien erfüllt werden. Das erweiterte Zugriffskontrollmodell für Data Lakes verwendet aus diesem Grund die logische Architektur des attribut-basierten Zugriffskontrollsystem und erweitert dies mit Elementen aus den anderen Zugriffskontrollmodellen. Die Entscheidungsfindung über eine Zugriffsrechtevergabe erfolgt über vier Überprüfungsschritte, die Sicherheitsartefakte und Attribute verwenden. Nach der Überprüfung ob die Anfrage der benötigten Form entspricht erfolgt ein Integritäts- und ein Sicherheits-Check. Danach entscheiden objekt-spezifische Regeln über die endgültige Zugriffsrechtevergabe. Dieser Zugriffskontrollmechanismus wurde im Rahmen eines Anwendungsfalls modelliert und implementiert. Die Implementierung zeigte vor allem auf, dass benötigte Technologien für die Realisierung der Zugriffskontrolle in unterschiedlichen Data Lake Lösungen herangezogen werden müssen.

Ausblick Das aufgestellte Zugriffskontrollmodell bietet ein logische Grundgerüst zur Umsetzung in einem Data Lake. Es ermöglicht die Modellierung grundlegender Sicherheitsmechanismen bei der Zugriffskontrolle innerhalb eines Data Lakes. Es handelt sich bei dem ausgearbeiteten Modell um einen ersten Lösungsansatz. Denkbar wäre eine Erweiterung des Zugriffskontrollmodells durch weitere Elemente aus anderen Zugriffskontrollmodellen. Vor allem das Chinese-Wall Modell ist hierfür interessant. Es ermöglicht die Verhinderung von Interessenkonflikten und könnte in einem Data Lake zur Abgrenzung verschiedener Datenbestände dienen.

Eine Weiterentwicklung des erweiterten Zugriffskontrollmodells ist möglich. Beispielsweise kann der Umgang mit Richtlinien innerhalb des Modells weiter spezifiziert werden. Neben der Formulierung der Richtlinien mithilfe von Richtliniensprachen ist auch das dynamische Aktualisieren der Richtlinien ein interessanter Punkt. Des Weiteren können Metadaten einen größeren Einfluss bei der Zugriffsrechtevergabe zugesprochen werden. Die Definition und Verwendung der benötigten Metadaten kann für das Modell noch genauer spezifiziert werden. Durch die Aufstellung eines sehr abstrakten Modells bleiben auch Aspekte zur Umsetzung in den verschiedenen Data Lake Lösungen offen.

Literaturverzeichnis

[AA10] S. Ahmad, R. Ahmad. „An improved security framework for data warehouse: A hybrid approach“. In: *Information Technology (ITSim), 2010 International Symposium in*. Bd. 3. IEEE. 2010, S. 1586–1590 (zitiert auf S. 16).

[AAA+18] H. F. Atlam, M. O. Alassafi, A. Alenezi, R. J. Walters, G. B. Wills. „XACML for Building Access Control Policies in Internet of Things“. In: *Proceedings of the 3rd International Conference on Internet of Things, Big Data and Security (IoTBDS 2018), Setúbal, Portugal*. 2018, S. 19–21 (zitiert auf S. 50, 51).

[Afy06] H. A. Afyouni. *Database Security and Auditing: protecting Data Integrity and Accessibility*. Hochschulbibliothek Reutlingen. Boston, Mass.: Thomson, 2006, XVIII, 429 Seiten. ISBN: 978-0-619-21559-0 (zitiert auf S. 38, 40).

[Alb02] H. Albrecht. *Lipobay-Skandal -Desaster ohne Nebenwirkungen*. Hrsg. von Z. O. GmbH. 1. Aug. 2002. URL: https://www.zeit.de/2002/32/200232_kasten_lipobay_xml (zitiert auf S. 68).

[Aut10] V. .-.-. V. der Automobilindustrie e.V. *Band 4: Sicherung der Qualität in der Prozesslandschaft*. Bd. 4. 2010 (zitiert auf S. 66–68).

[BK15] S. K. Bansal, S. Kagemann. „Integrating big data: a semantic extract-transform-load framework“. In: *Computer* 48.3 (2015), S. 42–50 (zitiert auf S. 19, 20).

[BZ12] A. Basta, M. Zgola. *Database security*. Englisch. Preparing tomorrow's information security professionals. Hochschulbibliothek Reutlingen. Boston, MA: Course Technology Cengage Learning, 2012, XVII, 301 Seiten. ISBN: 978-1-435-45390-6 (zitiert auf S. 34, 46).

[CF15] P. Colombo, E. Ferrari. „Privacy aware access control for Big Data: a research roadmap“. In: *Big Data Research* 2.4 (2015), S. 145–154 (zitiert auf S. 15).

[Chi12] T. Chia. *Confidentiality, Integrity, Availability: The three components of the CIA Triad*. 20. Aug. 2012. URL: https://security.blogoverflow.com/2012/08/confidentiality-integrity-availability-the-three-components-of-the-cia-triad/ (zitiert auf S. 13).

[Dev14] B. Devlin. *Data lake muddies the waters on big data management*. Hrsg. von TechTarget. 2014. URL: https://searchbusinessanalytics.techtarget.com/feature/Data-lake-muddies-the-waters-on-big-data-management (zitiert auf S. 21).

[Dix10] J. Dixon. „Pentaho, Hadoop, and data lakes“. In: *James Dixon's Blog* (Okt. 2010). URL: https://jamesdixon.wordpress.com/2010/10/14/pentaho-hadoop-and-data-lakes/ (zitiert auf S. 20).

[Dix14] J. Dixon. *Data Lakes Revisited*. 25. Sep. 2014. URL: https://jamesdixon.wordpress.com/2014/09/25/data-lakes-revisited/ (zitiert auf S. 21).

[Eck08] C. Eckert. *IT-Sicherheit: Konzepte - Verfahren - Protokolle*. Deutsch. 5., überarb. Aufl. Hochschulbibliothek Reutlingen. München ; Wien: Oldenbourg, 2008, XIV, 925 Seiten. ISBN: 978-3-486-58270-3. URL: http://bvbr.bib-bvb.de:8991/F?func=service&doc_library=BVB01&doc_number=015719788&line_number=0001&func_code=DB_RECORDS&service_type=MEDIA (zitiert auf S. 33–35, 37, 38, 41, 43, 44, 47, 48).

[Fan15] H. Fang. „Managing data lakes in big data era: What's a data lake and why has it became popular in data management ecosystem“. In: *Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), 2015 IEEE International Conference on*. IEEE. 2015, S. 820–824 (zitiert auf S. 22).

[FGL+16] T. Furche, G. Gottlob, L. Libkin, G. Orsi, N. W. Paton. „Data Wrangling for Big Data: Challenges and Opportunities.“ In: *EDBT*. 2016, S. 473–478 (zitiert auf S. 24).

[FM16] D. Fasel, A. Meier. *Big Data: Grundlagen, Systeme und Nutzungspotenziale*. Springer-Verlag, 2016 (zitiert auf S. 19).

[Fou18a] A. S. Foundation. *Apache Hadoop*. 2018. URL: http://hadoop.apache.org/ (zitiert auf S. 75, 76).

[Fou18b] A. S. Foundation. *Apache Hadoop 2.9.1*. 2018. URL: http://hadoop.apache.org/docs/stable/ (zitiert auf S. 75).

[Fou18c] A. S. Foundation. *HDFS Architecture*. 2018. URL: http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html (zitiert auf S. 77).

[Fou18d] A. S. Foundation. *HDFS Persmissions Guide*. 2018. URL: http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsPermissionsGuide.html (zitiert auf S. 77).

[FTVP06] E. Fernández-Medina, J. Trujillo, R. Villarroel, M. Piattini. „Access control and audit model for the multidimensional modeling of data warehouses“. In: *Decision Support Systems* 42.3 (2006), S. 1270–1289 (zitiert auf S. 15).

[Gem18] Gematik.de, Hrsg. *Telematikinfrastruktur – das sichere Netz für alle*. 17. Juli 2018. URL: https://www.gematik.de/telematikinfrastruktur/ (zitiert auf S. 13, 65).

[HGQ16] R. Hai, S. Geisler, C. Quix. „Constance: An intelligent data lake system“. In: *Proceedings of the 2016 International Conference on Management of Data*. ACM. 2016, S. 2097–2100 (zitiert auf S. 30).

[HM11] T. R. Hummel, C. Malorny. *Total Quality Management: Tipps für die Einführung*. Deutsch. 4. Aufl. Pocket-Power ; 1. UB Stadtmitte. München: Hanser, 2011, 123 Seiten. ISBN: 978-3-446-41609-3 (zitiert auf S. 65).

[Hu14] C. T. Hu. *Attribute Based Access Control (ABAC) Definition and Considerations*. Techn. Ber. 2014 (zitiert auf S. 49).

[IEE17] IEEE. „IEEE Approved Draft Standard for Information Technology - Portable Operating System Interface (POSIX(R))“. In: *IEEE P1003.1/D1, Jul 2017* (Jan. 2017). pp.108, S. 1–3946 (zitiert auf S. 40).

[Inc14] G. Inc. *Gartner Says Beware of the Data Lake Fallacy*. Hrsg. von N. H. Andrew White. 28. Juli 2014. URL: https://www.gartner.com/newsroom/id/2809117 (zitiert auf S. 21).

[Inm14] W. Inmon. *Textual Disambiguation. Transform text to its native format to help simplify its analysis for making decisions*. Hrsg. von I. B. D. Ã. Hub. 4. Juli 2014. URL: https://www.ibmbigdatahub.com/blog/textual-disambiguation (zitiert auf S. 24).

[Inm16] B. Inmon. *Data Lake Architecture: Designing the Data Lake and Avoiding the Garbage Dump*. Technics Publications, 2016 (zitiert auf S. 23, 24).

[Jon13] S. Jones. „Why Business needs a Lake for Data not a Warehouse“. In: (5. Dez. 2013). URL: https://www.capgemini.com/2013/12/why-business-needs-a-lake-for-data-not-a-warehouse/# (zitiert auf S. 21).

[JSW17] B. Jung, S. Schweißer, J. Wappis. *8D - Systematisch Probleme lösen*. Deutsch. 3. Auflage. Pocket Power ; 062. München: Hanser, 2017. ISBN: 978-3-446-45240-4. URL: http://nbn-resolving.de/urn/resolver.pl?urn=10.3139/9783446452404 (zitiert auf S. 65–68).

[Kep13] V. Keppel. „Klassifikation und Analyse von IT-Sicherheitsmodellen“. In: (2013) (zitiert auf S. 34).

[KQS+98] N. Katic, G. Quirchmay, J. Schiefer, M. Stolba, A. M. Tjoa. „A prototype model for data warehouse security based on metadata“. In: *Database and Expert Systems Applications, 1998. Proceedings. Ninth International Workshop on*. IEEE. 1998, S. 300–308 (zitiert auf S. 15, 16).

[Lit16] N. Litzel. *Was ist Hadoop?* 2016. URL: https://www.bigdata-insider.de/was-ist-hadoop-a-587448/ (zitiert auf S. 75, 76).

[May18] May. *Spahn will Digitalisierung zu einem Schwerpunkt machen*. Hrsg. von aerzteblatt.de. 15. März 2018. URL: https://www.aerzteblatt.de/nachrichten/91835/Spahn-will-Digitalisierung-zu-einem-Schwerpunkt-machen (zitiert auf S. 13, 65).

[NBL+10] Q. Ni, E. Bertino, J. Lobo, C. Brodie, C.-M. Karat, J. Karat, A. Trombeta. „Privacy-aware Role-based Access Control“. In: *ACM Trans. Inf. Syst. Secur.* 13.3 (Juli 2010), 24:1–24:31. ISSN: 1094-9224. DOI: 10.1145/1805974.1805980. URL: http://doi.acm.org/10.1145/1805974.1805980 (zitiert auf S. 15).

[NRD18] I. D. Nogueira, M. Romdhane, J. Darmont. „Modeling Data Lake Metadata with a Data Vault“. In: *Proceedings of the 22nd International Database Engineering & Applications Symposium*. ACM. 2018, S. 253–261 (zitiert auf S. 30).

[PP00] T. Priebe, G. Pernul. „Towards OLAP security design—survey and research issues“. In: *Proceedings of the 3rd ACM international workshop on Data warehousing and OLAP*. ACM. 2000, S. 33–40 (zitiert auf S. 16).

[PP15] P. Pasupuleti, B. S. Purra. *Data Lake Development with Big Data*. Packt Publishing Ltd, 2015 (zitiert auf S. 13).

[QHV16] C. Quix, R. Hai, I. Vatov. „GEMMS: A Generic and Extensible Metadata Management System for Data Lakes.“ In: *CAiSE Forum*. 2016, S. 129–136 (zitiert auf S. 29).

[RS00] A. Rosenthal, E. Sciore. „View security as the basis for data warehouse security.“ In: *DMDW*. 2000, S. 8 (zitiert auf S. 16).

[SB15] W. Stallings, L. Brown. *Computer security: principles and practice*. Englisch. Third ed., global ed. Hochschulbibliothek Reutlingen. Boston, Mass. [u.a.]: Pearson, 2015, 840 Seiten. ISBN: 978-1-292-06617-2 (zitiert auf S. 34, 36, 37, 39–41, 43–50).

[SKRC10] K. Shvachko, H. Kuang, S. Radia, R. Chansler. „The Hadoop Distributed File System“. In: *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*. 2010, S. 1–10. DOI: 10.1109/MSST.2010.5496972 (zitiert auf S. 76).

[SS94] R. S. Sandhu, P. Samarati. „Access control: principle and practice“. In: *IEEE communications magazine* 32.9 (1994), S. 40–48 (zitiert auf S. 42, 44).

[SV00] P. Samarati, S. C. de Vimercati. „Access control: Policies, models, and mechanisms“. In: *International School on Foundations of Security Analysis and Design*. Springer. 2000, S. 137–196 (zitiert auf S. 35–39, 41, 42, 47).

[TSRC15] I. G. Terrizzano, P. M. Schwarz, M. Roth, J. E. Colino. „Data Wrangling: The Challenging Yourney from the Wild to the Lake.“ In: *CIDR*. 2015 (zitiert auf S. 24–26).

[Ver18] A. F. Vermeulen. „Data Science Technology Stack“. In: *Practical Data Science*. Springer, 2018, S. 1–13 (zitiert auf S. 19, 20).

[Wil18] A. Wilkens. *Gesundheitsminister will Patientenakte auf Handys zugänglich machen*. Hrsg. von heise online. 16. Juli 2018. URL: https://www.heise.de/newsticker/meldung/Gesundheitsminister-will-Patientenakte-auf-Handys-zugaenglich-machen-4110662.html (zitiert auf S. 13, 65).

[Woo11] D. Woods. „Big Data Requires a Big, New Architecture“. In: *Forbes* (2011). URL: https://www.forbes.com/sites/ciocentral/2011/07/21/big-data-requires-a-big-new-architecture (zitiert auf S. 21).

[YJR+14] K. Yang, X. Jia, K. Ren, R. Xie, L. Huang. „Enabling efficient access control with dynamic policy updating for big data in the cloud“. In: *INFOCOM, 2014 Proceedings IEEE*. IEEE. 2014, S. 2013–2021 (zitiert auf S. 15).

Alle URLs wurden zuletzt am 07. 11. 2018 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift