

Masterarbeit zum Thema:

Entwicklung einer hochverfügbaren und skalierbaren Open-Source IDE in der Cloud

Georg Schulz
Matrikelnr.: 88737
E-mail: schulzge@hs-albsig.de

Dozent: Prof. Dr. Thomas Eppler

12. November 2022

Technische Informatik
Fakultät Informatik
Hochschule Albstadt-Sigmaringen (HSAS)

Danksagung

An dieser Stelle möchte ich mich für die Unterstützung während der Anfertigung dieser Masterarbeit bedanken.

Ein besonderer Dank gilt meinem ehemaligen Arbeitskollegen und Freund Christian Pfarr, der meine Masterarbeit betreut und begutachtet hat. Für die hilfreichen Anregungen, die konstruktive Kritik und die Informationsbereitschaft bei der Erstellung möchte ich mich herzlich bedanken.

Ebenfalls bedanken möchte ich mich bei meiner Schwester Elisabeth Schulz und Freund Sebastian Weber für die moralische Unterstützung und das Korrekturlesen.

Georg Schulz

Frankfurt am Main, den 12.11.2022

Abstract

Mit Elastic MapReduce (EMR) von Amazon Web Services (AWS) und Azure Hadoop Insight (HDInsight) von Microsoft Azure existieren Enterprise-Produkte, die eine cloudbasierte Entwicklungsumgebung über Jupyter-Notebooks für Data Scientisten bereitstellen. Bei der vergleichenden Analyse der angebotenen Produkte mit dem Konzept Jupyter-Hub on Hadoop von Jim Crist lässt sich feststellen, dass die gewählte IT-Architektur der Cloud-Dienste insgesamt keine *vollständige* Integration der IDE Jupyter in ein zugrundeliegendes Hadoop-Ökosystem darstellt. Ursache dafür ist unter anderem eine zusätzliche Authentifizierungsschicht über den REST-Server Apache Livy. Dabei setzt AWS auf eine Container-Lösung als Frontend zu einem davon architektonisch isolierten Hadoop-Ökosystem. Microsoft Azure wiederum schlägt die Verbindung mehrere Nutzer zu einem HDInsight-Cluster von einer lokalen Jupyter-Instanz vor und lagert damit die Integration von JupyterHub an den Endnutzer aus. Ziel dieser Masterarbeit ist deswegen die Entwicklung einer eigens entwickelten Open-Source IDE innerhalb einer cloudbasierten, hochverfüglichen und skalierbaren Umgebung. Dabei ist als Ergebnis festzuhalten, dass diese mehr Komfort in Bezug auf die Zusammenarbeit an einem Big Data-Projekt bietet. Data Scientisten können große Dateien über JupyterLab direkt in das HDFS laden und analysieren sowie sich über eine Berechtigungsstruktur innerhalb der IDE austauschen. Weitere Vorteile sind die Hochverfügbarkeit aller Komponenten der Entwicklungsumgebung, eine vollwertige Nutzung von Jupyter durch zahlreiche weitere Features sowie die vollständige Integration der Endnutzer in das Sicherheitssystem von Apache Hadoop. Dem gegenüber stehen ein vergleichsweiser erhöhter Aufwand in der Konfiguration und Wartung der IT-Infrastruktur. Eine Entscheidung für oder gegen populäre Enterprise-Produkte ist damit je nach Situation und Anforderung des Auftraggebers unterschiedlich zu bewerten. Da die entwickelte IDE in einer Infrastructure as Code (IAC) in einem öffentlichen Git-Repository vorliegt, kann sie jederzeit von Data Scientisten kostenlos genutzt und auf individuelle Vorlieben zugeschnitten werden.

Inhaltsverzeichnis

Inhaltsverzeichnis	I
Abbildungsverzeichnis	IV
Tabellenverzeichnis	V
Auflistungsverzeichnis	VI
Abkürzungsverzeichnis	VII
1. Einleitung	1
2. Theoretischer Hintergrund	3
2.1. Integrated Development Environment	3
2.1.1. Offline vs. Online	3
2.1.2. Das Jupyter-Projekt	4
2.2. Big Data-Systeme	6
2.2.1. Hadoop-Ökosystem	6
2.2.2. Apache Spark	9
2.2.3. Apache Hadoop MapReduce vs. Apache Spark	10
2.3. Ansible	12
2.3.1. Inventory	13
2.3.2. Inventory-Variablen	14
2.3.3. Ansible-Rolle	15
2.3.4. Ansible-Playbook	16
2.3.5. Ansible-Collection	17
3. Big Data in der Cloud - Vorstellung von Analyseplattformen	19
3.1. Amazon Elastic MapReduce	20
3.2. Azure Hadoop Insight	22
4. Planung der zugrundeliegenden IT-Infrastruktur	24
4.1. JupyterHub on Hadoop	24
4.2. Architekturüberblick	25
4.3. Hetzner Cloud	28
5. Technische Umsetzung der IDE	30
5.1. Automatisiertes Verfahren - Installation und Konfiguration	30

5.2. Vorbereitungen in der Hetzner Cloud	34
5.3. Überprüfung der einzelnen Komponenten	36
5.4. Test der Ausfallsicherheit	39
5.5. Berechtigungsstruktur der IDE	40
6. Vergleich mit Enterprise-Produkten	42
6.1. Amazon EMR	42
6.2. Azure HDInsight	45
6.3. Custom IDE	49
6.4. Gegenüberstellung der jeweiligen Funktionalitäten	52
7. Zusammenfassung und Diskussion	54
8. Ausblick	56
Glossar	57
Literatur	60
Anhangsverzeichnis	67
A. Ansible	67
A.1. ansible.cfg	67
A.2. setup.yml	68
B. Custom IDE	69
B.1. Preflight Check vor IDE-Installation	69
B.2. LDAP-Konfiguration des Hadoop-Ökosystems	71
B.3. Ansible Play Recap nach Installation über setup.yml	72
C. Amazon EMR	73
C.1. Anmeldung AWS	73
C.2. Startseite AWS-Konsole	74
C.3. EMR - AWS-Konsole	75
C.4. EMR Cluster - Hauptkonfiguration	76
C.5. EMR Cluster - erweiterte Konfiguration	77
C.6. EMR Cluster - Maschinenanzahl	77
C.7. EMR Cluster - Namen vergeben	78
C.8. EMR Cluster - EC2-Schlüsselpaar	78
C.9. EMR Cluster - Anwendungsverlauf	79
C.10.EMR Cluster - Security Groups	79
C.11.EMR Cluster - Regeln für Datenverkehr bearbeiten	80
C.12.EMR Cluster - JupyterHub-Port hinzufügen	80
C.13.AWS-Konsole - S3	81
C.14.Amazon S3 - Bucket erstellen	81

C.15.Amazon S3 - Blockade öffentlicher Zugriffe deaktivieren	82
D. Azure HDInsight	83
D.1. Azure-Portal - nach HDInsight suchen und erstellen	83
D.2. HDInsight - Cluster konfigurieren	84
D.3. HDInsight - Clustererstellung nicht möglich ohne Kontingenterhöhung . .	85
D.4. Microsoft Azure - Kontingent für HDInsight	86
D.5. HDInsight - Erfolgreiche Clusterbereitstellung	86
D.6. HDInsight Cluster - Speicherkonto	87
D.7. HDInsight Speicherkonto - Datei hochladen	87
D.8. HDInsight Speicherkonto - Dateiberechtigung anpassen	88
D.9. HDInsight Speicherkonto - Dateilink kopieren	88

Abbildungsverzeichnis

2.1.2.1. Jupyter-Notebook in JupyterLab mit Python3-Kernel und Pandas	5
2.2.1.1. Komponenten des Hadoop-Ökosystems	6
2.2.1.2. Detallierter MapReduce Workflow ohne Combiner	7
2.2.1.3. High Level Hadoop-Architektur	8
2.2.2.1. Spark Resilient Distributed Datasets (RDD)	9
3.0.1. Magic Quadrant for Cloud Infrastructure and Platform Services	19
3.1.1. JupyterHub in Amazon EMR	21
3.2.1. Hochverfügbarkeitsinfrastruktur von Azure HDInsight	23
4.2.1. High Level - Architektur	26
4.2.2. Low Level - Architektur	27
5.2.1. Notwendiges Projekt in der Hetzner Cloud anlegen	34
5.2.2. Notwendiges API Token in der Hetzner Cloud anlegen	34
5.2.3. Hinterlegen der Floating-IPs in der DNS-Verwaltung	35
5.2.4. TXT-Records für DNS-Challenge mit Let's Encrypt	36
6.1.1. Anmeldemaske - JupyterHub Amazon EMR	44
6.1.2. PySpark Kernel auswählen	44
6.1.3. EMR - CSV-Datei mit PySpark analysieren	45
6.2.1. VM-Konfiguration HDInsight	46
6.2.2. HDInsight Jupyter-Notebook - Anmeldemaske	47
6.2.3. HDInsight Jupyter-Notebook - Startseite	48
6.2.4. HDInsight Jupyter-Notebook - CSV-Datei mit PySpark analysieren	49
6.3.1. Auflisten aller YARN-Container der IDE	50
6.3.2. Custom IDE - Hochladen der CSV-Datei in das HDFS	50
6.3.3. Custom IDE - CSV-Datei in /user mit PySpark analysieren	51
6.3.4. Custom IDE - Löschen von Dateien anderer Nutzer	51

Tabellenverzeichnis

2.1.1.1. Offline und Online IDE im Vergleich	4
2.2.3.1. MapReduce vs. Spark	11
4.3.1. Übersicht der zugrundeliegenden IT-Infrastruktur in der Hetzner Cloud .	29
5.1.1. Installationsphasen der Custom IDE	33
5.3.1. Ansible Tag check für verschiedene Inventory-Gruppen	37
5.4.1. Ausfallsicherheit	39
6.4.1. Gegenüberstellung der Funktionalitäten von Enterprise-Produkten und Custom IDE	52

Auflistungsverzeichnis

2.3.1.1. Inventory-Datei - YAML	13
2.3.1.2. Inventory-Datei -INI	14
2.3.2.1. Auslagerung der Inventory-Variablen	14
2.3.3.1. Dateistruktur einer Ansible-Rolle	15
2.3.4.1. DNS Flush mit ansible-Befehl	16
2.3.4.2. DNS Flush mit ansible-Befehl ohne Modul-Angabe	16
2.3.4.3. DNS Flush mit Ansible-Playbook	17
2.3.4.4. Ausführung setup.yml mit Ansible-Tags	17
2.3.5.1. Dateistruktur nach Anlegen einer leeren Ansible-Collection mit Ansible-Galaxy	18
5.1.1. Git-Repository - Klonen	30
5.1.2. Git-Repository - erste Ebene	31
5.1.3. Git-Repository - Namespaces der Ansible-Collections	31
5.1.4. Git-Repository - Ansible-Rollen des Namespaces hadoop	32
5.1.5. Installation der IDE	32
5.3.1. Überprüfung aller Komponenten der IDE auf ihre Funktionsfähigkeit . . .	36
5.3.2. Überprüfung aller notwendigen Variablen	36
5.5.1. LDAP-Konfiguration des Service JupyterHub	40

Abkürzungsverzeichnis

- Amazon EC2** Amazon Elastic Compute Cloud. 20, 22
- API** Application Programming Interface. 10, 29, 33, 34
- AWS** Amazon Web Services. 1, 3, 6, 12, 19, 20, 22, 23, 28, 42, 43, 45–47, 52, 53
- CPU** Central Processing Unit. 45, 51
- EMR** Elastic MapReduce. 1, 3, 5, 20–24, 28, 42, 43, 45, 46, 48, 49, 51–53
- FQDN** Fully Qualified Domain Name. 57
- GUI** Graphical User Interface. 12, 43
- HDFS** Hadoop Distributed Filesystem. 3, 6, 8–10, 25, 28, 37, 40, 49, 51, 53, 55, 56
- HDInsight** Azure Hadoop Insight. 1, 3, 5, 22–25, 28, 42, 45, 46, 49, 51–53
- HLD** High Level Diagramm. 25
- HTTP** Hypertext Transfer Protocol. 56
- HTTPS** Hypertext Transfer Protocol Secure. 25
- IAC** Infrastructure as Code. 3, 12
- IDE** Integrated Development Environment. VI, 1–5, 12, 23–30, 32–36, 39–42, 48–55
- IP** Internet Protocol Address. 26, 28, 35
- LDAP** Lightweight Directory Access Protocol. 5, 21, 24–28, 36, 37, 40, 48
- LLD** Low Level Diagramm. 25–27
- NSCD** Name Service Caching Daemon. 16
- PAM** Pluggable Authentication Modules. 21
- RDD** Resilient Distributed Datasets. IV, 9, 10

REST Representational State Transfer. 1, 3, 26, 29, 52, 53

RR Record Reader. 7

RW Record Writer. 8

S3 Simple Storage Service. 20, 22, 25, 43, 45, 51, 53, 55, 56

SQL Structured Query Language. 10, 21, 56

SSH Secure Shell. 12, 15, 21, 33, 43

TCP Transmission Control Protocol. 56, 57

VIP Virtuelle IP-Adresse. 57

VRRP Virtual Router Redundancy Protocol. 57

YARN Yet Another Resource Negotiator. 6, 8, 22, 24, 27, 28, 37, 39, 40, 48

ZKFC Zookeeper Failover Controller. 22

Kapitel 1.

Einleitung

Eine integrierte Entwicklungsumgebung (Integrated Development Environment) vereinfacht für Data Scientisten die Datenanalyse und Implementierung von Machine Learning Algorithmen. Ziel von Big Data ist die Informationsgewinnung aus Massendaten. Dementsprechend wird für die Echtzeitanalyse von Daten eine hohe Rechenleistung vorausgesetzt, die mit einer Offline IDE aufgrund der lokalen Hardware-Limitierung oft nicht gewährleistet werden kann [24, vgl. S. 2804]. In Relation dazu sind Online IDEs aufgrund der Informationsspeicherung in einer Cloud nicht auf eine bestimmte Hardware begrenzt. Weitere Vorteile einer Online IDE sind unter anderem das kollaborative Arbeiten an einem Big Data Projekt, der Zugriff auf die Daten von mehreren Geräten sowie die Einhaltung datenschutzrechtlicher Bestimmungen durch das Auslagern auf den jeweiligen Cloudanbieter [77, vgl. S. 45]. Vor allem die Open-Source IDE Jupyter bietet die Möglichkeit, in der Data Science populäre Sprachen wie beispielsweise Python, Julia, Scala oder R für die Verarbeitung von Daten zu nutzen [55, vgl. S. 99].

Sowohl Amazon Web Services (AWS) als auch Microsoft Azure bieten mit Elastic MapReduce (EMR) [7] und Azure Hadoop Insight (HDInsight) [51] Enterprise-Produkte zur verteilten Datenanalyse- und speicherung in einer Cloud an. In beiden Produkten lässt sich mit Jupyter arbeiten, dabei wird die Analyseumgebung auf unterschiedliche Art und Weise in ein zugrundeliegendes Hadoop-Ökosystem integriert und die zu analysierenden Daten mit der leistungsstarken Open-Source Engine Apache Spark prozessiert und verarbeitet. Im Unterschied zu JupyterHub on Hadoop von Jim Crist [38] weisen die Produkte mit Apache Livy, einen REST-Server für Apache Spark, eine zusätzliche Authentifizierungsschicht bei der Datenverarbeitung auf. Zudem sind nicht alle an der Interaktion zwischen Jupyter und dem Berechnungs-Cluster beteiligten Komponenten hochverfügbar. Trotz hoher Lizenskosten ist es nicht vorgesehen, dass Data Scientisten Einstellungen an der jeweiligen Big Data-Plattform vornehmen bzw. sind die im Hintergrund ablaufenden Prozesse nicht vollständig nachvollziehbar. In Folge dessen kann der

Erwerb der entsprechenden Lizzenzen für Institute, mittelständische Unternehmen sowie Studierende unattraktiv erscheinen. Nicht zuletzt ist unklar, ob die Lizenz durch eine Konfiguration der Big Data-Plattform ihre Gültigkeit verliert. Ziel dieser Masterarbeit ist deswegen, die gegebenen Enterprise-Produkte auf ihre Stärken und Schwächen zu analysieren und daraus abgeleitet das Konzept einer Open-Source Self-Service IT innerhalb einer cloudbasierten IDE zu entwickeln. Es ergeben sich somit folgende Fragestellungen, die im Verlauf dieser Arbeit zu beantworten sind:

- Wie unterscheiden sich die angebotenen Enterprise-Produkte untereinander und gegenüber dem vorgelegten Konzept von Jim Crist hinsichtlich ihrer Integration der Analyseumgebung Jupyter in das Hadoop-Ökosystem?
- Welche Vor- und Nachteile hat eine entwickelte Open-Source IDE in Relation zu den bereits existierenden konkurrierenden Produkten?

Um die Fragestellungen umfassend beantworten zu können, werden allen voran die theoretischen Grundlagen erläutert. Dabei wird zuerst eine Abgrenzung zwischen Offline und Online IDE vorgenommen. Anschließend werden grundlegende Begriffe zu der IDE Jupyter, den verwendeten Big Data Technologien sowie dem Automatisierungstool Ansible definiert. Nachdem die theoretischen Grundlagen der Arbeit eingeführt worden sind, werden die Enterprise-Produkte zur Datenanalyse im Big-Data Umfeld vorgestellt. Im vierten Kapitel wird die methodische Herangehensweise mit dem Konzept JupyterHub on Hadoop, einem Architekturüberblick sowie der verwendeten Cloud-Technologie erklärt. Nachdem die IDE im fünften Kapitel vorgestellt wird, findet im sechsten Kapitel der Vergleich mit den gegebenen Enterprise-Produkten statt. Die Arbeit endet mit einer Diskussion und einem Ausblick, basierend auf den gesammelten Erfahrungen im Entwicklungsprozess.

Kapitel 2.

Theoretischer Hintergrund

2.1. Integrated Development Environment

2.1.1. Offline vs. Online

IDEs werden sowohl offline als auch online genutzt. Je nach Anwendungsfall ist es sinnvoll, sich für eine der beiden Varianten zu entscheiden. In einer Offline IDE werden Analysemodelle zentralisiert, lokal und ohne funktionsfähige Internetverbindung getestet, wobei keine Abstimmung mit anderen Data Scientisten notwendig ist. Ein Nutzer besitzt bei einer Offline IDE in der Regel volle Administrationsrechte und ist dadurch nicht wie bei einer Online IDE von System-Administratoren der jeweiligen Umgebung und vorausgesetzten Berechtigungen abhängig. Zusätzlich ist eine Offline IDE vollkommen individuell auf persönlichen Vorlieben konfigurierbar [76, vgl. S. 182].

Demgegenüber muss eine Online IDE vom Data Scientisten nicht installiert werden. Programmierte Anwendungen werden mit riesigen Datenmengen getestet und ebenfalls skaliert, wobei der Zugriff auf die Daten von mehreren Geräten möglich ist. Der Data Scientist hat dadurch den Vorteil, dass bei Verlust seines PCs keine Daten verloren gehen. Zudem werden datenschutzrechtliche Bestimmungen an den jeweiligen Cloudanbieter ausgelagert. Mit einer Online IDE kann kollaborativ an einem Big Data Projekt gearbeitet werden. Es werden deswegen oft wichtige Dateien und Dokumente an einer für alle zugänglichen zentralisierten Stelle abgelegt [77, vgl. S. 45].

Zusammenfassend lässt sich feststellen, dass eine Offline IDE einem Data Scientisten vor allem viel Freiheit und Autonomität im gesamten Entwicklungsprozess bietet. Ein großer Nachteil ist jedoch, dass Data Scientisten aufgrund der Charakteristika von Big Data zwangsläufig mit riesigen Datenmengen ihre Analysemodelle entwickeln und testen müssen. Dieses Vorgehen ist mit einer Offline IDE wegen der limitierten lokalen Hardware

oft nicht möglich. Die nachfolgende Tabelle enthält eine zusammenfassende Übersicht der Vor- und Nachteile von Offline- und Online IDEs (vgl. Tabelle 2.1.1.1).

Offline vs. Online IDE		
	ja	nein
Offline Arbeiten	ja	nein
Zentrales Management	ja	nein
Autonomes Entwickeln	ja	nein
Notwendige Installation und Konfiguration	nein	ja
Keine Hardware-Limitierung	nein	ja
Zugriff von mehreren Geräten	nein	ja
Kollaboratives Arbeiten	nein	ja

Tabelle 2.1.1.1.: Offline und Online IDE im Vergleich

2.1.2. Das Jupyter-Projekt

Die IDE Jupyter kann sowohl offline als auch online genutzt werden (vgl. Kapitel 2.1.1) und zählt in den Bereichen Data Science und künstlicher Intelligenz zu einem Standard-Framework. Sie erlaubt Data Scientisten sowohl die Datenanalyse mit bereits implementierten Algorithmen als auch deren Visualisierung und Dokumentation [73, vgl. S. 53]. Jupyter wird als Projekt und Community bezeichnet, die beide den Anspruch der Weiterentwicklung von Open-Source Software und Standards mit mehreren Programmiersprachen innehaben.¹ Der Name Jupyter referenziert die drei populären Programmiersprachen Julia, Python und R. Aus dem Jupyter-Projekt sind die drei Produkte Jupyter-

¹ Aufgrund zahlreicher Vorteile gegenüber der klassischen Tabellenkalkulation gilt Jupyter mittlerweile als Alternative zu Microsoft Excel. Hierzu zählen beispielsweise die bessere interne Dokumentation, Wiederverwendbarkeit und geringere Fehleranfälligkeit[60].

Notebook, JupyterLab und JupyterHub hervorgegangen. Das Jupyter-Notebook stellt dabei die Kernkomponente des Projektes dar. Jupyter-Notebooks besitzen Ein- und Ausgabezellen, die jeweils Code, Text sowie Plots enthalten können und mit der Dateinamensendung ".ipynb" enden. Sie können über die Benutzeroberfläche in Formate wie HTML, LaTeX, PDF, Markdown, Python oder Präsentationsfolien konvertiert werden. Um die Arbeit mit verschiedenen Programmiersprachen zu ermöglichen, wird ein Jupyter-Notebook mit einem Kernel verknüpft. Standardmäßig wird der IPython-Kernel verwendet. JupyterLab ist ein webbasiertes Benutzer-Interface für die Arbeit mit mehreren Jupyter-Notebooks (vgl. Abbildung 2.1.2.1). Zusätzliche Bestandteile von JupyterLab sind Text-Editoren, Konsolen und JupyterLab-Erweiterungen wie das Versionierungstool Git [66].

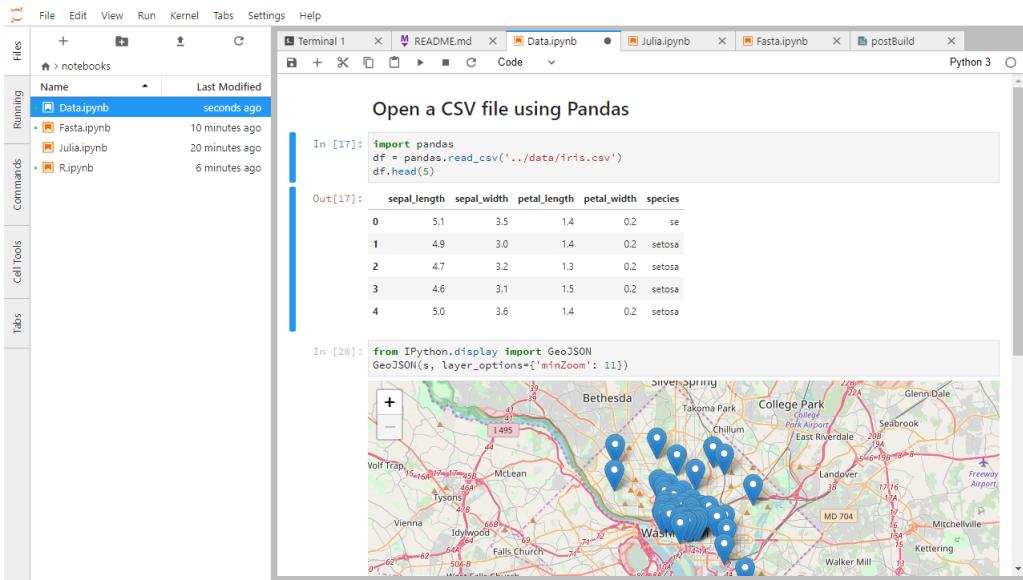


Abbildung 2.1.2.1.: Jupyter-Notebook in JupyterLab mit Python3-Kernel und Pandas
[66]

Mit JupyterHub greifen mehrere Nutzer auf in JupyterLab vorhandene Jupyter-Notebooks zu. Er dient als Proxydienst und kann in Verbindung mit dem Lightweight Directory Access Protocol (LDAP) und Kerberos (vgl. Kapitel 4.2) zur Authentifizierung und Autorisierung der jeweiligen Benutzer und Dienste verwendet werden [65, vgl. S. 59]. Die IDE Jupyter lässt sich mit dem Hadoop-Ökosystem (vgl. Kapitel 2.2.1) und dem In-Memory Big Data System Apache Spark (vgl. Kapitel 2.2.2) kombinieren. Mit Elastic MapReduce (EMR) (vgl. Kapitel 3.1) und Azure Hadoop Insight (HDInsight) (vgl. Kapitel 3.2) existieren Enterprise-Produkte, die diese drei Systeme miteinander vereinen und die für

die cloudbasierte Analyse von Massendaten genutzt werden.

2.2. Big Data-Systeme

2.2.1. Hadoop-Ökosystem

Grundlage der von AWS und Microsoft Azure zur Verfügung gestellten Enterprise-Produkte zur Datenanalyse (vgl. Kapitel 3) ist das Hadoop-Ökosystem. Apache Hadoop ist ein in Java geschriebenes Open Source Framework für die verteilte Analyse von Massendaten, das auf den von Google entwickelten Map-Reduce Algorithmus basiert. Durch die verteilte Architektur von Hadoop ist das System in der Lage, sehr große Datenmengen parallel zu verarbeiten. Hadoop gliedert sich in Yet Another Resource Negotiator (YARN) und Hadoop Distributed Filesystem (HDFS). Ersteres steht für die individuelle Ressourcenverwaltung in einem Big Data Cluster. Für Anwendungen und Benutzer können in Form von YARN Queues verfügbare Kapazitäten wie beispielsweise CPU, Arbeitsspeicher und GPU konfiguriert werden. Mit Letzterem ist die verteilte Speicherung sehr großer Datenmengen beschrieben [23, vgl. S. 188]. Durch die Zerlegung von Dateien in Datenblöcke werden diese auf alle teilnehmenden Knoten redundant verteilt. Zusätzlich wird die Hadoop-Architektur von mehreren in *Hadoop common* enthaltenen Java Bibliotheken, und Werkzeugen komplementiert [30] (vgl. Abbildung 2.2.1.1).

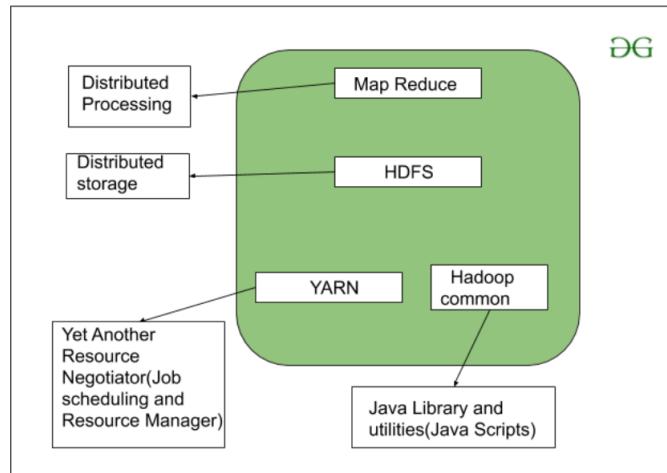


Abbildung 2.2.1.1.: Komponenten des Hadoop-Ökosystems
[30]

Der Map-Reduce Algorithmus besteht aus den zwei Operationen *Map* und *Reduce*, wo-

bei beide Vorgänge auf den gleichen oder unterschiedlichen Hadoop-Servern ausgeführt werden können. Mehrere Schritte sind notwendig, damit riesige Datenmengen vom MapReduce-Algorithmus verarbeitet werden. Zu Beginn zerlegt der Record Reader (RR) die eingehenden Daten in Datentupel, die \langle Schlüssel, Wert \rangle -Paare enthalten. Der Schlüssel ist dabei die Information über den Ort und der Wert über den Inhalt der Daten. Die $\text{Map}()$ -Funktion an sich ist benutzerdefiniert und prozessiert die vom RR produzierten Tupel mit mehreren Mappern (vgl. Abbildung 2.2.1.2).

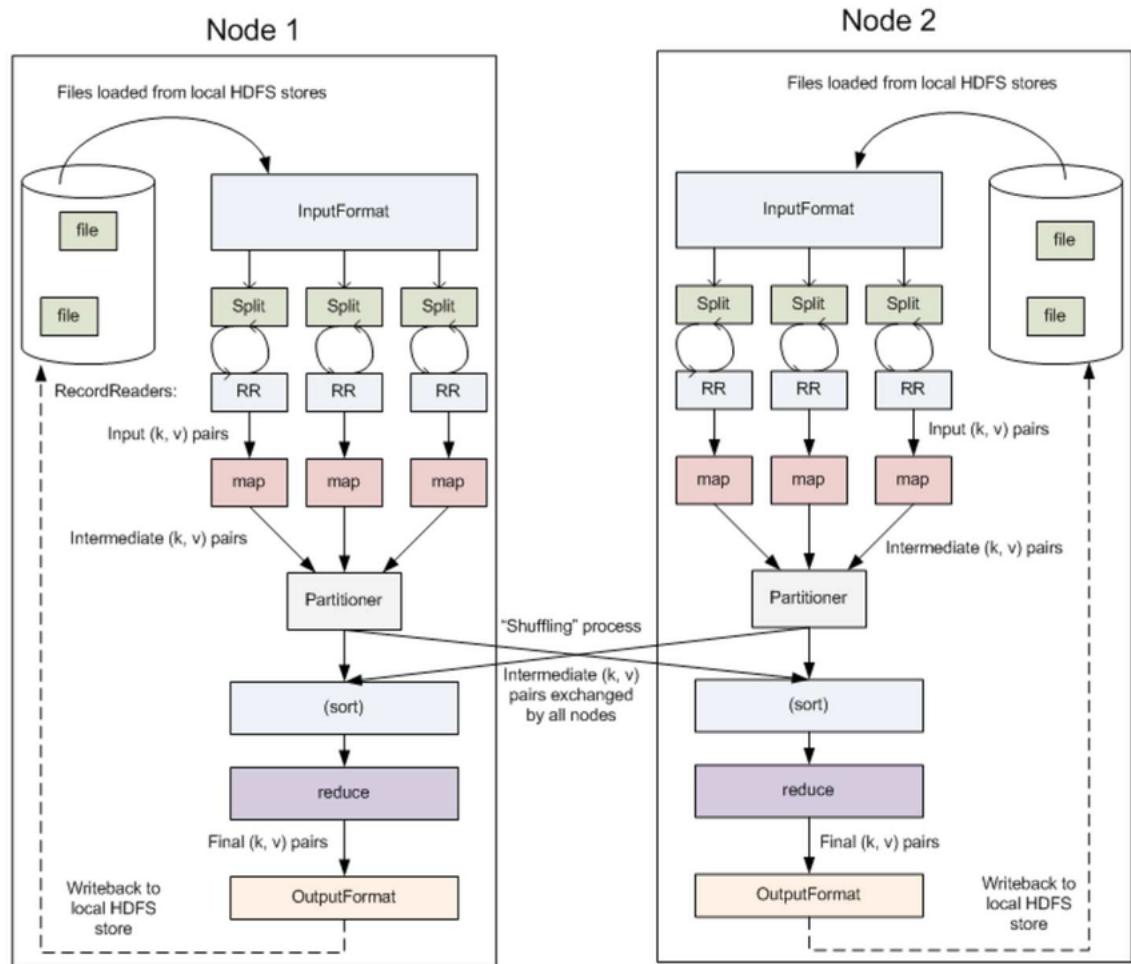


Abbildung 2.2.1.2.: Detallierter MapReduce Workflow ohne Combiner
[42]

So wird eine Datei mit 100 Datensätzen gleichzeitig von 100 Mappern verarbeitet. Genauso ist es möglich, dass die 100 Datensätze mit 50 Mappern bearbeitet werden. Das Hadoop-Framework entscheidet je nach Datenumfang und Verfügbarkeit der Speicherblö-

cke auf den Mapper-Servern über die Anzahl eingesetzter Mapper [64]. Mit dem *Combiner* und *Partitioner* gibt es noch zwei Prozesse, die zwischen der Map- und der Reduce-Funktion erfolgen. Ersterer ist ein optionaler Schritt und beschreibt die Reduktion der \langle Schlüssel, Wert \rangle -Paare auf den Mapper-Servern zu einer vereinfachten Form, bevor die Daten an den *Partitioner* weitergegeben werden. Dadurch sind insgesamt weniger Daten zu verarbeiten, was die Reorganisation und Sortierung der Datentupel erleichtert. Der darauf folgende Partitions-Prozess weist die erhaltenen \langle Schlüssel, Wert \rangle -Paare einem Reducer-Server zu. Der Hash-Wert eines Schlüssel aus dem Map-Prozess wird von dem Standard-Partitioner einer Partition zugeteilt. Die Anzahl der Partitionen stimmt mit denen der Reducer überein. Nach erfolgreicher Partitionierung werden die Daten an die Reducer gesendet [30]. Ein Reducer kann erst mit der Arbeit beginnen, wenn alle Mapper-Server fertig sind. Die Reduce-Funktion sortiert die berechneten Ergebnisse, fügt sie wieder zusammen und schreibt sie mit dem Record Writer (RW) zurück in das verteilte Dateisystem. Durch die Parallelisierung ist die Datenverarbeitung wesentlich schneller als in relationalen Datenbanken [23, vgl. S. 189].

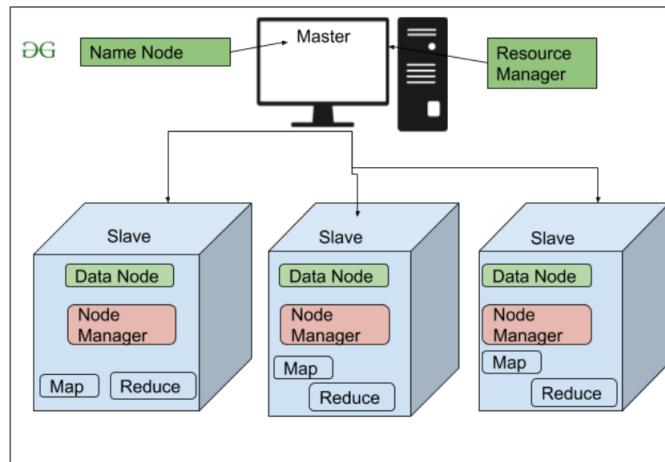


Abbildung 2.2.1.3.: High Level Hadoop-Architektur
[30]

YARN und HDFS basieren beide auf dem Master-Slave-Modell (vgl. Abbildung 2.2.1.3). So besteht ein Hadoop-Cluster aus einem YARN Resource Manager und HDFS Namenode, die wiederum einen oder mehrere YARN Nodemanagers und HDFS Datanodes verwalten und steuern. Die Koordination zwischen den einzelnen Hadoop-Komponenten wird dabei von dem Service Apache Zookeeper übernommen. Apache Hadoop stellt zusätzlich den Service Zookeeper Failover Controller zur Verfügung, der in einem hochverfügbar-

ren Hadoop-Cluster mit multiplen HDFS Namenodes für das Umschwenken bei einem ausfallenden HDFS Namenode zuständig ist [19].

2.2.2. Apache Spark

Als Erweiterung des Hadoop-Stacks zählt Apache Spark zu den populärsten Frameworks für die effiziente Analyse von Massendaten. Es wurde ursprünglich an der University of California in Berkeley im AMPLab² erfunden [26, vgl. S. 15]. Apache Spark ist ein In-Memory Big Data-System, das sich vor allem für die Verarbeitung von enormen Datensätzen eignet. Die Datenverarbeitung findet dabei im Arbeitsspeicher statt, während das Schreiben auf der Festplatten vermieden wird. Das ursprüngliche Ziel der Spark-Erfinder aus dem AMPLab war die Optimierung von Apache Hadoop MapReduce (vgl. Kapitel 2.2.1) [26, vgl. S. 16]. Grundlage von Spark ist das Konzept von sogenannten Resilient Distributed Datasets (RDD) (vgl. Abbildung 2.2.2.1).

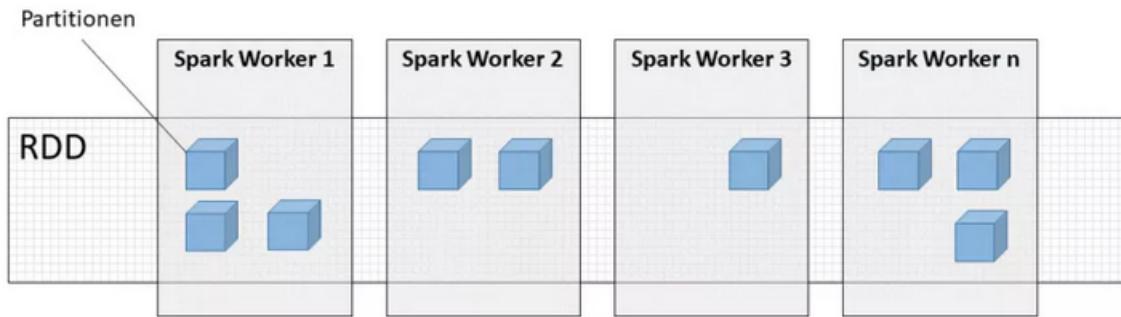


Abbildung 2.2.2.1.: Spark Resilient Distributed Datasets (RDD)
[43]

Mit *distributed* wird die Verteilung der Daten in Partitionen auf verschiedenen Knoten im Cluster beschrieben. *Resilient* steht dabei für das fehlertolerante Verhalten der Dateneinheiten, da verlorene oder beschädigte Partitionen auf den Spark-Workern wiederhergestellt werden können. Mit *Dataset* ist wiederum eine Sammlung von partitionierten Daten ohne Typisierung beschrieben. Alle Operationen in Spark werden auf den RDDs parallel ausgeführt. Sie können durch das Laden von verteilten Datenspeichern wie beispielsweise dem HDFS, Apache HBase oder durch das Auslesen von lokal gespeicherten Dateien erstellt werden [43]. In Java selber ist das Erzeugen von RDDs mit der *parallelize()* Methode aus einer Collection wie List oder Array möglich. RDDs unterstützen

² Das AMPLab steht für *Algorithms, Machines and People* und ist ein Labor der University of California in Berkeley mit Schwerpunkt auf Big Data-Analysen [10].

mit *Action* und *Transformation* zwei verschiedene Arten von Funktionen. Die *count()*-Funktion gibt beispielsweise die Anzahl der Elemente eines RDDs zurück, während die Funktion *union()* zwei RDDs zu einem RDD vereint. Jede in Spark verwendete Transformation verfolgt den Ansatz der *Lazy Evaluation*. Das bedeutet, dass eine Transformation nicht an der jeweiligen definierten Stelle ausgeführt wird, sondern erst beim Aufruf durch eine Action. Das führt insgesamt zu einer effektiveren Berechnung der Transformationen [54].

Nachteile von RDDs sind vor allem der Umgang mit strukturierten Daten. Deswegen wurden in der Spark-Version 1.3 *Spark DataFrames* eingeführt, die sich an den DataFrames in R oder denen von Pandas in Python orientieren. Es gibt zusätzlich zur *SparkDataFrame* API auch eine *Spark SQL* API, die Entwicklern das Abfragen von relationalen SQL-Abfragen ermöglicht. Ein *Spark DataFrame* besteht aus Spalten über deren Namen man einen leichten Zugriff auf den DataFrame hat. Daten verschiedener Datenspeicher (z.B. HDFS, JSON, Apache Hive oder Apache Parquet) können als DataFrame geladen bzw. gespeichert werden. Ein Query-Optimierer schreibt dabei relationale SQL-Abfragen um und führt so einen verbesserten Ausführungsplan der Abfragen aus. Genauso erhöht auch der Catalyst-Optimierer die Performanz von *Spark DataFrames*. Dabei handelt es sich um einen Prozess, der den effizientesten Plan zur Ausführung der definierten Datenoperationen ermittelt [54].

Eine Schwäche der DataFrames in Spark ist, dass sie keine Typisierung beim Kompilieren aufweisen und deswegen erst zur Laufzeit Fehler bzgl. der Typisierung der Daten auftreten. Zusätzlich ist die direkte Verarbeitung von unstrukturierten Daten nicht möglich. Um diese Problematik zu beheben, wurden mit Spark-Version 1.6 *Spark Datasets* erstmals eingeführt. Die bestehende *Spark DataFrame* API wurde um eine Typsicherheit und die Unterstützung objektorientierter Programme erweitert [54].

2.2.3. Apache Hadoop MapReduce vs. Apache Spark

Abschließend folgt eine Gegenüberstellung von Apache Hadoop MapReduce (vgl. Kapitel 2.2.1) mit Apache Spark (vgl. Kapitel 2.2.2), um die unterschiedlichen Aufgaben beider Systeme noch einmal zusammenzufassen (vgl. Tabelle 2.2.3.1). Beide Systeme haben die Verarbeitung von Massendaten zum Ziel, verfolgen dabei aber grundlegend andere Ansätze. Der MapReduce-Algorithmus wurde für die verteilte Verarbeitung von riesigen Datenmengen auf der Festplatte entwickelt. Im Kontrast dazu hat Apache Spark den Anspruch, die vergleichsweise langsamen Datenverarbeitung des MapReduce-Algorithmus

zu optimieren.

Worin unterscheiden sich Hadoop MapReduce und Spark?		
	Hadoop MapReduce	Spark
<i>Verarbeitung</i>	Festplatte	Arbeitsspeicher, Festplatte
<i>Geschwindigkeit</i>	geringe Performanz	hohe Performanz
<i>Einsetzbarkeit</i>	Batch-Verarbeitung	Batch/Realtime-Verarbeitung und Machine Learning
<i>Realtime</i>	nein	ja
<i>Robustheit</i>	Fehlertoleranz	Fehlertoleranz
<i>Sicherheit</i>	mehr Security-Features	weniger Security-Features

Tabelle 2.2.3.1.: MapReduce vs. Spark

Der MapReduce-Algorithmus liest und schreibt von der Festplatte, wodurch die In-Memory Datenverarbeitung von Apache Spark bis zu 100 Mal schneller ist. Während Apache Hadoop lediglich Batch-Verarbeitung anbietet, kann Apache Spark für Batch- und Graph-Verarbeitung sowie Machine Learning eingesetzt werden. MapReduce kann im Gegensatz zu Apache Spark aufgrund der vergleichsweise geringen Performanz ebenfalls keine Daten in Echtzeit verarbeiten. Beide Systeme sind gegenüber Ausfällen einzelner am Cluster beteiligter Knoten robust, da es sich um hochverfügbare und verteilte Big Data-Systeme handelt. Vor allem hinsichtlich IT-Security besitzt MapReduce in Relation zu Apache Spark mehr Features. Dementsprechend bietet Spark lediglich eine Authentifizierung mittels *Shared Secret* (Passwort-Authentifizierung) und Kerberos an. In Hadoop wiederum können Authentifizierung, Autorisierung, Auditing und Verschlüsselung konfiguriert werden. Dabei ist anzumerken, dass Spark bei einer Integration in Hadoop auch

die Security Features von Hadoop nutzen kann [3] [62].

2.3. Ansible

Für die Installation und Konfiguration der zu entwickelnden IDE gibt es mehrere Möglichkeiten. Management-Systeme wie Cloudera vereinfachen die Installation und Verwaltung von Big Data-Systemen [1, vgl. S. 170]. Dabei entstehen jedoch nicht nur Lizenz-Kosten, sondern vor allem Schwierigkeiten bei der Installation von zugeschnittenen Eigenentwicklungen wie im vorliegenden Fall. Da es sich um ein Open-Source Produkt handelt, sollte die Vorgehensweise zur Installation und Konfiguration der IDE in einem öffentlichen Git-Projekt in Form einer Infrastructure as Code (IAC) für jeden zugänglich sein. In Cloudera ist ein derartiges Vorhaben nicht möglich, da die Installation und Konfiguration von Big Data-Systemen über die GUI erfolgt. Deswegen wird für die Entwicklung der zugrundeliegenden IT-Infrastruktur das Automatisierungstool Ansible verwendet.

Ähnlich wie Puppet, Chef oder Salt ist auch Ansible ein Automatisierungs-Werkzeug zur Administration von Systemen [34]. Im Gegensatz zu den anderen üblichen Automatisierungstechniken kommt Ansible bei der Kontaktaufnahme mit zu administrierenden Maschinen mit den bereits auf den Systemen vorhandenen Rahmenbedingungen aus. So wird für die Kommunikation mit unixoiden Betriebssystemen der jeweilige SSH-Server und für Windows das Windows Remote-Management (WinRM) genutzt. Der klare Vorteil davon ist, dass die Installation und Konfiguration eigener Agenten entfällt [35, vgl. S. 5 f.]. Insgesamt setzt Ansible auf einen modularen Ansatz, da die jeweiligen Konfigurationen der Systeme von verschiedenen Ansible-Modulen übernommen werden. Die Module werden dabei von großen Software- und Hardware-Herstellern und der Ansible-Community stetig weiterentwickelt. Man kann damit einerseits Ad-hoc-Kommandos auf zahlreichen Maschinen ausführen oder diese in sogenannten Ansible-Playbook zusammenfassen und ganze IT-Landschaften einrichten.

Mit Ansible können ebenfalls IT-Infrastrukturen in der Cloud vollständig automatisiert aufgebaut und administriert werden. Sowohl für populäre Cloud-Anbieter wie AWS oder Microsoft Azure als auch für weniger populäre kostengünstiger Cloud-Dienste wie Hetzner (vgl. Kapitel 4.3) existieren entsprechende Ansible-Module [47]. Für die Arbeit mit Ansible ist es notwendig, einige Grundbegriffe einzuführen. Im Folgenden werden die für den Installationsprozess der IDE relevanten Ausdrücke kurz definiert, allerdings umfassen

diese nur einen Teil der Ansible-Terminologie.³

2.3.1. Inventory

Ansible benutzt zur Verwaltung von Zielsystemen eine sogenannte *Inventory*-Datei. Diese liegt in der Regel in einem *INI* oder *YAML*-Format vor. Innerhalb eines jeden Inventory sind alle notwendigen Zielsysteme definiert, die mithilfe von Ansible konfiguriert werden sollen. Theoretisch können alle Systeme in Gruppen und Untergruppen zusammengefasst und dadurch nur einen Teil der Zielsysteme mit Ansible über dasselbe Inventory administrieren werden. Eine gesonderte Stellung in der Hierarchie einer Inventory-Datei nimmt die Ansible-Gruppe *all* ein, da mit ihr alle Zielsysteme angesprochen werden. Die Inventory-Hierarchie lässt sich gut am YAML-Format veranschaulichen (vgl. Auflistung 2.3.1.1) [25].

```
all:
  children:
    inventory_group1:
      hosts:
        server1.example.com:
        server2.example.com
    inventory_group2:
      hosts:
        server3.example.com
    inventory_group3:
      hosts:
        server4.example.com
    inventory_group4:
      vars:
        jks_keystore_password: SECRET_PASSWORD
      children:
        inventory_group2:
        inventory_group3
```

Auflistung 2.3.1.1: Inventory-Datei - YAML

Auf der obersten Hierarchieebene ist die Ansible-Gruppe *all* definiert, der alle folgenden Gruppen untergeordnet sind. Die Unterordnung von Ansible-Gruppen wird in einem Inventory in YAML über den Eintrag *children*: geregelt. In den Ansible-Gruppen auf der zweiten Hierarchieebene (inventory_group1, inventory_group2, inventory_group3) legt der Eintrag *hosts*: die Zielsysteme (server1.example.com, server2.example.com, ser-

³ Für ein detailliertes Verständnis von Ansible ist es sinnvoll, sich mit der offiziellen Ansible-Dokumentation auseinanderzusetzen: <https://docs.ansible.com/>

ver3.example.com und server4.example.com) fest. Ansible-Gruppen auf der zweiten Hierarchieebene lassen sich dort ebenfalls zu einer weiteren Ansible-Gruppe (hier: inventory_group4) zusammenfassen. Das Zuweisen von Variablen zu Gruppen oder Zielsystemen in einem Inventory geschieht über den Eintrag *vars*: (hier: jks_keystore_password).

Beim INI-Format einer Inventory-Datei wird auf die für das YAML-Format notwendigen Einrückungen verzichtet. Damit mehrere Ansible-Gruppen zu einer weiteren Gruppe (hier: inventory_group4) zusammengefasst werden, ist das Suffix *:children* erforderlich. Die Zuordnung von Variablen wird mit dem Suffix *:vars* konfiguriert (vgl. Auflistung 2.3.1.2) [25].

```
[inventory_group1]
server1.example.com
server2.example.com

[inventory_group2]
server3.example.com

[inventory_group3]
server4.example.com

[inventory_group4:children]
inventory_group2
inventory_group3

[all:vars]
jks_keystore_password=SECRET_PASSWORD
```

Auflistung 2.3.1.2: Inventory-Datei - INI

2.3.2. Inventory-Variablen

Das Setzen von Variablen im Inventory kann bei einer Vielzahl von Variablen sehr unübersichtlich werden. Deswegen gibt es die Möglichkeit, diese für die Ansible-Gruppen und Host-Einträge in die separaten Ordner *group_vars* und *host_vars* auszulagern. Damit Ansible die ausgelagerten Variablen für die im Inventory enthaltenen Ansible-Gruppen und Host-Einträge interpretiert, müssen die jeweiligen Dateinamen übereinstimmen. Hier wird beispielsweise über die Datei *all.yml* eine Variable für alle Inventory-Einträge gesetzt (vgl. Auflistung 2.3.2.1) [36].

```
georg@georg - InfinityBook - Pro - 15 - v5:~/git/ansible$ cat group_vars/all.yml
```

```
jks_keystore_password: SECRET_PASSWORD
georg@georg - InfinityBook - Pro - 15 - v5:~/git/ansible$ tree
.
├── group_vars
│   ├── all.yml
│   ├── inventory_group1.yml
│   ├── inventory_group2.yml
│   ├── inventory_group3.yml
│   └── inventory_group4.yml
└── host_vars
    └── server1.example.com.yml
└── inventory

2 directories, 7 files
```

Auflistung 2.3.2.1: Auslagerung der Inventory-Variablen

2.3.3. Ansible-Rolle

Sollte Ansible auf einem Linux-Server installiert sein und der momentane Nutzer per SSH Zugriff auf die Zielsysteme eines Inventory haben, kann er alles Weitere für die Administration mit Ansible in einer sogenannten Ansible-Rolle definieren. Diese enthält auf dem Zielsystem auszuführende Schritte (*tasks*, *handlers*), zu kopierende Dateien (*files*, *templates*), sowie Abhängigkeiten (*meta*), Variablen (*defaults*, *vars*) und Tests (*tests*) für die jeweilige Ansible-Rolle (vgl. Auflistung 2.3.3.1) [57].

```
└── my_ansible_role
    ├── defaults
    │   └── main.yml
    ├── files
    ├── handlers
    │   └── main.yml
    ├── meta
    │   └── main.yml
    ├── README.md
    ├── tasks
    │   └── main.yml
    ├── templates
    ├── tests
    │   ├── inventory
    │   └── test.yml
    └── vars
        └── main.yml

11 directories, 15 files
```

Auflistung 2.3.3.1: Dateistruktur einer Ansible-Rolle

Dabei sind *handlers* prinzipiell das Gleiche wie *tasks*. Sie werden nur dann ausgeführt, wenn ein Schritt in *tasks* eine Systemänderung vorgenommen hat und über das Schlüsselwort *notify* aktiviert. Genauso liegen in *templates* im Unterschied zu *files* auf das Zielsystem zu kopierende Dateien, die zusätzlich Variablen enthalten. So lässt sich beispielsweise dafür sorgen, dass Konfigurationsdateien mit systemspezifischen Einträgen wie dem *inventory_hostname* versehen werden. Variablen in *defaults* und *vars* unterscheiden sich wiederum in ihrer Hierarchie. So wird der Wert einer Variable in *defaults* von der gleichnamigen Variable in *vars* überschrieben. Eine Ansible-Rolle kann ebenfalls eine *ansible.cfg* enthalten, mit der man Ansible individuell konfigurieren kann (siehe Anhang A.1).

2.3.4. Ansible-Playbook

In Ansible gibt es mit *ansible* und *ansible-playbook* zwei zentrale Kommandos, die ein Nutzer für die Administration einer IT-Infrastruktur verwendet. Mit dem *ansible*-Befehl lassen sich Ad-hoc-Befehle auf dem Zielsystem ausführen. Das ist vor allem nützlich, wenn man ein- und denselben Shell-Befehl auf vielen Zielsystemen ausführen muss. So kann man den DNS Cache eines Linux-Systems nach einer DNS-Anpassung beispielsweise wie folgt mit dem Name Service Caching Daemon (NSCD) auf allen Systemen eines Inventory beseitigen (vgl. Auflistung 2.3.4.1) [59].

```
georg@georg - InfinityBook - Pro - 15 - v5 : ~$ ansible all -m shell -a "nsqd -i hosts" --become
```

Auflistung 2.3.4.1: DNS Flush mit ansible-Befehl

Dabei übergibt der Nutzer dem *ansible*-Befehl mit der Option *-m* das entsprechende Ansible-Modul und mit der Option *-a* die Parameter des Ansible-Moduls. Um root-Rechte auf dem Zielsystem zu besitzen, muss zusätzlich die Option *--become* angegeben werden. In diesem Fall kann der Ansible-Befehl auch ohne die Angabe des Ansible-Moduls erfolgen, da es sich bei dem *shell*-Modul um das Default-Modul des Ansible-Befehls handelt (vgl. Auflistung 2.3.4.2).

```
georg@georg - InfinityBook - Pro - 15 - v5 : ~$ ansible all -a "nsqd -i hosts" --become
```

Auflistung 2.3.4.2: DNS Flush mit ansible-Befehl ohne Modul-Angabe

Für die Konfiguration eines Zielsystems ist es üblich, mehrere dafür notwendige Schritte in einem sogenannten *Ansible-Playbook* zusammenzufassen. So lässt sich der oben genannte Befehl in folgendes Ansible-Playbook übersetzen, das dann anschließend mit dem Befehl *ansible-playbook* ohne die Angabe von weiteren Parametern ausgeführt wird (vgl. Auflistung 2.3.4.3) [58].

```
georg@georg - InfinityBook - Pro - 15 - v5:~/git/ansible$ cat my_ansible_playbook.yml
- hosts: all
  become: yes
  tasks:
    - name: flush DNS cache on host
      shell: "nsqd -i host"
georg@georg - InfinityBook - Pro - 15 - v5:~/git/ansible$
georg@georg - InfinityBook - Pro - 15 - v5:~/git/ansible$
georg@georg - InfinityBook - Pro - 15 - v5:~/git/ansible$ ansible-playbook
  my_ansible_playbook.yml
```

Auflistung 2.3.4.3: DNS Flush mit Ansible-Playbook

Bei einer umfangreichen Automatisierung lohnt es sich, sogenannte *Ansible-Tags* zur Limitierung der Schritte zu setzen. Beispielsweise werden mit dem Ansible-Playbook *setup.yml* (siehe Anhang A.2) wie folgt Server in der Hetzner-Cloud bestellt, für DNS konfiguriert und der Teil der Automatisierung für das Reverse DNS ausgeschlossen (vgl. Auflistung 2.3.4.4) [63].

```
georg@georg - InfinityBook - Pro - 15 - v5:~/git/ansible$ ansible-playbook setup.yml --
  tags hetzner,dns --skip-tags reverse_dns
```

Auflistung 2.3.4.4: Ausführung *setup.yml* mit Ansible-Tags

2.3.5. Ansible-Collection

Können mehrere Ansible-Rollen einem übergeordneten Thema zugewiesen werden, fasst man diese in der Regel in einer sogenannten *Ansible-Collection* zusammen. Verschiedene Ansible-Collections werden dann wiederum einem übergeordneten *Namespace* zugeteilt. Namespaces stellen in der Praxis unter anderem verschiedene Konzerne, Entwicklerteams oder auch Organisationen innerhalb eines Konzernes dar. Eine Ansible-Collection enthält neben mehreren Ansible-Rollen oft auch verschiedene *Ansible-Plugins*. Ein Ansible-Plugin ist beispielsweise ein selbst entwickeltes Ansible-Modul, das unabhängig von den

Standard-Ansible-Modulen bei Einbindung⁴ der Ansible-Collection genutzt werden kann (vgl. Auflistung 2.3.5.1) [72].

```
georg@georg - InfinityBook - Pro - 15 - v5 :~/git/ansible$ tree
.
└── my_namespace
    └── my_collection
        ├── docs
        ├── galaxy.yml
        └── plugins
            ├── README.md
            ├── my_ansible_plugin_1
            └── my_ansible_plugin_2
        ├── README.md
        └── roles
            ├── my_ansible_role_1
            ├── my_ansible_role_2
            └── my_ansible_role_1

5 directories, 3 files
```

Auflistung 2.3.5.1: Dateistruktur nach Anlegen einer leeren Ansible-Collection mit Ansible-Galaxy

⁴ Eine oder mehrere Ansible Collections werden in einer Ansible-Rolle über den optionalen Ordner *collections* eingebunden.

Kapitel 3.

Big Data in der Cloud - Vorstellung von Analyseplattformen

Wenn es um die Bereitstellung von Ressourcen für die Berechnung von Analysemodellen in der Cloud geht, gibt es mittlerweile ein gutes Dutzend von Anbietern. Besonders stechen dabei die beiden Cloudanbieter Amazon Web Services (AWS) und Microsoft Azure hervor. Sowohl AWS als auch Microsoft Azure befinden sich in Gartners aktuellen "Magic Quadrant for Cloud Infrastructure and Platform Services" in der oberen rechten Ecke des Leader-Quadranten und werden damit ausgezeichnet für ihre Fähigkeit zur Umsetzung und Vollständigkeit der Vision (vgl. Abbildung 3.0.1).

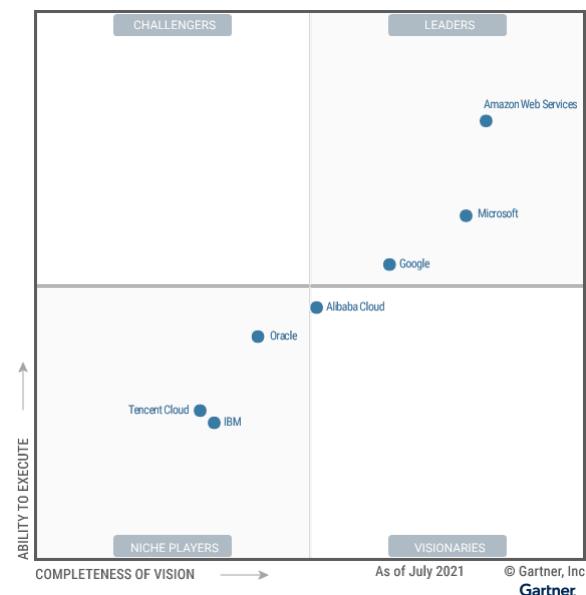


Abbildung 3.0.1.: Magic Quadrant for Cloud Infrastructure and Platform Services
[56]

AWS sichert sich damit zum elften Mal in Folge den ersten Platz im Bereich Cloud

Computing, dicht gefolgt von Microsoft Azure, Google und Alibaba Cloud [56]. Deswegen ist es sinnvoll, sich im weiteren Verlauf dieser Arbeit auf die von AWS und Microsoft Azure zur Verfügung gestellten Analyseplattformen zu konzentrieren.

3.1. Amazon Elastic MapReduce

Amazon Elastic MapReduce (EMR) wird von AWS bereitgestellt und gehört zu den branchenführenden Big Data-Plattformen in der Cloud. EMR stellt für die Analyse von Massendaten neben dem Hadoop-Framework (vgl. Kapitel 2.2.1) mehrere Open-Source Tools wie Apache Spark (vgl. Kapitel 2.2.2), Apache HBase, Apache Hive, Apache Flink, Apache Hudi sowie Apache Presto bereit. Die Grundidee ist, einen auf dem Hadoop-Ökosystem basierenden Web-Service einfach bereitzustellen und damit die Analyse von Massendaten für jeden mit einer Kreditkarte nutzbar zu machen. Die Installation und Konfiguration der Big Data-Plattform sowie eine entsprechende Wartung wird dabei auf den Cloudanbieter Amazon ausgelagert. Amazon argumentiert, dass sich dadurch kundenseitig unter anderem Kosten für das Bereitstellen der notwendigen IT-Infrastruktur und IT-Security sowie entsprechende Personalkosten einsparen lassen. Amazon EMR nutzt als Ablage zu analysierender Daten den Amazon Simple Storage Service (S3), wobei die Berechnung auf mehreren skalierbaren Instanzen in der Amazon Elastic Compute Cloud (Amazon EC2) ausgeführt werden. Das Hadoop-Framework inklusive weiterer Tools aus dem Hadoop-Stack ist auf diesen Amazon EC2-Instanzen installiert und konfiguriert. Entsprechend berechnet Amazon einen Kostenaufschlag im Vergleich zur klassischen Amazon EC2-Instanz ohne diese bereits gegebenen Rahmenbedingungen. Je nach Größe der Amazon EMR-Instanz und der Nutzungsdauer variieren die Kosten [7, vgl. S 5 ff.].

EMR bietet mehrere Optionen für die Arbeit mit Jupyter-Notebooks an. Einerseits sind das spezielle EMR-Notebooks, die im Gegensatz zu einem traditionellen Notebook ohne eigenen Server auskommen. Der Inhalt der Notebooks wird über einen Client ausgeführt und von den Daten getrennt in einem S3-Bucket zur sicheren Speicherung und flexiblen Wiederherstellung abgelegt. Die Ausführung des im Notebook definierten Codes erfolgt über einen Kernel im EMR-Cluster. Es ist möglich, mit einem EMR-Notebook mit mehreren EMR-Clustern zu arbeiten. Gleichermassen können mehrere Benutzer das gleiche EMR-Cluster mit multiplen Notebooks nutzen und diese über den S3-Bucket untereinander austauschen [4]. Abgesehen von den EMR-Notebooks wird in EMR ebenfalls JupyterHub in ein EMR-Cluster integriert (vgl. Kapitel 2.1.2), um die Entwicklung von

Analysemodellen in Jupyter-Notebooks zu gewährleisten (vgl. Abbildung 3.1.1). Bei der Installation von JupyterHub in EMR startet auf dem Master-Knoten¹ des EMR-Clusters ein Docker-Container mit allen Komponenten für Jupyter und Sparkmagic. Die Authentifizierung der Benutzer zu dem gestarteten JupyterHub erfolgt über das Lightweight Directory Access Protocol (LDAP) oder Pluggable Authentication Modules (PAM) mit einer SSH-Verbindung [5].

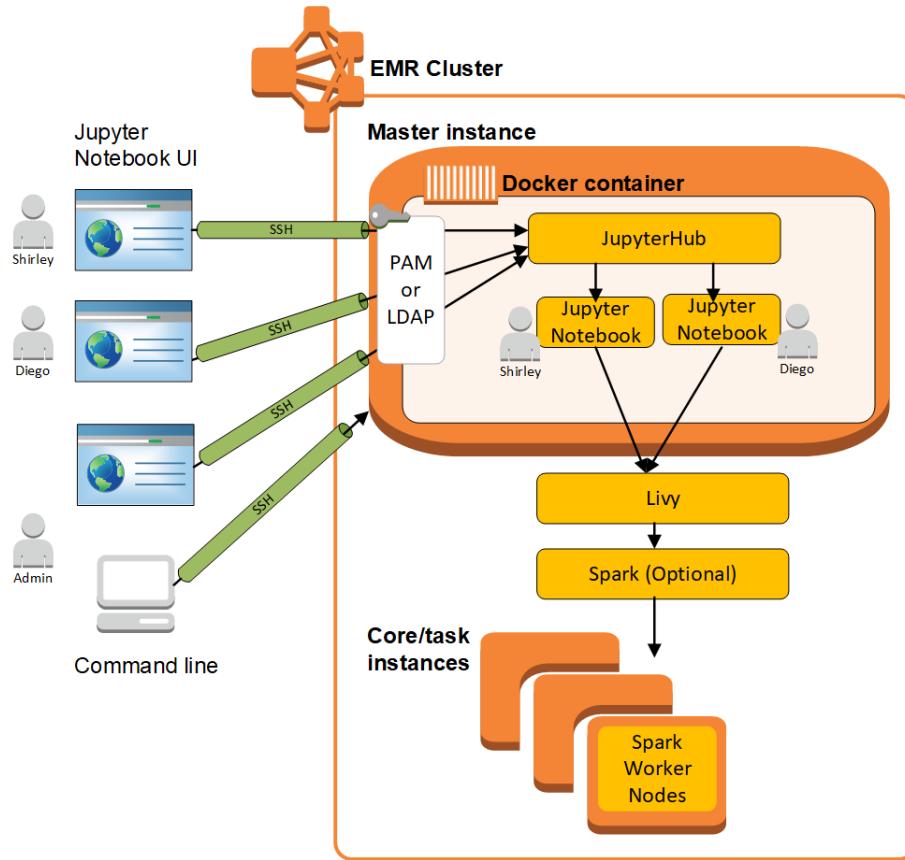


Abbildung 3.1.1.: JupyterHub in Amazon EMR
[5]

Standardmäßig stehen im JupyterHub von EMR der Python-3- sowie PySpark- und Spark-Kernel bereit und werden für die Ausführung von Spark-Code und interaktiven SQL-Abfragen genutzt. Es lassen sich jedoch auch zusätzliche Kernel im Docker-Container manuell hinzufügen. Die Abfragen werden anschließend auf den Worker-Knoten über Apache Livy (vgl. Kapitel 1) ausgeführt. Sowohl Spark als auch Livy werden bei der

¹ Die Nutzung von multiplen Master-Instanzen zwecks Hochverfügbarkeit ist sowohl für das Hadoop-Ökosystem als auch JupyterHub gegeben [7].

Erstellung eines EMR-Clusters mit Jupyter automatisch installiert. Es ist anzumerken, dass Apache Livy keine Authentifizierung anbietet und der Zugriff von Livy auf Spark in diesem Fall ohne Authentifizierung erfolgt [6]. Eine Authentifizierung ist in Livy nur über ein gesondertes Knox-Gateway möglich, das hier allerdings zusätzlich konfiguriert werden muss [41]. Neben den EMR-Notebooks und EMR-JupyterHub lässt sich auch in EMR-Studio mit Jupyter-Notebooks kollaborativ über einen S3-Bucket an einem Big Data-Projekt arbeiten [8].²

3.2. Azure Hadoop Insight

Ähnlich wie AWS mit Elastic MapReduce (EMR) (vgl. Kapitel 3.1) stellt auch der Cloudanbieter Microsoft Azure mit Azure Hadoop Insight (HDInsight) einen auf mehreren Open-Source-Tools basierenden Analysedienst bereit. Die Grundidee stimmt dabei mit der von AWS überein, mittels Web-Service sollen mehrere Services aus dem Hadoop-Stack für die Analyse von Massendaten genutzt werden. Dazu zählen abgesehen vom Hadoop-Ökosystem (vgl. Kapitel 2.2.1) unter anderem Apache Spark (vgl. Kapitel 2.2.2), Apache Hive, Apache HBase, Apache Kafka, Apache Storm, Apache Oozie und Apache Zookeeper. Für die Registrierung wird ebenfalls eine Kreditkarte vorausgesetzt. Microsoft Azure argumentiert bei der Vermarktung von HDInsight mit Kosteneinsparungen durch die Auslagerung der IT-Infrastruktur auf den Cloudanbieter. Vergleichsweise zu den Amazon EC2-Instanzen von AWS stellt Microsoft Azure für ein HDInsight-Cluster die sogenannten Azure Virtual Machines bereit, wobei als Datenablage der Azure Data Lake Storage dient. Auch hier variieren die Kosten je nach Nutzung und Größe der Instanzen.

In HDInsight heißen die Master-Instanzen *Headnodes*. Genauso wie in EMR ist auch hier eine Hochverfügbarkeit gegeben (vgl. Abbildung 3.2.1). Mehrere Headnodes können für ein HDInsight-Cluster konfiguriert werden. Fällt einer der Headnodes aus, übernimmt der Master Failover Controller³ das Umschwenken auf den primären oder sekundären Headnode. Die Worker-Knoten erhalten über den Service Apache Zookeeper die Information über den momentanen aktiven und passiven Headnode. Dadurch sind alle beteiligten Services (YARN, Auftragsverlaufsserver für Hadoop MapReduce und Apache Livy) für die Abfragen gegen das auf den Worker-Knoten zu verortende Apache Spark hochverfügbar.

² Da es sich bei der IT-Infrastruktur von EMR-Studio genauso wie bei dem EMR-JupyterHub um eine Container-Lösung als Frontend zu dem davon isolierten Hadoop-Ökosystem handelt, wird dies nicht zusätzlich betrachtet.

³ Die Funktionalität ist hier ähnlich zu der des ZKFCs (vgl. Kapitel 2.2.1)

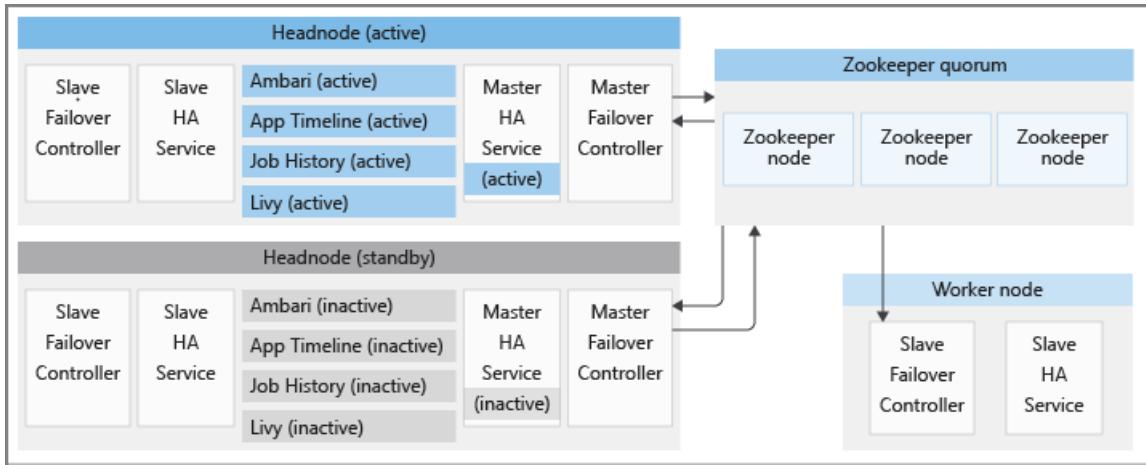


Abbildung 3.2.1.: Hochverfügbarkeitsinfrastruktur von Azure HDInsight
[51]

In HDInsight kann auf zwei verschiedenen Wegen mit Jupyter-Notebooks⁴ gearbeitet werden. Einerseits mit einer interaktiven Jupyter-Shell in der Azure Cloud und andererseits mit einer lokalen Jupyter-Installation, die man für eine Verbindung mit einem HDInsight-Cluster einrichten muss. Damit stellt HDInsight gegenüber EMR keine Integration eines JupyterHubs (vgl. Kapitel 3.1) in ein zugrundeliegendes HDInsight-Cluster bereit. Ähnlich wie die EMR-Notebooks von AWS nutzt auch die interaktive Jupyter-Shell von Microsoft im HDInsight-Cluster bereitgestellte Kernel. Insgesamt drei Kernel (PySpark, PySpark3, Spark) unterstützen die Sprachen Python2, Python3 und Scala. Besteht der Wunsch nach anderen Sprachen oder Bibliotheken, die nicht durch diese Kernel zur Verfügung stehen, wird eine lokale Jupyter-Installation empfohlen. Microsoft Azure argumentiert mit der insgesamt einfacheren Handhabung einer lokalen Installation gegenüber dem Konfigurationsaufwand für die Einbindung in ein HDInsight-Cluster. Die Nutzer wären so in der Lage, den Analysecode ihrer lokalen IDE über Git zu versionieren und so kollaborativ auch mit mehreren Nutzern an einem Big Data-Projekt zu arbeiten⁵. Eine lokale Jupyter-Installation ermöglicht die Verbindung zu verschiedenen HDInsight-Clustern [52]. Damit verfolgt Microsoft Azure eine andere Strategie als AWS, da es hier einen Teil der für die Analysemodelle benötigten IT-Infrastruktur an den Endnutzer auslagert.

⁴ HDInsight bietet ebenfalls an, Daten in einem Spark-Cluster mittels Apache Zeppelin-Notebooks zu analysieren [50]. Da der Fokus dieser Arbeit auf der Arbeit mit Jupyter-Notebooks liegt (vgl. Kapitel 1), wird dies von der weiteren Analyse ausgeschlossen.

⁵ Anzumerken ist hierbei, dass in diesem Fall alle Livy-Sitzungen über den gleichen technischen Livy-Nutzer laufen, da Apache Livy keine Authentifizierung anbietet (vgl. Kapitel 3.1)

Kapitel 4.

Planung der zugrundeliegenden IT-Infrastruktur

Nach der Einführung der theoretischen Grundlagen sowie der Vorstellung der Analyseplattformen folgt nun die methodische Herangehensweise zur Beantwortung der Fragestellungen. Allen voran wird das Konzept *JupyterHub on Hadoop* (vgl. Kapitel 4.1) näher erläutert und mit den Enterprise-Produkten EMR und HDInsight unmittelbar verglichen. Anschließend wird die gewählte IT-Architektur (vgl. Kapitel 4.2) näher beschrieben. Das Kapitel endet mit der Planung der praktischen Umsetzung mit der von Hetzner zur Verfügung gestellten Cloud-Technologie (vgl. Kapitel 4.3).

4.1. JupyterHub on Hadoop

Mit dem von Jim Crist entwickelten Konzept *JupyterHub on Hadoop* [38] besteht die Möglichkeit, die IDE Jupyter (vgl. Kapitel 2.1.2) in das Hadoop-Ökosystem (vgl. Kapitel 2.2.1) vollständig zu integrieren. Die Endnutzer melden sich über eine Webadresse an einem JupyterHub-Knoten (vgl. Kapitel 2.1.2) an.¹ Anschließend wird dem Nutzer über YARN ein YARN-Container zugewiesen. Der Data Scientist kann dadurch in einer skalierbaren und hoch verfügbaren Big Data-Umgebung seine Analysemodelle entwickeln und ist zusätzlich vollkommen im Sicherheitssystem von Apache Hadoop integriert und authentifiziert. Für die Nutzung von JupyterHub on Hadoop muss eine JupyterHub-Instanz installiert und für die Kommunikation mit einem bestehenden Hadoop-Ökosystem konfiguriert werden. Im Unterschied zu EMR befindet sich der Service JupyterHub auf der jeweiligen Maschine nicht in einem Docker-Container, sondern direkt auf dem verwendeten Betriebssystem. Die Installation von Jupyter erfolgt wie auch bei HDInsight über

¹ Die Autorisierung und Authentifizierung der Benutzer erfolgt hier in der Regel über das Lightweight Directory Access Protocol (LDAP).

den Installations-Manager Anaconda. Im Vergleich zu HDInsight kann die JupyterHub-Instanz und das Hadoop-Cluster sowohl offline als auch online genutzt werden.²

Generell gibt es bei der Installation und Konfiguration von JupyterHub on Hadoop mehrere Optionen bzgl. der IDE Jupyter. Die JupyterHub-Instanz kann mit den gleichen Sicherheitszertifikaten für das Hypertext Transfer Protocol Secure (HTTPS) ausgestattet werden, die auch vom Hadoop-Ökosystem genutzt werden [71]. Eine Autorisierung und Authentifizierung aller beteiligten Dienste und Benutzer wird über Kerberos und LDAP gewährleistet. Mit verschiedenen Content Managern wird als Ablageort für die Jupyter-Notebooks das HDFS, ein S3-Bucket oder eine PostgreSQL-Datenbank gewählt [2]. JupyterLab (vgl. Kapitel 2.1.2) kann als Frontend aktiviert werden [71], das unter anderem die Anbindung an Dask, Git oder GitHub ermöglicht [37]. Da die Nutzer in JupyterHub on Hadoop vollständig im Hadoop-Ökosystem integriert sind, interagieren sie direkt mit Apache Spark. Im Gegensatz zu den vorgestellten Enterprise-Produkten wird hier also keine zusätzliche Authentifizierungsschicht mit Apache Livy (vgl. Kapitel 3.1 und 3.2) benötigt. Dies vereinfacht insgesamt die IT-Architektur.

Zusammenfassend hat man bei der Nutzung von JupyterHub on Hadoop mit einem bestehenden Hadoop-Cluster weitaus mehr Möglichkeiten, mit der IDE Jupyter Daten zu analysieren und seine Analysemodelle weiterzuentwickeln. Zusätzlich ist eine vollständige Authentifizierung der Endnutzer über das Sicherheitssystem von Apache Hadoop gegeben. Dem stehen ein hoher Konfigurationsaufwand und ein dafür notwendiges tiefergehendes Verständnis für den Aufbau von Big Data-Systemen entgegen. Im weiteren Verlauf dieser Arbeit wird dieses Konzept zur Entwicklung der IDE aufgegriffen (vgl. Kapitel 4.2).

4.2. Architekturüberblick

Für die praktische Umsetzung und Entwicklung der Online IDE muss vorab eine Architektur definiert werden. Die Architektur einer IT-Infrastruktur wird auf verschiedenen Ebenen betrachtet. Es ist üblich, dabei zwischen einem High Level Diagramm (HLD) und Low Level Diagramm (LLD) zu unterscheiden [49]. Das folgende HLD beschreibt dabei auf einer höheren, abstrakten Ebene die Funktionsfähigkeit der IDE. Wie in JupyterHub on Hadoop (vgl. Kapitel 4.1) vorgesehen, meldet sich der Endbenutzer über eine Web-

² Bei HDInsight ist vorgesehen, eine lokal Notebook-Instanz mit dem Spark-Cluster in der Azure-Cloud zu verbinden. (vgl. Kapitel 3.2)

adresse an einem JupyterHub-Knoten an. Nach der Anmeldung wird der Nutzer an JupyterLab weitergeleitet und kann im Hadoop-Ökosystem arbeiten. Im Unterschied zu den vorgestellten Enterprise-Produkten (vgl. Kapitel 3) ist hier der zusätzlicher REST-Server Apache Livy nicht für die Arbeit mit Spark notwendig. Die Endnutzer sind vollständig im Sicherheitssystem von Apache-Hadoop integriert und authentifiziert (vgl. Abbildung 4.2.1).

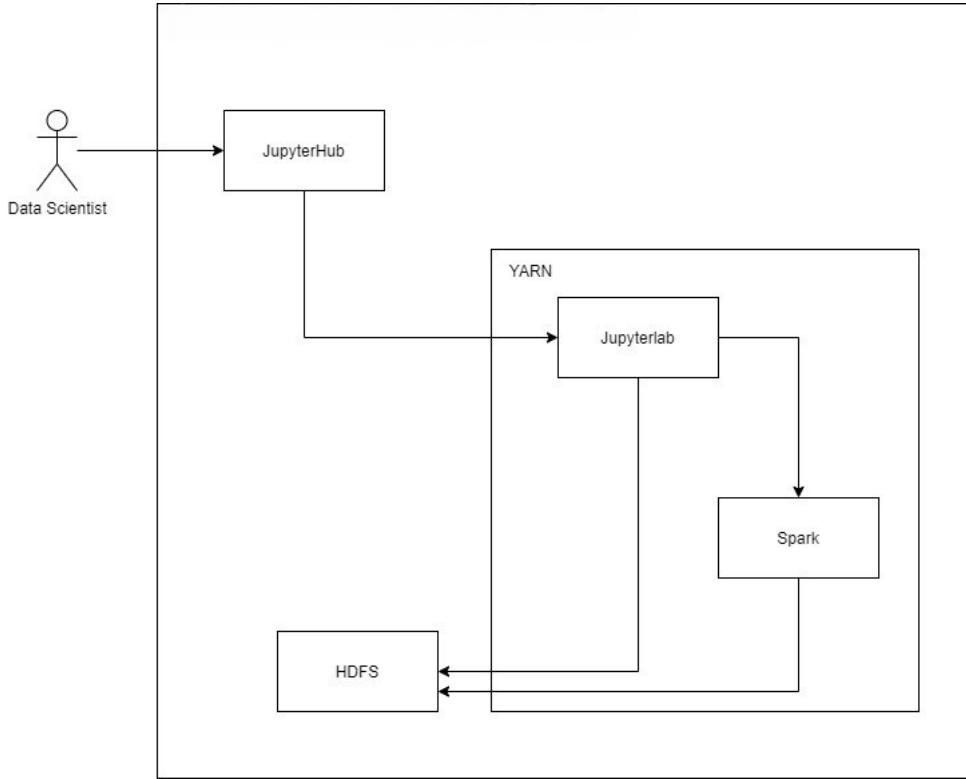


Abbildung 4.2.1.: High Level - Architektur

Der detaillierter Ablauf des Authentifizierungs- und Autorisierungsprozesses aller an der IDE beteiligten Dienste und Benutzer wird erst auf der Ebene des LLDs deutlich. Dementsprechend ist es aufgrund der zahlreichen Abhängigkeiten weitaus komplexer (vgl. Abbildung 4.2.2). Als Frontend dienen zwei JupyterHub-Instanzen³, um eine Hochverfügbarkeit zu gewährleisten. Beide Instanzen sind über eine gemeinsame Adresse erreichbar. Dabei handelt es sich um eine virtuelle IP, die über den Keepalived-Service ausfallsicher geschwenkt wird, sollte einer der Knoten ausfallen. Zur Authentifizierung und Autorisierung der Benutzer werden ebenfalls zwei LDAP-Instanzen genutzt. Der LDAP-Service

³ Als Backend für die JupyterHub-Instanzen wird ein ausfallsicheres PostgreSQL-Cluster über ein auf beiden Instanzen vorhandenen HAProxy verwendet.

selbst ist nicht hochverfügbar, deswegen wird auf den JupyterHub-Instanzen ein HA-Proxy installiert. Der HAProxy leitet die Anfragen an einen der LDAP-Server weiter⁴. Die Jupyter-Notebooks werden in einem YARN-Container gestartet. Genauer betrachtet wird der YARN-Container in einem der YARN-Nodemanager gestartet und über den YARN-Resourcemanager verwaltet und gesteuert. Die Authentifizierung und Autorisierung der jeweiligen an der IDE beteiligten Services erfolgt über zwei hochverfügbare Kerberos-Instanzen. Da sich die YARN-Resourcemanager und YARN-Nodemanager untereinander und gegenseitig über zwei verschiedene Kerberos-Server authentifizieren und die anfragenden Endnutzer über ebenfalls zwei LDAP-Instanzen abgefragt werden, ergeben sich im LLD die folgenden zahlreichen Abhängigkeiten.

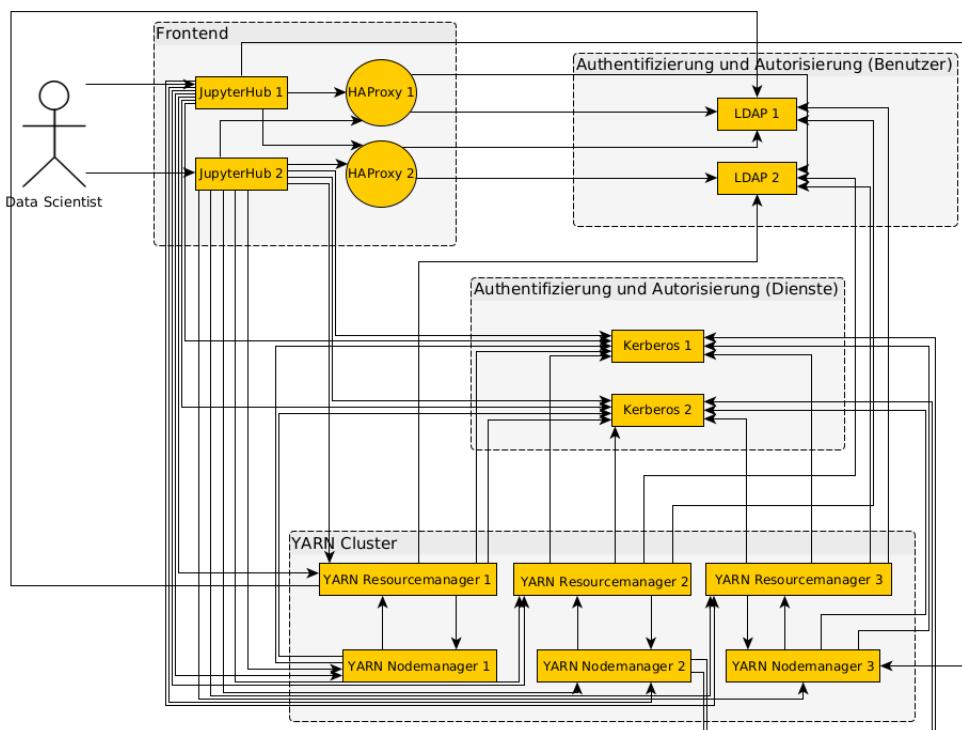


Abbildung 4.2.2.: Low Level - Architektur

⁴ Der HAProxy ist hier nicht zwangsläufig für eine Hochverfügbarkeit des LDAP-Services notwendig. Es können in der JupyterHub-Konfiguration auch mehrere LDAP-Server hinterlegt werden. Er sorgt jedoch für eine gleichmäßige Auslastung der LDAP-Server, da ansonsten in der Regel der erste angegebene LDAP-Server vom Service JupyterHub gewählt wird.

4.3. Hetzner Cloud

Für die zugrundeliegende IT-Infrastruktur werden virtuelle Maschinen in der Cloud bestellt und die IDE anschließend mittels Ansible (vgl. Kapitel 2.3) installiert und konfiguriert. Das Hosten von virtuellen Maschinen in der Hetzner Cloud ist weitaus kostengünstiger als bei AWS oder Microsoft Azure. Dies stimmt gerade überein mit dem Ziel der Arbeit, eine Open-Source IDE für Institute, mittelständische Unternehmen sowie Studierende mit einem in der Regel verhältnismäßig kleinerem Kostenbudget zu entwickeln (vgl. Kapitel 1). Als Folge der geringen Kosten werden weitaus weniger Cloud-Lösungen angeboten. Dies steht jedoch nicht im Widerspruch zum Anliegen dieser Arbeit, mit einer vollständig selbstentwickelten IDE bereits fertiggestellte Cloud-Lösungen wie beispielsweise EMR und HDInsight (vgl. Kapitel 3) zu ersetzen.

Die zugrundeliegende Cloud-Technologie Hetzner wird von der Hetzner Online GmbH⁵ zur Verfügung gestellt. In der Hetzner Cloud werden zehn verschiedene Servertypen gehostet, wobei die Kosten für den kleinsten Server (1 virtueller CPU, zwei GB RAM, 20 GB SSD, 20 TB Traffic) 3,92 € und die für den größten Server (16 virtuelle CPU, 32 GB RAM, 360 GB SSD, 20 TB Traffic) 64,74 € im Monat betragen. Von den jeweiligen Serverinstanzen können Snapshots und Backups erstellt werden. Zusätzlich ist das Kaufen von Floating-IPs möglich, damit die IT-Infrastruktur der entwickelten IDE immer unter den gleichen statischen IP-Adressen zu erreichen ist [70]. Da es sich um die Entwicklung einer vollständig hochverfügbaren IDE handelt (vgl. Kapitel 4.2), müssen für die HDFS-Namenodes und YARN-Resourcemanager jeweils drei Master-Knoten⁶ und für die Services Kerberos und LDAP jeweils zwei Instanzen in der Hetzner Cloud bestellt werden (vgl. Tabelle 4.3.1). Für alle zehn Instanzen wird eine Floating-IP bestellt, das ergibt in Bezug auf die Kosten für die Floating-IPs einen monatlichen Gesamtbetrag von 35,70 €⁷.

⁵ Die Hetzner Online GmbH hat ihren Sitz in Gunzenhausen und wurde ursprünglich von Martin Hetzner 1997 als "Hetzner Online Service" gegründet [33].

⁶ Das Hadoop-Cluster wird mit der Hadoop-Version 3.3 installiert und konfiguriert.

⁷ Für die Konfiguration der IDE mit einer gemeinsamen Adresse über zwei Keepalived-Instanzen wird eine weitere Floating-IP vorausgesetzt (vgl. Kapitel 4.2). Das Bestellen dieser zusätzlichen IP ist allerdings erst nach einer Support-Anfrage für eine Kontingenterhöhung der Floating-IPs möglich, da jeder Nutzer eine Begrenzung auf zehn Floating IPs besitzt

Server	Bezeichnung	CPU	RAM	Anzahl	Kosten (Tag)	Kosten (Monat)
Master	CPX41	8	16	3	0,97 €	29,39 €
Worker	CPX51	16	32	3	2,70 €	64,74 €
Hub	CPX31	4	8	2	0,65 €	15,59 €
Security	CPX11	2	4	2	0,19 €	4,58 €
Gesamt		30	60	10	12,69 €	322,73 €

Tabelle 4.3.1.: Übersicht der zugrundeliegenden IT-Infrastruktur in der Hetzner Cloud

Mittels einer REST-API ist es möglich, die für die Online IDE notwendige IT-Infrastruktur über Ansible (vgl. Kapitel 2.3) zu bestellen und zu steuern. Zur Einsparung von Kosten im Entwicklungsprozess und dem Testen aller an der Installation der IDE beteiligten Automatisierungen werden die bestellten virtuellen Maschinen regelmäßig über die REST-API gelöscht und wieder neu aufgebaut.

Kapitel 5.

Technische Umsetzung der IDE

Nach der Planung der zugrundeliegenden IT-Infrastruktur kann diese nun realisiert werden. Dafür wird in GitHub ein entsprechendes öffentliches Git-Repository angelegt¹. Im Folgenden wird die technische Umsetzung der IDE näher erläutert sowie Teile der IT-Infrastruktur überprüft und auf Ausfallsicherheit getestet. Das Kapitel endet mit einer Beschreibung der Berechtigungsstruktur.²

5.1. Automatisiertes Verfahren - Installation und Konfiguration

Zur Nachvollziehbarkeit des automatisierten Prozesses für den Aufbau der IDE wird das dafür angelegte öffentlich zugängliche Git-Repository auf einen internetfähigen Laptop geklont (vgl. Auflistung 5.1.1).

```
georg@georg - InfinityBook - Pro - 15 - v5:~/git$ pwd  
/home/georg/git  
georg@georg - InfinityBook - Pro - 15 - v5:~/git$ git clone https://github.com/GeorgSchulz  
/CommunityLab.git
```

Auflistung 5.1.1: Git-Repository - Klonen

Das Git-Repository gliedert sich auf der ersten Ebene thematisch in die Bereiche Ansible (Ordner: *collections*, *group_vars*, Dateien: *ansible.cfg*, *inventory*)³, Anwendungsbeispiele (Ordner: *examples*) und Dokumentation (Ordner: *documentation*). Die verschiedenen Phasen der IDE-Installation werden durch das zentrale Ansible-Playbook *setup.yml* gesteuert (vgl. Auflistung 5.1.2).

¹ siehe Git-Repository CommunityLab

² Dieses Kapitel hat nicht den Anspruch einer detaillierten Beschreibung des gesamten Installationsprozesses der IDE. Diese liegt in Form der Benennung der Einzelschritte in den verwendeten Ansible-Playbooks vor.

³ siehe dazu Kapitel 2.3.5, Kapitel 2.3.2, Kapitel 2.3.3 und Kapitel 2.3.1

```
georg@georg - InfinityBook - Pro - 15 - v5:~/git/CommunityLab$ tree -L 1
.
└── ansible.cfg
└── collections
└── documentation
└── group_vars
└── inventory
└── LICENSE
└── README.md
└── setup.yml

4 directories, 5 files
```

Auflistung 5.1.2: Git-Repository - erste Ebene

Die Phasen des Installations-Prozesses befinden sich im Ordner `collections` in Form von Namespaces⁴ (vgl. Auflistung 5.1.3).

```
georg@georg - InfinityBook - Pro - 15 - v5:~/git/CommunityLab$ tree -L 2
    collections/
collections/
└── ansible_collections
    ├── authentication
    ├── authorization
    ├── bigdata
    ├── hadoop
    ├── hetzner
    ├── ide
    ├── jupyter
    ├── loadbalancing
    ├── rdbs
    └── tls
└── requirements.yml

11 directories, 1 file
```

Auflistung 5.1.3: Git-Repository - Namespaces der Ansible-Collections

Die jeweiligen Namespaces bestehen aus den für die Installationsphasen selbst entwickelten Ansible-Collections. Hier beispielsweise für alle relevanten Ansible-Collections der Installationsphase Hadoop gegliedert nach serverspezifischen (*yarn*, *hdfs*), clientspezifischen (*client*) sowie Ansible-Collections für Server und Clients (vgl. Auflistung 5.1.4).

⁴ siehe dazu Kapitel 2.3.5

```

georg@georg - InfinityBook - Pro - 15 - v5 : ~/git/CommunityLab$ tree -L 3
    collections/ansible_collections/hadoop
collections/ansible_collections/hadoop
|___ client
    |___ roles
        |___ setup
|___ common
    |___ roles
        |___ setup
|___ hdfs
    |___ roles
        |___ check
        |___ common
        |___ datanode
        |___ journalnode
        |___ namenode
|___ yarn
    |___ roles
        |___ check
        |___ common
        |___ nodemanager
        |___ resourcemanager

19 directories, 0 files

```

Auflistung 5.1.4: Git-Repository - Ansible-Rollen des Namespaces hadoop

Alle Namespaces werden in dem Ansible-Playbook *setup.yml* aufgerufen und die IDE über folgenden Befehl installiert (vgl. Auflistung 5.1.5).

```
georg@georg - InfinityBook - Pro - 15 - v5 : ~/git/CommunityLab$ ansible-playbook setup.yml
```

Auflistung 5.1.5: Installation der IDE

Individuelle Rahmenbedingungen der jeweiligen IT-Umgebung werden in *group_vars* in der Datei *all.yml* mittels Variablen gesetzt. Neben der Installation aller Systeme in der Hetzner Cloud hat der Nutzer die Option, seine bereits vorhandenen Systeme bei einem anderen Cloud-Dienst oder On-Premise zu nutzen. Dafür notwendig ist das Anlegen einer Inventory-Datei auf der ersten Ebene des Repository. Die Inventory-Datei sowie die selbst geschriebenen Collections werden über die *ansible.cfg*⁵ erkannt, ohne dass der Nutzer diese bei Ausführung der *setup.yml* übergeben muss. Beispiele für ein *custom* In-

⁵ siehe Git-Repository *ansible.cfg*

ventory sowie eine all.yml befinden sich im *examples*-Ordner⁶ des Repository. Zwingende Voraussetzung für beide Vorgehensweisen ist ein SSH-Keypair des ausführenden Nutzers. Die Ausführung der *setup.yml*⁷ kann mit verschiedenen Ansible-Tags (vgl. Kapitel 2.3.4) auf die entsprechenden Installations-Phasen begrenzt werden (vgl. Tabelle 5.1.1).

Installationsphase	Bedeutung	Ansible-Gruppen
preflight	Überprüfung aller notwendigen Variablen	ansible
hetzner	Bestellung aller notwendigen Server über die Hetzner-API	ansible
dns	Konfiguration der bestellten Server für Namensauflösung	all
environment	Installation Java, Anlegen technischer Nutzer und Gruppen	all
tls	Bestellung und Verteilung aller Zertifikate mittels Let's Encrypt	all
ldap	Einrichtung LDAP-Server und LDAP-Clients	securities,all
kerberos	Einrichtung Kerberos-Server und Kerberos-Clients	securities,all
zookeeper	Installation Apache Zookeeper Cluster	masters
hadoop	Installation HDFS und YARN Cluster	masters,workers
spark	Installation aller Spark-Libraries	workers
jupyter	Installation PostgreSQL-Cluster, JupyterHubs und JupyterLab	workers,hubs

Tabelle 5.1.1.: Installationsphasen der Custom IDE

In der ersten Phase des Installationsprozesses wird automatisch geprüft, ob der Nutzer die all.yml und alle erforderlichen Variablen richtig definiert hat bzw. auch die Erreichbarkeit aller Systeme sowie der genutzten Domäne getestet. Ist dies nicht der Fall, bricht der Installationsprozess ab und der Nutzer wird auf eine entsprechende Fehlermeldung hingewiesen (siehe Anhang B.1). Erfolgt die Installation der IDE in der Hetzner Cloud ist die Definition einer eigenen Inventory-Datei nicht notwendig. Ursache dafür ist, dass alle Systeme nach der Vorbereitung in der Hetzner Cloud automatisch über den Namespace hetzner mit der in der *all.yml* definierten Variablen-Liste *cloud_servers* in einem dynamischen Inventory erstellt werden. Dafür muss ebenfalls der genutzte öffentliche SSH-Key in der *all.yml* mit der Variable *ansible_public_key* übergeben werden. Bei sensiblen Variablen wie beispielsweise dem Hetzner API-Token (vgl. Kapitel 5.2) wird empfohlen, die Werte vorher mit dem Ansible Vault zu verschlüsseln.

⁶ siehe Git-Repository examples-Ordner

⁷ siehe Git-Repository setup.yml

5.2. Vorbereitungen in der Hetzner Cloud

Für die IT-Infrastruktur der IDE müssen mehrere virtuelle Maschinen in der Hetzner Cloud bestellt werden (vgl. Kapitel 4.3). Nach erfolgreicher Registrierung bei Hetzner wird dafür ein neues Projekt in der Hetzner Cloud angelegt (vgl. Abbildung 5.2.1).

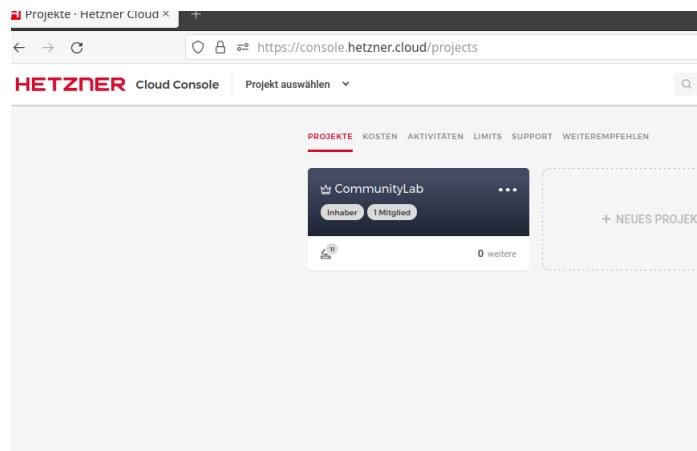


Abbildung 5.2.1.: Notwendiges Projekt in der Hetzner Cloud anlegen

In dem jeweiligen Projekt wird ein API-Token generiert, über den mittels Ansible alle virtuellen Maschinen bestellt und konfiguriert werden. Der API-Token muss deswegen Lese- und Schreibrechte im entsprechenden Projekt besitzen (vgl. Abbildung 5.2.2).

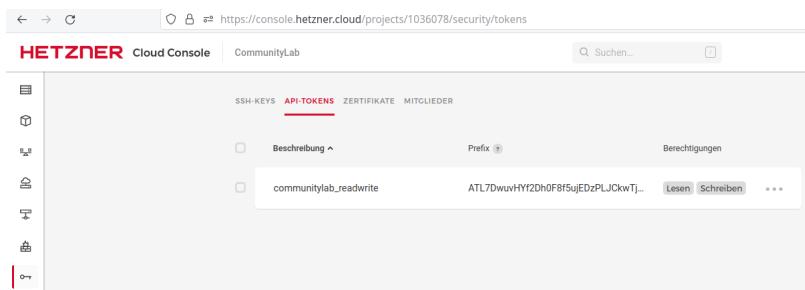


Abbildung 5.2.2.: Notwendiges API Token in der Hetzner Cloud anlegen

Voraussetzung für die in der IDE-Installation vorgesehene verschlüsselte Kommunikation

der einzelnen Komponenten ist die Austellung von Sicherheitszertifikaten von einer Certification Authority und eine dafür verfügbare Domäne. Dabei kann die Installation der IDE auch mit eigenen Sicherheitszertifikaten über Variablen in einem *custom* Inventory erfolgen. Für die Ausstellung der Sicherheitszertifikate wird die Certification Authority Let's Encrypt verwendet. Damit die Zertifikate über die Automatisierung bestellt werden können, müssen die Floating-IPs (vgl. Kapitel 4.3) in der jeweiligen DNS-Verwaltung hinterlegt werden (vgl. Abbildung 5.2.3).

The screenshot shows the Hetzner konsoleH DNS-Verwaltung interface. On the left, there is a sidebar with various management links like MariaDB/MySQL, PostgreSQL, Konfiguration, Subdomains, DNS-Verwaltung, Extras, and Statistics. The main area is titled 'DNS: click-your-it.de'. It displays two tables of records.

A-Records:

Hostname	TTL	Recordtyp	Ziel
@	188.40.28.39	A	
hub1	95.217.244.23	A	
hub2	95.217.244.40	A	
jupyterhub	95.217.246.13	A	
master1	95.217.244.83	A	
master2	95.217.245.232	A	
master3	95.217.244.36	A	
security1	95.217.245.177	A	
security2	95.217.244.240	A	
worker1	95.217.245.122	A	
worker2	95.217.245.221	A	
worker3	95.217.245.188	A	
www	188.40.28.39	A	

CNAME-Records:

Hostname	TTL	Recordtyp	Ziel
autoconfig		CNAME	mail.your-server.de.

Abbildung 5.2.3.: Hinterlegen der Floating-IPs in der DNS-Verwaltung

Erfolgt die Verbindung zu beiden JupyterHub-Knoten über eine gemeinsame Adresse (vgl. Kapitel 4.2), muss für die Zertifikatsbestellung der Hub-Instanzen ein für die DNS-Challenge benötigter Text-Record in der DNS-Verwaltung abgespeichert werden (vgl. Abbildung 5.2.4).

TXT-Records			
Hostname	TTL	Recordtyp	Ziel
@	14400	TXT	["v=spf1 +a +mx ?all"]
_acme-challenge.jupyterhub	14400	TXT	[1-NPINRm5SgvNvngqQm2XjdJxCg1z9mfK8Wb0WErs]
acme-challenge.jupyterhub	14400	TXT	[slJMZqdsPNvU_WAZHrSIKbjKo9e3NX2b6Qwt7oUg]

DNS-Server

Abbildung 5.2.4.: TXT-Records für DNS-Challenge mit Let's Encrypt

5.3. Überprüfung der einzelnen Komponenten

Nach erfolgreicher Installation der IDE mithilfe der *setup.yml* ist es hilfreich, nochmals alle Komponenten auf ihre Funktionsfähigkeit zu überprüfen. Dafür ist der Ansible-Tag (vgl. Kapitel 2.3.4) *check* vorgesehen (vgl. Auflistung 5.3.1).

```
georg@georg - InfinityBook:~/git/jupyter_on_hadoop$ ansible-playbook setup.yml --tags check
```

Auflistung 5.3.1: Überprüfung aller Komponenten der IDE auf ihre Funktionsfähigkeit

Der Tag kann auch auf ausgewählte Gruppen des verwendeten Inventory limitiert werden. Dadurch werden nur die Komponenten der IDE überprüft, die den jeweiligen Gruppen angehören (vgl. Auflistung 5.3.2).

```
georg@georg - InfinityBook:~/git/jupyter_on_hadoop$ ansible-playbook setup.yml --tags check --limit ansible
```

Auflistung 5.3.2: Überprüfung aller notwendigen Variablen

In diesem Sinne werden bei der Limitierung auf die Gruppe *ansible*⁸ alle für die Installation vorgesehenen Variablen überprüft (vgl. Kapitel 5.1). Der *check* für die Gruppen *securities* wiederum überprüft die Funktionsfähigkeit der Services LDAP und Kerberos (vgl. Tabelle 5.3.1).

⁸ Diese Gruppe entspricht dem localhost, also in der Regel dem Laptop von dem aus die IDE in der Hetzner Cloud installiert wird

Inventory-Gruppe	ausgeführter Check
ansible	Überprüfung aller gesetzten Variablen in group_vars/all.yml
securities	Überprüfung LDAP, Kerberos
masters	Überprüfung HDFS, YARN, Zookeeper
workers	PostgreSQL, HAProxy

Tabelle 5.3.1.: Ansible Tag check für verschiedene Inventory-Gruppen

Bezüglich LDAP prüft eine Abfrage gegen den LDAP-Server, ob alle Nutzer in der LDAP-Gruppe vorhanden sind, die in der Berechtigungsstruktur vorgesehen ist (vgl. Kapitel 5.5). Der Kerberos-Server wird hinsichtlich Hochverfügbarkeit untersucht. Dahingehend wird kurzfristig ein sogenannter Kerberos Principal auf dem primären Kerberos-Server angelegt und dieser dann mithilfe der Kerberos Propagation auf den sekundären Kerberos-Server repliziert. Bei erfolgreicher Replikation des Datenbestands des primären Kerberos-Servers auf den sekundären Kerberos-Server wird der angelegte Kerberos Principal auf beiden Kerberos-Servern wieder gelöscht.

Die *masters*-Gruppe beinhaltet die Services HDFS, YARN und Apache Zookeeper. Im HDFS (vgl. Kapitel 2.2.1) werden mehrere Schritte ausgeführt, damit die Funktionsfähigkeit des Services gegeben ist. So wird erstens sichergestellt, dass einer der drei HDFS-Namenodes aktiv ist.⁹ Zweitens wird getestet, ob die HDFS-Namenodes alle vorgesehnen HDFS-Datanodes aktiv verwalten. Zu guter Letzt wird geschaut, ob der technische Nutzer des HDFS in das verteilte Dateisystem schreiben kann. Gleichermassen wird mit dem Hadoop-Service YARN verfahren. Es wird einerseits überprüft, ob einer der drei YARN-Resourcemanager aktiv ist¹⁰ und anderseits, ob dieser auch alle seine vorgesehnen YARN-Nodemanager aktiv verwaltet. Der Service Apache Zookeeper ist funktionsfähig, wenn der Root-ZNode abgefragt werden kann.

⁹ In einem ausfallsicheren HDFS ist immer nur ein HDFS-Namenode aktiv, wobei die anderen HDFS-Namenodes passiv sind.

¹⁰ Auch bei einem hochverfügaren YARN-Cluster ist immer nur einer der Resourcemanager aktiv.

Die *workers*-Gruppe beinhaltet das Backend der JupyterHub-Knoten in Form eines PostgreSQL-Clusters (vgl. Kapitel 4.2). Mit einer *SELECT*-Abfrage wird sichergestellt, dass das JupyterHub-Schema innerhalb der Datenbank über den HAProxy erstellt werden kann. In diesem Fall wird gewährleistet, dass mindestens einer der beiden Load Balancer funktionsfähig ist. Anschließend wird noch der für den JupyterHub-Service notwendige Datenbanknutzer sowie das Schema in der PostgreSQL angelegt und somit verifiziert, dass die schreibende¹¹ Datenbank-Instanz über die Load Balancer angesprochen wird.

¹¹ In einem PostgreSQL-Cluster mit Patroni gibt es immer nur eine aktive schreibende Datenbank

5.4. Test der Ausfallsicherheit

Nach der Überprüfung der IDE auf ihre inhaltliche Funktionsfähigkeit sollte sie ebenfalls nochmals hinsichtlich Ausfallsicherheit der einzelnen Komponenten kontrolliert werden (vgl. Tabelle 5.4.1). Dafür wird der jeweilige Service immer auf einer der relevanten virtuellen Maschinen gestoppt und anschließend ein YARN-Container gestartet, indem ein Nutzer sich neu über das Frontend anmeldet. Als Ergebnis sind alle Services der IDE mit Ausnahme des JupyterHub-Services vollständig ausfallsicher. Grund dafür ist, dass die Sessions der Nutzer vom Proxy JupyterHub nach dem Wechsel über den Keepalived-Service nichtpersistiert werden. Verzichtet man auf den Keepalived-Service und nutzt beide JupyterHub-Instanzen, werden die Nutzer ohne Probleme zu ihren jeweiligen YARN-Containern durchgeroutet.

Ausfallsicherheit aller IDE-Komponenten	
HAProxy	ja
JupyterHub	nein
LDAP	ja
Kerberos	ja
Apache Zookeeper	ja
HDFS	ja
YARN	ja
PostgreSQL	ja

Tabelle 5.4.1.: Ausfallsicherheit

5.5. Berechtigungsstruktur der IDE

Nachdem die IDE auf Funktionsfähigkeit und Ausfallsicherheit überprüft worden ist, wird die vorgesehene Berechtigungsstruktur kurz vorgestellt. In der Datenbank des LDAP-Servers wird über das Deployment die LDAP-Gruppe *ide_users* mit allen definierten Nutzern und deren Passwort angelegt.¹² Nach der Anmeldung über den JupyterHub-Service überprüft dieser, ob die eingegebene Benutzerkennung und das Passwort entsprechend in der Datenbank des LDAP-Servers hinterlegt sind und ob dieser Nutzer auch der Gruppe *ide_users* angehört (vgl. Auflistung 5.5.1).

```
c.JupyterHub.authenticator_class = 'ldapauthenticator.LDAPAuthenticator'  
c.LDAPAuthenticator.use_ssl = True  
c.LDAPAuthenticator.server_port = 636  
c.LDAPAuthenticator.server_address = "0.0.0.0"  
c.LDAPAuthenticator.lookup_dn_search_user = 'CN=admin,{{ ldap_organization_upper }}'  
c.LDAPAuthenticator.lookup_dn_search_password = '{{ ldap_password }}',  
c.LDAPAuthenticator.bind_dn_template = 'UID={username},OU=people,{{  
    ldap_organization_upper }}',  
c.LDAPAuthenticator.lookup_dn = True  
c.LDAPAuthenticator.search_filter = '(&(objectClass=posixAccount)(uid={username}))'  
,  
c.LDAPAuthenticator.username_pattern = '[a-zA-Z0-9_.][a-zA-Z0-9_.-]{0,252}[a-zA-Z0-  
-9_.-$-]?',  
c.LDAPAuthenticator.lookup_dn_user_dn_attribute = 'uid',  
c.LDAPAuthenticator.user_attribute = 'uid',  
c.LDAPAuthenticator.escape_userdn = False  
c.LDAPAuthenticator.user_search_base = 'OU=people,{{ ldap_organization_upper }}',
```

Auflistung 5.5.1: LDAP-Konfiguration des Service JupyterHub

Bei erfolgreicher Anmeldung im JupyterHub leitet dieser den Nutzer mit Kerberos-Authentifizierung an das Hadoop-Ökosystem (vgl. Kapitel 2.2.1) weiter. Dort wird wie auch in JupyterHub über einen weiteren LDAP-Filter überprüft¹³, ob der Nutzer der Gruppe *ide_users* angehört (siehe Anhang B.2). Der Nutzer ist nun berechtigt über YARN einen Container im Hadoop-Ökosystem zu starten. Da alle Daten des Containers über den *HDFS Content Manager* direkt im HDFS abgelegt werden, muss er außerdem Schreibrechte in diesem verteilten Dateisystem besitzen. Dafür werden zwei Access Control List-Einträge im Root-Verzeichnis des HDFS und den Verzeichnissen *user* und *share*

¹² Die Nutzung eines bereits vorhandenen LDAP-Server sowie einer anderen LDAP-Gruppe kann über ein *custom_inventory* mit Variablen implementiert werden.

¹³ Voraussetzung für die Abfrage des LDAP-Servers aus dem Hadoop-Ökosystem ist, dass auch das zugrundeliegende Betriebssystem mit diesem kommunizieren kann.

gesetzt. Als Konsequenz können alle Nutzer kollaborativ miteinander arbeiten und sich über den Ordner *share* Jupyter-Notebooks sowie größer mit der IDE zu analysierende Dateien austauschen. Gleichermassen ist es anderen Nutzern nicht möglich, nutzerspezifische Daten anderer Nutzer zu löschen oder zu ändern.

Kapitel 6.

Vergleich mit Enterprise-Produkten

Nachdem die selbstentwickelte IDE vorgestellt worden ist, folgt nun der Vergleich mit den Enterprise-Produkten von AWS und Microsoft Azure. Interessant für das Vorhaben ist die Analyse großer Dateien mittels Apache Spark über ein Jupyter-Notebook (vgl. Kapitel 1). EMR und HDInsight werden vor allem genutzt, um große Dateien zu analysieren, wie es mit einem lokalen Notebook aufgrund der begrenzten Ressourcen nicht möglich wäre. Deswegen werden im folgenden Kapitel die dafür benötigten Schritte in den Enterprise-Produkten beschrieben. Ziel ist, ein Jupyter-Notebook mit einem PySpark-Kernel (vgl. Kapitel 2.1.2) zu erstellen und eine sehr große Datei (`train_data.csv`, 16 GB) [9] einzulesen bzw. einen Teil dieses Datensatzes auszugeben. Die Gegenüberstellung orientiert sich an der Nutzerfreundlichkeit der Enterprise-Produkte, wobei kein notwendiges Vorwissen der Data Scientisten in Bezug auf IT-Infrastruktur vorausgesetzt wird. Damit gemeint sind beispielsweise spezielle Konfigurationen in den Enterprise-Produkten wie die Authentifizierung und Autorisierung der Benutzer (vgl. Kapitel 4.2). Des Weiteren liegt der Fokus auf der Zusammenarbeit mit mehreren Nutzern über Jupyter.

6.1. Amazon EMR

Nach der Registrierung bei AWS erfolgt die Anmeldung auf der AWS-Konsole mit dem Stammbenutzer über die jeweilige E-Mail-Adresse (siehe Anhang C.1). Auf der Startseite der AWS-Konsole wird nach EMR gesucht (siehe Anhang C.2). Im EMR-Bereich der AWS-Konsole sind nun die verschiedenen Produkte wie beispielsweise EMR Cluster und EMR Notebook (vgl. Kapitel 3.2) auswählbar und konfigurierbar (siehe Anhang C.3). Da das Hauptaugenmerk auf JupyterHub liegt, wird über den Reiter *Cluster* ein EMR Cluster erstellt. In der Hauptkonfiguration des Clusters wird ein Clustername vergeben, die EMR Software Spark ausgewählt und die Anzahl der Maschinen auf eine reduziert¹

¹ Zur Veranschaulichung der Funktionsfähigkeit wird lediglich eine Maschine benötigt.

(siehe Anhang C.4). Ansonsten werden die gegebenen Standard-Einstellungen verwendet. In der erweiterten Konfiguration werden zusätzlich zu den bereits ausgewählten Einstellungen noch JupyterHub und das JupyterHub-Gateway ausgewählt (siehe Anhang C.5). In den restlichen Einstellungen der erweiterten Konfiguration wird lediglich noch die Maschinenanzahl der Core-Knoten auf Null reduziert (siehe Anhang C.6), der Clustername noch einmal vergeben (siehe Anhang C.7) sowie ein SSH-Schlüssel für die direkte Verbindung zu den Instanzen über SSH hinterlegt (siehe Anhang C.8). Das EMR Cluster ist nun konfiguriert und kann gestartet werden.

Nach Bereitstellung des EMR-Clusters ist im Anwendungsverlauf die Adresse des JupyterHub verfügbar (siehe Anhang C.9). Die Verbindung zu diesem ist aus dem öffentlichen Netz allerdings noch nicht möglich, da dafür vorerst eine Portfreischaltung erfolgen muss. Der Port für JupyterHub wird in EMR über *Security Groups* freigeschaltet (siehe Anhang C.10). Über die Schaltfläche *Regeln für eingehenden Datenverkehr bearbeiten* innerhalb der EMR Master-Instanz findet eine Weiterleitung auf die entsprechende Übersicht statt (siehe Anhang C.11). Dort wird für den JupyterHub-Port jeglicher Zugriff aus dem öffentlichen Netz gestattet (siehe Anhang C.12). Anschließend kann man sich gegen die GUI von JupyterHub verbinden. Die Verbindung aus dem öffentlichen Netz erfolgt allerdings über ein selbst signiertes Zertifikat, dem nicht von einer öffentlichen Certification Authority vertraut wird. Name und Passwort für die Erstanmeldung können der Dokumentation von AWS entnommen werden² (vgl. Abbildung 6.1.1).

² siehe AWS-Dokumentation User Access



Sign in

Username:
joyyan
Password:
.....
Sign in

Abbildung 6.1.1.: Anmeldemaske - JupyterHub Amazon EMR

Sobald die Anmeldung im JupyterHub abgeschlossen ist, wird ein PySpark-Kernel ausgewählt und ein Jupyter-Notebook erstellt (vgl. Abbildung 6.1.2).

Abbildung 6.1.2.: PySpark Kernel auswählen

Große Datensätze werden in EMR über einen S3-Bucket hochgeladen (vgl. Kapitel 3.1). Über die AWS-Konsole findet eine Weiterleitung in den S3-Bereich statt (siehe Anhang C.13). Dort wird ein S3-Bucket erstellt (siehe Anhang C.14). Für die Erreichung des S3-Buckets aus dem öffentlichen Netz wird zusätzlich die Blockade aller öffentlichen Zugriffe in AWS deaktiviert (siehe Anhang C.15). Nach dem Starten der Spark-Applikation über Apache Livy wird der S3-Bucket in das Jupyter-Notebook eingebunden, die große CSV-Datei eingelesen und ein Teil des Datensatzes ausgegeben (vgl. Abbildung 6.1.3).

```

In [1]: from pyspark.sql import SparkSession
from pyspark.sql.functions import col,sum,when,avg,mean,min
df = spark.read.options(delimiter=",").csv("s3://jupyterlargefiles/train_data.csv")
Starting Spark application
ID          YARN Application ID   Kind   State   Spark UI   Driver log   User   Current session?
0   application_1667413691696_0001   pyspark   idle   Link   None
SparkSession available as 'spark'.
In [2]: df.head(1)
Row[c#>customer_ID', 'c1'=5.2', 'c2'=P.2', 'c3=0.39', 'c4=B.1', 'c5=B.2', 'c6=R.1', 'c7='5.3', 'c8='0.41', 'c9='1.2', 'c10='0.12', 'c11='0.12', 'c12='0.12', 'c13='0.12', 'c14='0.12', 'c15='0.12', 'c16='0.12', 'c17='0.12', 'c18='0.12', 'c19='0.12', 'c20='0.12', 'c21='0.12', 'c22='B.7', 'c23='B.8', 'c24=D.59', 'c25=D.51', 'c26=B.9', 'c27=R.3', 'c28='0.52', 'c29=P.3', 'c30=B.10', 'c31=D.53', 'c32='S.5', 'c33=B.11', 'c34='S.6', 'c35=D.54', 'c36=R.4', 'c37='0.55', 'c38='0.55', 'c39='0.55', 'c40='0.55', 'c41='0.55', 'c42='0.55', 'c43='0.55', 'c44='0.55', 'c45='0.55', 'c46='B.14', 'c47='D.59', 'c48='0.69', 'c49='0.61', 'c50=B.15', 'c51='S.11', 'c52='D.62', 'c53='D.63', 'c54='D.64', 'c55='D.65', 'c56=B.16', 'c57='B.17', 'c58='B.18', 'c59=B.19', 'c60=D.66', 'c61=B.20', 'c62=D.68', 'c63='S.12', 'c64='D.69', 'c65='D.69', 'c66='D.69', 'c67='D.69', 'c68='D.69', 'c69='D.69', 'c70='D.70', 'c71='D.70', 'c72='D.70', 'c73='D.73', 'c74='D.73', 'c75='P.4', 'c76='D.74', 'c77='D.75', 'c78='D.76', 'c79='B.24', 'c80='H.7', 'c81='B.77', 'c82='B.25', 'c83='B.26', 'c84='B.27', 'c85='B.29', 'c86=R.8', 'c87=R.9', 'c88='S.16', 'c89=D.80', 'c90=R.10', 'c91='B.10', 'c92='D.81', 'c93='D.81', 'c94='D.81', 'c95='D.81', 'c96='D.81', 'c97='D.81', 'c98='D.81', 'c99='D.81', 'c100='D.14', 'c101='R.15', 'c102='D.84', 'c103='R.16', 'c104='D.29', 'c105='B.30', 'c106='S.18', 'c107='D.86', 'c108='D.87', 'c109='R.17', 'c110='D.21', 'c111='B.18', 'c112='B.31', 'c113='B.32', 'c114='B.33', 'c115='B.34', 'c116='B.35', 'c117='B.36', 'c118='B.37', 'c119='B.38', 'c120='D.39', 'c121='B.39', 'c122='B.27', 'c123='B.27', 'c124='B.39', 'c125='D.93', 'c126='D.94', 'c127='R.24', 'c128='R.25', 'c129='D.96', 'c130='S.22', 'c131='S.23', 'c132='S.24', 'c133='D.97', 'c134='B.38', 'c135='B.39', 'c136='B.40', 'c137='B.41', 'c138='B.42', 'c139='B.43', 'c140='B.44', 'c141='B.45', 'c142='B.37', 'c143='B.28', 'c144='B.27', 'c145='B.38', 'c146='D.100', 'c147='D.109', 'c148='D.110', 'c149='D.111', 'c150='B.39', 'c151='D.112', 'c152='B.40', 'c153='B.27', 'c154='D.113', 'c155='D.114', 'c156='D.115', 'c157='D.116', 'c158='D.124', 'c159='D.125', 'c160='D.126', 'c161='D.127', 'c162='B.128', 'c163='B.129', 'c174='B.41', 'c175='B.42', 'c176='D.130', 'c177='D.131', 'c178='D.132', 'c179='D.133', 'c177=R.28', 'c178=R.134', 'c179=R.135', 'c180=D.128', 'c181='B.131', 'c182='D.138', 'c183='D.139', 'c184='D.140', 'c185='D.141', 'c186='D.142', 'c187='D.143', 'c188='D.144', 'c189='D.145']

```

Abbildung 6.1.3.: EMR - CSV-Datei mit PySpark analysieren

Das Hinzufügen weiterer Nutzer in den erstellten JupyterHub ausschließlich über zusätzliche Konfigurations-Schritte möglich (hier die anleitung verlinken). Ein Zusammenarbeiten aller Nutzer innerhalb von JupyterHub ist nur über den S3-Bucket gegeben, da jeder Nutzer isoliert voneinander in einem anderen Docker-Container arbeitet³.

6.2. Azure HDInsight

Die Anmeldung im Azure-Portal erfolgt über den Microsoft-Account. Dort wird nach HDInsight gesucht (siehe Anhang D.1). Für die Erstellung des HDInsight-Clusters muss eine Ressourcengruppe, ein Clustername sowie eine Region angeben werden. Die Arbeit mit Apache Spark wird über den Clustertyp Spark ermöglicht. Abschließend wird noch ein Admin-Kennwort vergeben, das später für die Verbindung mit dem Jupyter-Notebook benötigt wird (siehe Anhang D.2). Weitere Konfigurationen sind für den Anwendungsfall nicht erforderlich.

Das Spark-Cluster kann vorerst nicht erstellt werden, da die für einen Entwickler-Account⁴ gegebenen Ressourcen in HDInsight unzureichend sind (siehe Anhang D.3). Die Standard-Einstellungen für ein Spark-Cluster setzen bereits so viele CPUs voraus, dass es ohne vorher beantragte Kontingenterhöhung nicht erstellt werden kann. Das ist in Relation zu AWS ein großer Nachteil und widerspricht dem generellen Konzept der einfachen Daten-

³ Das ist auch für das von EMR angebotenen EMR-Studio der Fall.

⁴ Für die Arbeit mit HDInsight wird ein Entwickler-Account bei Microsoft-Azure bestellt (hier referenzieren).

analyse mittels Kreditkarte ohne vorher gegebene Rahmenbedingungen. Aufgrund der fehlenden Ressourcen wird für die Region *Germany West Central* ein Kontingent von 42 CPUs beantragt⁵ (siehe Anhang D.4). Das HDInsight-Cluster kann danach mit folgender VM-Konfiguration bewertet und erstellt werden (vgl. Abbildung 6.2.1).

Knotentyp	Knotengröße	Knotenzahl	Geschätzte Kosten/Stunde
Hauptknoten-K...	E8 V3 (8 Kerne, 64 GB RAM), 0.76 USD/Stunde	2	1.52 USD
Zookeeper-Knot...	A2 v2 (2 Kerne, 4 GB RAM), 0.12 USD/Stunde	3	0.00 (KOSTENLOS)
Worker-Knoten	E20 V3 (20 Kerne, 160 GB RAM), 1.90 USD/Stu...	1	1.90 USD

Aktivieren des verwalteten Datenträgers
 Automatische Skalierung aktivieren [Weitere Informationen](#)

Geschätzte Gesamtkosten/Stunde 3.42 USD

[Bewerten + erstellen](#) [« Zurück](#) [Weiter: Tags »](#)

Abbildung 6.2.1.: VM-Konfiguration HDInsight

⁵ Der Beantragungsprozess dauert mehrere Tage, da dafür ein Support-Ticket eröffnet werden muss.

Ist die Bereitstellung des HDInsight-Clusters erfolgt, wird zur Ressource gewechselt (siehe Anhang D.5). Wie auch bei AWS wird die zu analysierende CSV-Datei an einen öffentlich zugänglichen Ort gelegt (vgl. Kapitel 6.1). Nun folgt die Auswahl des Speicherkontos über die Übersicht des Clusters (siehe Anhang D.6). Im Speicherkonto findet danach das Hochladen der Datei (siehe Anhang D.7) und das Vergeben der Dateirechte (siehe Anhang D.8) statt. Der Dateilink wird abschließend kopiert (siehe Anhang D.9), damit die Datei später aus dem Jupyter-Notebook geladen werden kann.

Die Anmeldung am Jupyter-Notebook über die Startseite des Clusters ist jetzt möglich. Dafür wird der bei der Erstellung des Clusters vergebene Nutzer sowie dessen Passwort verwendet. Im Unterschied zu dem EMR-Cluster ist die Verbindung mit Sicherheitszertifikaten verschlüsselt, denen von einer öffentlichen Certification Authority vertraut wird (vgl. Abbildung 6.2.2).

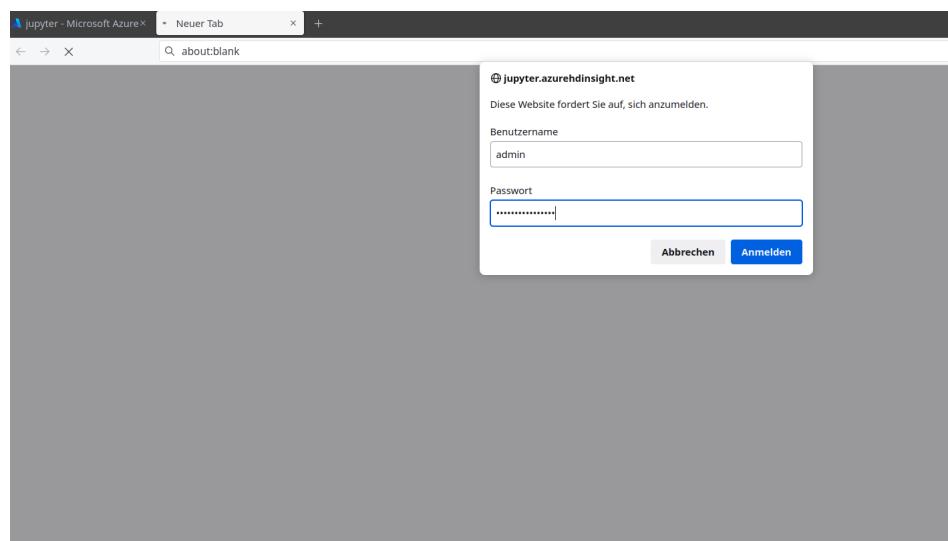


Abbildung 6.2.2.: HDInsight Jupyter-Notebook - Anmeldemaske

Die Startseite des Jupyter-Notebooks enthält die Ordner *PySpark* und *Scala*, in der mehrere Beispiele für die Datenanalyse gegeben sind (vgl. Abbildung 6.2.3).



Abbildung 6.2.3.: HDInsight Jupyter-Notebook - Startseite

Abschließend wird die in das Speicherkonto geladene CSV-Datei mittels PySpark analysiert. Auch hier erfolgt wie bei AWS die Authentifizierung gegen Spark über den technischen Livy-Nutzer⁶ (vgl. Abbildung 6.2.4).

⁶ siehe dazu Kapitel 6.1

The screenshot shows a Jupyter Notebook interface on an HDInsight cluster. The top navigation bar includes 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help'. A 'PySpark' tab is selected in the top right. Below the toolbar, there's a 'CellToolbar' with various icons. The main area contains two code cells:

```
In [1]: from pyspark import SparkFiles
from pyspark.sql import SparkSession
from pyspark.sql.functions import udf,col.count,sum,avg,mean,min
spark = SparkSession.builder.appName("spark").getOrCreate()
url = "https://jupyterhdstorage.blob.core.windows.net/jupyter-2022-11-06106-26-24-738z/train_data.csv"
spark.sparkContext.addFile(url)
df = spark.read.options(delimiter=",").csv("file:///tmp/SparkFiles/get('train_data.csv')")

```

This cell is running and has 0 tasks. The submission time was 11 minutes ago.

```
In [4]: df.head(10)

```

This cell is also running and has 0 tasks. It displays the first 10 rows of the DataFrame, which are very long strings of numerical values separated by commas. The output is truncated at the end.

Abbildung 6.2.4.: HDInsight Jupyter-Notebook - CSV-Datei mit PySpark analysieren

Weitere Nutzer können hier nicht wie bei EMR hinzugefügt werden, da es sich lediglich um ein Jupyter-Notebook und keinen JupyterHub handelt. Microsoft Azure empfiehlt in diesem Kontext die Zusammnenarbeit mit mehreren Nutzern über eine lokale Jupyter-Instanz (vgl. Kapitel 3.2).

6.3. Custom IDE

Nach der Installation der IDE können sich alle der LDAP-Gruppe zugeordneten Nutzer mit ihrer jeweiligen Benutzerkennung über das Frontend JupyterHub anmelden (vgl. Kapitel 5). Wird auf die Ausfallsicherheit der beiden Hub-Knoten über Keepalived verzichtet, erfolgt die Anmeldung über einen der beiden Hub-Knoten (vgl. Kapitel 4.2). Dabei wird nach der Anmeldung für jeden Nutzer ein YARN-Container gestartet (vgl. Abbildung 6.3.1).

```

yarnmaster1:~$ kill
[...]
default principal: yarn/master1.click-your-it.de@COMMUNITY-LAB

Valid starting      Expires            Service principal
11/06/2022 06:25:49  11/06/2022 16:25:49  krbtgt/COMMUNITY-LAB@COMMUNITY-LAB
[...]
yarnmaster1:~$ 
yarnmaster1:~$ 
yarnmaster1:~$ yarn
yarnmaster1:~$ /opt/apache-hadoop/hadoop/bin/yarn application -list
WARNING: All illegal reflective access operations will be denied in a future release
WARNING: Illegal reflective access by org.xbill.DNS.ResolverConfig (file:/opt/apache-hadoop/hadoop-3.3.3/share/hadoop/common/lib/dnsjava-2.1.7.jar) to method sun.net.dns.ResolverConfiguration.open()
WARNING: Please consider reporting this to the maintainers of org.xbill.DNS.ResolverConfig
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
Total number of applications (application-types: []) states: [SUBMITTED, ACCEPTED, RUNNING] and tags: []
application_1667715897269_0001      jupyterhub          skin           georg        default    RUNNING   UNDEFINED   10%   http://65.108.59.92:40445/
application_1667715897269_0002      jupyterhub          skin           matthias    default    RUNNING   UNDEFINED   10%   http://95.217.9.143:32857/
yarnmaster1:~$ 

```

Abbildung 6.3.1.: Auflisten aller YARN-Container der IDE

Die Nutzer werden vom technischen Nutzer des JupyterHub-Services impersonifiziert und danach an das Hadoop-Ökosystem weitergeleitet. Dort führen sie ihre Datenanalysen auf dem daraufsitzenden JupyterLab durch. Der YARN-Container wird lokal auf einem der Worker-Knoten im /tmp-Verzeichnis gespeichert⁷. Die CSV-Datei wird über JupyterLab zu Beginn in den YARN-Container geladen. Jeder Nutzer kann in seinem Container vorhandene Dateien über das Terminal in sein nutzerspezifischen HDFS-Pfad oder das dort vorhandene *share*-Verzeichnis hochladen. Wenn die Nutzer an einem Jupyter-Notebook gemeinsam arbeiten möchten, wird dies im *share*-Verzeichnis abgelegt (vgl. Abbildung 6.3.2).

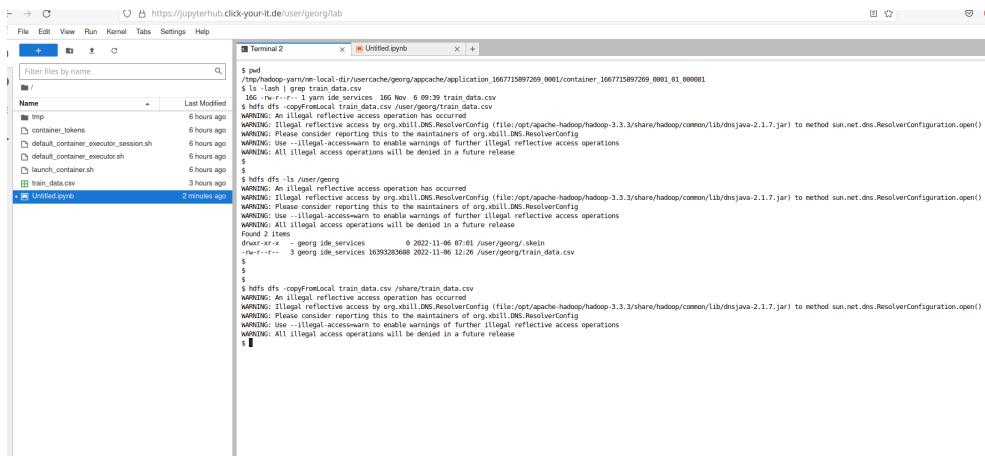


Abbildung 6.3.2.: Custom IDE - Hochladen der CSV-Datei in das HDFS

Abschließend wird die CSV-Datei mit PySpark eingelesen und ein Teil des Datensatzes ausgegeben. Eine Spark-Session startet dafür nicht im Vergleich zu EMR und HDInsight, da der Nutzer sich bereits im Hadoop-Ökosystem befindet (vgl. Abbildung 6.3.3).

⁷ Hier kann alternativ auch ein anderes Verzeichnis konfiguriert werden.

```

File Edit View Run Kernel Tabs Settings Help
Filter files by name
Name Last Modified
tmp 6 hours ago
container_tokens 6 hours ago
default_container_executor_session.sh 6 hours ago
default_container_executor.sh 6 hours ago
launch_container.sh 6 hours ago
train_data.csv 3 hours ago
Untitled.ipynb a minute ago

Terminal 2 x Untitled.ipynb + +
from pyspark import SparkFiles
from pyspark.sql import SparkSession
from pyspark.sql.functions import udf,col,count,sum,avg,mean,min
spark = SparkSession.builder.appName("spark").getOrCreate()
df = spark.read.options(delimiter=",").csv("hdfs://user/georg/train_data.csv")
Setting default log level to "WARN".
To adjust logging level, use sc.setLogLevel(newLevel).
WARNING: Illegal reflective access has occurred
WARNING: Illegal reflective access by org.apache.hadoop.shaded.org.xbill.DNS.ResolverConfig (file:/opt/miniforge/Miniforge3-Linux-x86_64/envs/jupyterlab/lib/python3.9/site-packages/pyspark/resources/hadoop-client-runtime-3.3.2.jar) to method sun.net.dns.ResolverConfiguration.open()
WARNING: Please consider reporting this to the maintainers of org.xbill.DNS.ResolverConfig
WARNING: Use -illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
2021-06-12 12:33:03 WARN package: Truncated the string representation of a plan since it was too large. This behavior can be adjusted by setting 'spark.sql.debug.maxToStringFields'
df.show()

RowId,customer_ID,clv,$.2$,c2w_P.2$,c3w_D.39$,c4w_R.1$,c5w_S.3$,c6w_R.41$,c9w_B.3$,c10w_D.42$,c11w_D.43$,c12w_D.44$,c13w_B.4$,c14w_D.45$,c15w_S.5$,c16w_R.2$,c17w_D.46$,c18w_D.47$,c19w_D.48$,c20w_D.49$,c21w_B.6$,c22w_B.7$,c23w_B.8$,c24w_D.50$,c25w_D.51$,c26w_B.9$,c27w_D.52$,c28w_D.53$,c29w_D.54$,c30w_D.55$,c31w_D.56$,c32w_B.13$,c33w_D.57$,c34w_D.58$,c35w_D.59$,c36w_D.60$,c37w_D.61$,c38w_B.15$,c39w_S.11$,c40w_D.62$,c53w_D.63$,c54w_D.64$,c55w_D.65$,c56w_B.16$,c57w_B.17$,c58w_B.18$,c59w_B.19$,c60w_D.66$,c61w_D.20$,c62w_D.68$,c63w_D.12$,c64w_R.6$,c65w_S.13$,c66w_D.14$,c67w_D.15$,c68w_R.7$,c69w_D.77$,c70w_B.25$,c71w_D.79$,c72w_R.8$,c73w_R.9$,c74w_S.16$,c75w_D.89$,c76w_R.10$,c77w_R.11$,c78w_B.27$,c79w_D.81$,c80w_D.82$,c81w_D.83$,c82w_D.84$,c83w_D.85$,c84w_D.86$,c85w_D.87$,c86w_D.88$,c87w_D.89$,c88w_D.90$,c89w_D.91$,c90w_D.92$,c91w_D.93$,c92w_D.94$,c93w_D.95$,c94w_D.96$,c95w_D.97$,c96w_D.98$,c97w_D.99$,c98w_D.100$,c99w_D.101$,c100w_D.102$,c101w_D.103$,c102w_D.104$,c103w_D.105$,c104w_D.106$,c105w_D.107$,c106w_D.108$,c107w_D.109$,c108w_D.110$,c109w_D.111$,c110w_D.112$,c111w_D.113$,c112w_D.114$,c113w_D.115$,c114w_D.116$,c115w_D.117$,c116w_D.118$,c117w_D.119$,c118w_D.120$,c119w_D.121$,c120w_D.122$,c121w_D.123$,c122w_D.124$,c123w_D.125$,c124w_D.126$,c125w_D.127$,c126w_D.128$,c127w_D.129$,c128w_D.130$,c129w_D.131$,c130w_D.132$,c131w_D.133$,c132w_D.134$,c133w_D.135$,c134w_D.136$,c135w_D.137$,c136w_D.138$,c137w_D.139$,c138w_D.140$,c139w_D.141$,c140w_D.142$,c141w_D.143$,c142w_D.144$,c143w_D.145$)

```

Abbildung 6.3.3.: Custom IDE - CSV-Datei in /user mit PySpark analysieren

Die CSV-Datei kann auch von jedem Nutzer direkt eingelesen werden, wenn sie sich in einem Verzeichnis eines anderen Nutzers befindet. Sie kann dort allerdings von anderen Nutzern nicht editiert oder gelöscht werden, da die Berechtigungsstruktur der IDE (vgl. Kapitel 5.5) das nicht vorsieht (vgl. Abbildung 6.3.4).

```

File Edit View Run Kernel Tabs Settings Help
Filter files by name
Name Last Modified
tmp 2 minutes ago
container_tokens 2 minutes ago
default_container_executor_session.sh 2 minutes ago
default_container_executor.sh 2 minutes ago
launch_container.sh 2 minutes ago
train_data.csv a minute ago

Terminal 1 x
$ hdfs dfs -copyToLocal /share/train_data.csv
WARNING: Please consider reporting this to the maintainers of org.xbill.DNS.ResolverConfig
WARNING: Use -illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
$ ls -lsh | grep train_data.csv
166 -rw-r--r-- 1 yarn ide_services 106 Nov 6 12:38 train_data.csv
$ hdfs dfs -rm /share/train_data.csv
WARNING: Please consider reporting this to the maintainers of org.xbill.DNS.ResolverConfig
WARNING: Please consider reporting this to the maintainers of org.xbill.DNS.ResolverConfig
WARNING: Use -illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
Deleted /share/train_data.csv
$ 
$ 
$ hdfs dfs -rm -u user/georg/train_data.csv
WARNING: An illegal reflective access operation has occurred
WARNING: Please consider reporting this to the maintainers of org.xbill.DNS.ResolverConfig
WARNING: Please consider reporting this to the maintainers of org.xbill.DNS.ResolverConfig
WARNING: Use -illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
rm: permission denied: user/matthias, access=rifff. Under=/user/georg: georg:ide_services:dnarw-rx-r
$ 

```

Abbildung 6.3.4.: Custom IDE - Löschen von Dateien anderer Nutzer

6.4. Gegenüberstellung der jeweiligen Funktionalitäten

Nach der Vorstellung der Enterprise-Produkte folgt nun zusammenfassend eine Übersicht der gegebenen Funktionalitäten im Vergleich zur selbst entwickelten IDE (vgl. Tabelle 6.4.1). Die Zusammenfassung orientiert sich am Komfort bei der Datenanalyse mit größeren Dateien sowie der Zusammenarbeit mit mehreren Nutzern innerhalb von Jupyter (vgl. Kapitel 6).

Funktionalitäten	HDInsight	EMR	Custom IDE
Nutzung ohne Vorbereitung	nein	ja	nein
kollaboratives Arbeiten über geteilten Speicher	nein	ja	ja
Hochladen großer Dateien in Jupyter	nein	nein	ja
vollständige Integration in das Hadoop-Ökosystem	nein	nein	ja
Kerberos-Authentifizierung ohne Knox-Gateway	nein	nein	ja

Tabelle 6.4.1.: Gegenüberstellung der Funktionalitäten von Enterprise-Produkten und Custom IDE

Einzig EMR kann ohne Vorbereitung direkt für die Analyse der CSV-Datei verwendet werden. Für das Erstellen des HDInsight-Clusters ist eine Kontingenterhöhung der CPUs notwendig sowie für die selbst entwickelte IDE die Installation mithilfe von Ansible. Abgesehen von HDInsight kann in allen Produkten kollaborativ an einer Datei über einen geteilten Speicher gearbeitet werden. Dafür wird in EMR der S3-Bucket und in der IDE das HDFS verwendet. Bei HDInsight ist die Zusammenarbeit innerhalb des Clusters nicht vorgesehen, da Microsoft hier das Verbinden von einer lokalen Jupyter-Instanz gegen das Cluster vorsieht (vgl. Kapitel 6.2). Das Hochladen sehr großer Dateien in Jupyter ist ohne Weiteres einzig in der IDE möglich. Hier ist durch das HDFS wesentlich mehr Speicherplatz als in dem Docker-Container des EMR-JupyterHubs verfügbar. Abschließend handelt es sich lediglich bei der entwickelten IDE um eine vollständige Integration in das Hadoop-Ökosystem, während das Starten der Spark-Applikationen über Apache Livy in den Enterprise-Produkten eher ein Zusammenspiel der Komponenten Jupyter, Hadoop

und Spark darstellt. Folglich ist ohne zusätzliche Konfigurationsschritte ausschließlich in der IDE eine Kerberos-Authentifizierung aller beteiligten Dienste vorhanden, da dies in den Enterprise-Produkten die Implementierung eines gesonderten Knox-Gateways voraussetzt (vgl. Kapitel 3.1).

Kapitel 7.

Zusammenfassung und Diskussion

Ziel dieser Masterarbeit war eine Analyse populärer cloudbasierter Enterprise-Produkte, die von Data Scientisten für die Analyse von Massendaten und die Entwicklung von Machine Learning-Algorithmen genutzt werden. Anhand der verwendeten IT-Infrastruktur, beispielsweise deren Integration in das zugrundeliegende Hadoop-Ökosystem, sollten Aussagen über Stärken und Schwächen untereinander bzw. im Vergleich zu einer eigens entwickelten IDE nach dem Vorbild von Jim Crists JupyterHub on Hadoop erfolgen (vgl. Kapitel 1).

Aufgrund der Marktstellung der Cloud-Anbieter Amazon Web Services (AWS) und Microsoft Azure werden die vorgestellten Analyseplattformen auf Elastic MapReduce (EMR) (vgl. Kapitel 3.1) und Azure Hadoop Insight (HDInsight) (vgl. Kapitel 3.2) begrenzt. Beide Cloud-Dienste bieten mit ihren jeweiligen Analyseplattformen ähnliche Funktionalitäten an und sollen ohne weitere Voraussetzungen nur mit einer Kreditkarte verwendet werden können. Im Unterschied zu AWS ist die unmittelbare Nutzung von Spark in HDInsight mit einer Kontingenterhöhung der Ressourcen verbunden. Unterschiede beider Analyseplattformen werden vor allem hinsichtlich ihrer Integration der IDE Jupyter in das zugrundeliegende Hadoop-Ökosystem deutlich. In diesem Sinne stellen sowohl AWS als auch Microsoft Azure keine *wirkliche* IDE bereit. Es handelt sich vielmehr um ein Zusammenspiel der Komponenten Jupyter, Hadoop und Spark. Dieses ist nicht zuletzt aufgrund der zusätzlichen Authentifizierungsschicht durch den REST-Server Apache Livy, einer nicht gegebenen Hochverfügbarkeit aller Komponenten sowie dem Fehlen zahlreicher Features der IDE Jupyter in seiner Gänze unvollständig. AWS wirbt damit, dass es sich bei EMR-Studio um eine IDE innerhalb einer skalierbaren Big Data-Umgebung handelt. Dennoch stellt es hinsichtlich der gegebenen IT-Infrastruktur eine Container-Lösung dar, die als Frontend zu einem davon architektonisch getrennten Hadoop-Ökosystem dient (vgl. Kapitel 3.1).

Demgegenüber liegt der klare Hauptvorteil einer eigens entwickelten IDE mittels Nutzung des von Jim Crist vorgeschlagenen Konzepts Jupyter on Hadoop (vgl. Kapitel 4.1) bei einer vollständigen Integration der IDE in das Hadoop-Ökosystem. Es können dadurch nicht nur zahlreiche Features der IDE Jupyter wie beispielsweise die Anbindung zu Git oder Dask über eine JupyterLab-Integration, sondern auch verschiedene Ablageorte für die Jupyter-Notebooks (z.B. HDFS, S3-Bucket, PostgreSQL) konfiguriert werden. Data Scientisten sind somit vollständig unabhängig vom Fortschritt und den Release-Zyklen des jeweiligen Cloud-Dienstes. Die IDE Jupyter kann vollkommen selbstständig auf die individuellen Bedürfnisse angepasst werden. Zusätzlich sind Data Scientisten gegenüber EMR und HDInsight bei der Entwicklung ihrer Analysemodelle im Sicherheitssystem von Apache Hadoop integriert und authentifiziert.

Es wurde gezeigt, dass das Konzept JupyterHub on Hadoop aufgrund der Abwesenheit des REST-Servers Apache Livy die IT-Infrastruktur insgesamt vereinfacht. Als Folge davon bietet es einen erhöhten Komfort bei der Arbeit mit Spark über Jupyter. In diesem Sinne waren für die Arbeit mit Spark sowohl in AWS als auch HDInsight mehrere Einstellungen vorzunehmen. Die zu analysierende CSV-Datei musste für die Einbindung in das EMR-JupyterHub und das Jupyter-Notebook von HDInsight vorab berechtigt werden. Bei der entwickelten IDE entfällt dieser Konfigurationsschritt, da sie im Unterschied zu EMR die Möglichkeit der Zusammenarbeit im HDFS anbietet (vgl. Kapitel 6).

Nachteil einer eigens entwickelten IDE ist jedoch vor allem der hohe Konfigurationsaufwand und damit einhergehende mögliche Schwierigkeiten bei der Wartung der entsprechenden IT-Infrastruktur. Die Nutzung von Enterprise-Produkten wie EMR und HDInsight hat dementsprechend auch Vorteile. Sowohl die Wartung der IT-Infrastruktur als auch datenschutzrechtliche Bestimmungen werden auf den Cloudanbieter ausgelagert und sparen Kosten für den jeweiligen Auftraggeber ein (vgl. Kapitel 3). Eine abschließende Entscheidung für die Einsetzung von Enterprise-Produkten oder einer eigens entwickelten IDE ist somit von der jeweiligen Situation und damit einhergehenden Anforderungen abhängig.

Kapitel 8.

Ausblick

Aus den Ergebnissen dieser Masterarbeit geht hervor, dass populäre cloudbasierte Analyseplattformen für Data Scientisten mehrere Schwächen besitzten, die mit einer eigens entwickelten IDE korrigiert werden könnten. Die automatisierte Bereitstellung dieser IDE ist in dieser Masterarbeit entwickelt worden und für jeden über ein öffentliches Git-Repository nutzbar. Der Test auf Ausfallsicherheit lässt darauf schließen, dass die entwickelte Frontend-Lösung für JupyterHub in Bezug auf den Datenverkehr der Sessions beider JupyterHub-Anwendungen nicht vollständig hochverfügbar ist (vgl. Kapitel 5.4). Eine Lösung dieser Problematik kann vermutlich über den Reverse Proxy Traefik erfolgen. Dabei wird jede Anfrage an einen der JupyterHub-Knoten vorab über eine Middleware geroutet [69]. Für die Arbeit mit Git innerhalb der IDE kann im Git-Repository ebenfalls die Ansible Collection JupyterLab erweitert werden [31]. Dafür sind die Conda- und Python-Pakete der zugehörigen Anaconda-Umgebungen¹ anzupassen. Durch eine Versionsübersicht beider Anaconda-Umgebungen² hat der Nutzer einen funktionsfähigen Stand der entwickelten IDE.

¹ siehe Git-Repository Pakete JupyterLab und Git-Repository Pakete JupyterHub

² siehe Git-Repository Versionsübersicht JupyterLab-Pakete und Git-Repository Versionsübersicht JupyterHub-Pakete

Glossar

Access Control List Mit einer Access Control List werden in Betriebssystemen und Anwendungsprogrammen Zugriffe auf Dateien und Funktionen beschränkt [48]. 40

Anaconda Mit der Software Anaconda werden zahlreiche Pakete und Bibliotheken aus den Bereichen Data Science, Wissenschaftlichem Programmieren sowie allgemeiner Entwicklung verwaltet und installiert. In Anaconda enthalten sind der mittlerweile lizenzierte Installations-Manager conda, wobei über Miniforge auch ein Open-Source Installations-Manager zur Verfügung steht. Anaconda kann mit mehreren IDEs wie beispielsweise Jupyter konfiguriert werden [11]. 25, 54

Ansible Vault Mit dem Ansible-Vault werden sensitive Daten in Ansible-Playbooks oder Inventories verschlüsselt, damit sie dort nicht im Klartext stehen [12]. 33

Apache Flink Apache Flink ist eine Framework zur Verarbeitung von Datenströmen mit geringer Verzögerung und eignet sich für die Echtzeitanalyse von Daten [13]. 20

Apache HBase Apache HBase ist eine nicht-relationale Spalten-orientierte und skalierbare Big Data-Datenbank, die auf dem HDFS aufsitzt. Daten in HBase können mithilfe von Apache Phoenix ebenfalls mit SQL analysiert werden [20]. 9, 20, 22

Apache Hive Apache Hive ist eine Data Warehouse Software, die für die Speicherung und Anfrageverarbeitung das Hadoop-Ökosystem (vgl. Kapitel 2.2.1) oder Apache HBase nutzt und eine Schnittstelle für SQL anbietet [14]. 10, 20, 22

Apache Hudi Apache Hudi (engl. Hadoop Upserts Deletes and Incrementals) ist ein Open-Source Framework zur effizienten Verwaltung von Datensätzen in verteilten Dateisystemen wie dem HDFS oder Cloud Stores wie einem S3-Bucket [44]. 20

Apache Kafka Apache Kafka wird zur Verarbeitung und Speicherung von Datenströmen verwendet. Mit einer zusätzlichen Schnittstelle wird das Laden und Exportieren von Datenströmen zu Drittsystemen ermöglicht [22]. 22

Apache Oozie Mit Apache Oozie werden Jobs im Hadoop-Ökosystem verwaltet. Es ist ein serverbasiertes Workflow-Planungssystem [15]. 22

Apache Parquet Apache Parquet ist ein Speicherformat des Hadoop-Ökosystems [16].

10

Apache Presto Apache Presto ist eine Open-Source SQL-Abfrage-Engine, mit der sich

Daten aus verschiedenen Quellen wie beispielsweise dem HDFS, PostgreSQL oder einem S3-Bucket interaktiv und verteilt analysieren lassen [68]. 20

Apache Storm Apache Storm wurde hauptsächlich in der Programmiersprache Clojure

entwickelt und ist ein verteiltes Stream-Processing Berechnungs-Framework [17].

22

Apache Zeppelin Apache Zeppelin ist ein web-basiertes Notebook, das eine datengetrie-

bene und interaktive Datenanalyse mit beispielsweise SQL, Scala, Python und R

ermöglicht und in das Hadoop-Ökosystem vollständig integriert werden kann [18].

23

Apache Zookeeper Apache Zookeeper wird für die Bereitstellung von Hochverfügbar-

keit, Performanz und Redundanz in verteilten Systemen genutzt [21]. 22, 37

Certification Authority Eine CA (Certificate Authority oder Certification Authority)

ist für die Austellung von digitalen Zertifikaten zuständig. Im Unterschied zu *self*

signed Zertifikaten vertraut diesen jeder Internetbrowser [45]. 35, 43, 46

Cloudera Cloudera ist ein Managementsystem für die Installation und Konfiguration von

Big Data-Systemen [27]. 12

Dask Dask ist eine Open-Source Library, die eine skalierte und verteilte Datenanalyse

unterstützt und in Python geschrieben ist. Es kann zwischen den Python-Paketen

Pandas, NumPy und scikit-learn und den äquivalenten Paketen in Dask gewechselt

werden. Dask wird in Jupyterlab über eine Extension eingebunden [28]. 25, 53

HAProxy HAProxy ist eine frei erhältliche Software, die einen hochverfügbaren Proxy-

Server und Load-Balancer für die Anfragen von Transmission Control Protocol

(TCP) und HTTP-basierten Applikationen bereitstellt [32]. 27, 38

Integrated Development Environment Zur Vereinfachung der Entwicklung von Pro-

grammen werden Tools benötigt, die unterstützend im gesamten Entwicklungspro-

zess wirken. Indem das Schreiben des Programmcodes mit möglichst wenig Auf-

wand in kürzester Zeit gewährleistet wird, tragen diese Tools unmittelbar zur Pro-

duktivitätssteigerung des Entwicklers bei. Derartige Tools können individuell sein

oder in einer integrierten Entwicklungsumgebung zusammengefasst werden [53, vgl. S. 1]. I, 1, 3

Keepalived Keepalived nutzt das Virtual Router Redundancy Protocol (VRRP). Eine Virtuelle IP-Adresse (VIP) wird von mehreren Systemen genutzt. Fällt der Keepalived-Master aus, wird die VIP auf den Keepalived-Standby-Knoten übertragen [40]. 26, 28, 39, 48

Kerberos Kerberos, benannt nach dem dreiköpfigen Höllen Hund und Bewacher des Eingangs zur Unterwelt, ist ein Authentifizierungsdienst für TCP/IP-basierte Netzwerke [46]. 36, 37

Kerberos Principal Die Authentifizierung von Systemen, Diensten oder Nutzern erfolgt in Kerberos mithilfe eines symmetrischen Schlüssels. Jedem Schlüssel ist ein Name, ein sogenannter Kerberos Principal, zugeordnet. Dieser setzt sich immer aus der folgenden Namenskonvention zusammen. Für Systeme host/<FQDN>@<REALM>, Dienste <servicename>/<FQDN>@<REALM> und letztlich für Nutzer <benutzer>/<FQDN>@<REALM> [75]. 37

Kerberos Propagation Mit der Kerberos-Propagation ist die Replikation der Kerberos-Datenbank von der Kerberos Master-Instanz auf einen oder mehrere Kerberos-Slaves beschrieben [61]. 37

Master-Slave-Modell Das Master-Slave-Modell steht beschreibend für ein Architekturkonzept, das die Kommunikation und den Ressourcenzugriff mehrerer Geräte, Anwendungen oder Prozesse regelt. Ein Master übernimmt Steuerfunktionen und erteilt ein oder mehreren Slaves das Recht, zu kommunizieren oder Ressourcen zu nutzen. Alternative Modelle sind das Client-Server-Modell oder das Peer-to-Peer-Modell [29]. 8

Middleware Als Middleware wird eine Software bezeichnet, die zwischen verschiedenen Systemen vermittelt und die Interaktion zwischen diesen vereinfacht [74]. 54

PostgreSQL PostgreSQL ist eine relationale Datenbanken, die als freie Software verfügbar ist [67]. 25, 38, 53, 56

Sparkmagic Sparkmagic ist eine Bibliothek, mit der man in Jupyter-Notebooks mit Spark-Clustern (vgl. Kapitel 2.2.2) über Apache Livy interagieren kann [39]. 21

ZNode Jeder Knoten innerhalb eines Zookeeper-Clusters wird als ZNode bezeichnet. 37

Literatur

- [1] A. Hong, W. Xiao und J. Ge. „Big Data Analysis System Based on Cloudera Distribution Hadoop“. In: *2021 7th IEEE Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)*. 2021, S. 169–173. doi: 10.1109/BigDataSecurityHPSCIDS52275.2021.00040.
- [2] Add a Contents Manager — JupyterHub on Hadoop 0.1.0 documentation. 1/29/2021. URL: <https://jupyterhub-on-hadoop.readthedocs.io/en/latest/contents-managers.html#storing-notebooks-on-hdfs> (besucht am 23.09.2022).
- [3] Alex Bekker. *Apache Spark vs. Hadoop MapReduce: Welches Big Data Framework sollte man wählen?* 2018. URL: <https://www.scnsoft.de/blog/spark-vs-hadoop-mapreduce> (besucht am 23.09.2022).
- [4] Amazon AWS. *EMR Notebooks*. 2021. URL: <https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-managed-notebooks.html> (besucht am 23.09.2022).
- [5] Amazon AWS. *JupyterHub*. 2021. URL: <https://docs.aws.amazon.com/emr/latest/ReleaseGuide/emr-jupyterhub.html> (besucht am 23.09.2022).
- [6] Amazon AWS. *User impersonation*. 2021. URL: <https://docs.aws.amazon.com/emr/latest/ReleaseGuide/emr-jupyterhub-user-impersonation.html> (besucht am 23.09.2022).
- [7] Amazon AWS. *Was ist Amazon EMR? Management Guide*. 2021. URL: https://docs.aws.amazon.com/de_de/emr/latest/ManagementGuide/emr-mgmt.pdf#emr-what-is-emr (besucht am 23.09.2022).
- [8] Amazon Web Services, Inc. *Amazon EMR Studio / Verwaltete IDE-Umgebung / Amazon Web Services*. 9/2/2022. URL: <https://aws.amazon.com/de/emr/features/studio/> (besucht am 23.09.2022).
- [9] American Express - Default Prediction. 11/6/2022. URL: <https://www.kaggle.com/competitions/amex-default-prediction/overview> (besucht am 18.09.2022).

- [10] AMPLab - UC Berkely. *AMP: Algorithms Machines People: Scale, immediacy, & continuous improvement*. URL: <https://amplab.cs.berkeley.edu/> (besucht am 23.09.2022).
- [11] Anaconda Inc. *Anaconda: Data science technology for groundbreaking research. a competitive edge. a better world. human sensemaking*. 2021. URL: <https://www.anaconda.com/> (besucht am 17.09.2022).
- [12] *Ansible Vault — Ansible Core Documentation*. 2022-09-15. URL: https://docs.ansible.com/ansible-core-devel/vault_guide/vault.html#vault (besucht am 17.09.2022).
- [13] Apache Software Foundation. *Apache Flink: Stateful Computations over Data Streams*. 2022. URL: <https://flink.apache.org/> (besucht am 17.09.2022).
- [14] Apache Software Foundation. *Apache Hive TM*. 2014. URL: <https://hive.apache.org/> (besucht am 17.09.2022).
- [15] Apache Software Foundation. *Apache Oozie Workflow Scheduler for Hadoop*. 2021. URL: <https://oozie.apache.org/> (besucht am 17.09.2022).
- [16] Apache Software Foundation. *Apache Parquet*. 20212. URL: <https://parquet.apache.org/> (besucht am 17.09.2022).
- [17] Apache Software Foundation. *Apache Storm*. 2022. URL: <https://storm.apache.org/> (besucht am 17.09.2022).
- [18] Apache Software Foundation. *Apache Zeppelin*. 2021. URL: <https://zeppelin.apache.org/> (besucht am 17.09.2022).
- [19] Apache Software Foundation. *HDFS High Availability Using the Quorum Journal Manager*. 2021. URL: <https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HDFSHighAvailabilityWithQJM.html> (besucht am 23.09.2022).
- [20] Apache Software Foundation. *Welcome to Apache HBase™*. 2022. URL: <https://hbase.apache.org/> (besucht am 17.09.2022).
- [21] Apache Software Foundation. *Welcome to Apache ZooKeeper™*. 2020. URL: <https://zookeeper.apache.org/> (besucht am 17.09.2022).
- [22] Apache Software Foundation. *What is Apache Kafka?* 2022. URL: <https://www.confluent.io/what-is-apache-kafka> (besucht am 17.09.2022).

- [23] Thet Hsu Aung und Wint Thida Zaw. „Improved Job Scheduling for Achieving Fairness on Apache Hadoop YARN“. In: *Proceedings of the 4th International Conference on Advanced Information Technologies*. [Piscataway, NJ]: IEEE, 2020, S. 188–193. ISBN: 978-1-7281-8364-0. DOI: 10.1109/ICAIT51105.2020.9261793.
- [24] Giuseppe Bruno u. a. „Big data processing: Is there a framework suitable for economists and statisticians?“ In: *2017 IEEE International Conference on Big Data (Big Data)*. 2017, S. 2804–2811. DOI: 10.1109/BigData.2017.8258247.
- [25] *Building an inventory — Ansible Core Documentation*. 2022-09-16. URL: https://docs.ansible.com/ansible-core-devel/getting_started/get_started_inventory.html (besucht am 17.09.2022).
- [26] Bill Chambers und Matei Zaharia. *Spark: the definitive guide: Big data processing made simple*. First edition. Beijing u. a.: O'Reilly und EBSCO Industries, February 2018. ISBN: 9781491912300.
- [27] Cloudera. *Gründe für Cloudera / Cloudera*. 2022. URL: <https://de.cloudera.com/why-cloudera.html> (besucht am 17.09.2022).
- [28] Dask core developers. *Dask: Dask natively scales Python*. 2022. URL: <https://dask.org/> (besucht am 17.09.2022).
- [29] DATACOM Buchverlag GmbH. *Master-Slave-Betrieb*. 2022. URL: <https://www.itwissen.info/Master-Slave-Betrieb-master-slave-operation.html> (besucht am 17.09.2022).
- [30] dikshantmalidev. *Hadoop – Architecture*. 2020. URL: <https://www.geeksforgeeks.org/hadoop-architecture/> (besucht am 23.09.2022).
- [31] GitHub. *jupyterlab/jupyterlab-git: A Git extension for JupyterLab*. 9/24/2022. URL: <https://github.com/jupyterlab/jupyterlab-git> (besucht am 24.09.2022).
- [32] HA Proxy Community. *HAProxy: The Reliable, High Performance TCP/HTTP Load Balancer*. 2022. URL: <http://www.haproxy.org/> (besucht am 17.09.2022).
- [33] Hetzner Online GmbH. *About Us*. 2022. URL: <https://www.hetzner.com/unternehmen/ueber-uns> (besucht am 23.09.2022).
- [34] Dave Hill. *A Comparison of Configuration Management Tools in Respect of Performance and Complexity*. 2021. URL: <https://www.davehill.ie/resources/dave-hill-thesis.pdf> (besucht am 17.09.2022).

- [35] Lorin Hochstein und Rene Moser. *Ansible: Up and Running: Automating Configuration Management and Deployment the Easy Way*. "O'Reilly Media, Inc.", 2017. ISBN: 9781491979778.
- [36] *How to build your inventory — Ansible Core Documentation: Adding variables to inventory*. 2022-09-16. URL: https://docs.ansible.com/ansible-core-devel/inventory_guide/intro_inventory.html#variables-in-inventory (besucht am 17.09.2022).
- [37] *Integration with Dask — JupyterHub on Hadoop 0.1.0 documentation*. 1/29/2021. URL: <https://jupyterhub-on-hadoop.readthedocs.io/en/latest/dask.html> (besucht am 23.09.2022).
- [38] Jim Christ. *JupyterHub on Hadoop*. 2019. URL: <https://jupyterhub-on-hadoop.readthedocs.io/en/latest/> (besucht am 23.09.2022).
- [39] Jupyter Development Team. *sparkmagic*. 2022. URL: <https://github.com/jupyter-incubator/sparkmagic> (besucht am 17.09.2022).
- [40] *Keepalived — Open Source Admin-Handbuch der Linuxfabrik*. 6/3/2022. URL: <https://docs.linuxfabrik.ch/software/keepalived.html> (besucht am 23.09.2022).
- [41] Kevin Risden. *Apache Knox - Apache Livy Service*. 2018. URL: <https://risdenk.github.io/2018/03/02/apache-knox-apache-livy-service.html> (besucht am 23.09.2022).
- [42] Hakan Kocakulak und Tugba Taskaya Temizel. „A Hadoop solution for ballistic image analysis and recognition“. In: 2011, S. 836–842. DOI: 10.1109/HPCSim.2011.5999917. URL: https://www.researchgate.net/publication/224255467_A_Hadoop_solution_for_ballistic_image_analysis_and_recognition.
- [43] Laurenz Wuttke. *Einführung in Apache Spark: Komponenten, Vorteile und Anwendungsbereiche*. 2020. URL: <https://datasolut.com/was-ist-spark/> (besucht am 23.09.2022).
- [44] Stefan Luber. *Was ist Apache Hudi?* 2021-03-11. URL: <https://www.bigdata-insider.de/was-ist-apache-hudi-a-1006518/> (besucht am 17.09.2022).
- [45] Stefan Luber. „Was ist eine CA?“ In: *Security-Insider* (19.03.2018). URL: <https://www.security-insider.de/was-ist-eine-ca-a-696169/> (besucht am 17.09.2022).
- [46] Stefan Luber und Peter Schmitz. *Definition Kerberos: Was ist Kerberos?* 2019. URL: <https://www.security-insider.de/was-ist-kerberos-a-887891/> (besucht am 23.09.2022).

- [47] Lukas Kaemmerling. *hetzner.hcloud.hcloud_server: Create and manage cloud servers on the Hetzner Cloud*. 2021. URL: https://docs.ansible.com/ansible/latest/collections/hetzner/hcloud/hcloud_server_module.html (besucht am 23.09.2022).
- [48] Ben Lutkevich. *What is Access Control List (ACL)? - SearchSoftwareQuality*. 2022-09-17. URL: <https://www.techtarget.com/searchnetworking/definition/access-control-list-ACL> (besucht am 17.09.2022).
- [49] Sam Malayek. „High-Level vs Low-Level | by Sam Malayek | Level Up Coding“. In: *Level Up Coding* (25.4.2022). URL: <https://levelup.gitconnected.com/high-level-vs-low-level-54b8faea4ffa> (besucht am 25.09.2022).
- [50] Microsoft. *Installieren von Jupyter Notebook auf Ihrem Computer und Herstellen einer Verbindung mit Apache Spark in HDInsight*. 2021. URL: <https://docs.microsoft.com/de-de/azure/hdinsight/spark/apache-spark-jupyter-notebook-install-locally> (besucht am 23.09.2022).
- [51] Microsoft. *Verwenden von Apache Zeppelin Notebooks mit Apache Spark-Cluster in Azure HDInsight*. 2021. URL: <https://docs.microsoft.com/de-de/azure/hdinsight/spark/apache-spark-zeppelin-notebook> (besucht am 23.09.2022).
- [52] Microsoft. *Worum handelt es sich bei Azure HDInsight?* 2021. URL: <https://docs.microsoft.com/de-de/azure/hdinsight/overview> (besucht am 23.09.2022).
- [53] Ali Mustafa Kattan, Rosni Abdullah und Rosalina Salam. „Java: From "Hard Coding" to Using an Integrated Development Environment“. In: *The Second International Conference on Distributed Frameworks for Multimedia Applications*. Hrsg. von Rahmat Budiarto. Piscataway, NJ: IEEE Operations Center, 2006, S. 1–6. ISBN: 1-4244-0409-6. DOI: [10.1109/DFMA.2006.296905](https://doi.org/10.1109/DFMA.2006.296905).
- [54] Nurmukhammad Begiboev. *Was ist der Unterschied zwischen RDD, DataFrame und Dataset in Apache Spark?* 2020-05-18. URL: <https://blog.oio.de/2020/05/18/was-ist-der-unterschied-zwischen-rdd-dataframe-und-dataset-in-apache-spark/> (besucht am 23.09.2022).
- [55] Jorge Piazzentin Ono u. a. „Interactive Data Visualization in Jupyter Notebooks“. In: *Computing in Science & Engineering* 23.2 (2021), S. 99–106. ISSN: 1521-9615. DOI: [10.1109/MCSE.2021.3052619](https://doi.org/10.1109/MCSE.2021.3052619).

- [56] Raj Bala, Bob Gill, Dennis Smith, Kevin Ji, David Wright. *Magic Quadrant for Cloud Infrastructure and Platform Services*. 2021. URL: <https://www.gartner.com/doc/reprints?id=1-2710E4VR&ct=210802&st=sb> (besucht am 23.09.2022).
- [57] *Roles — Ansible Documentation: Role directory structure*. 2022-09-15. URL: https://docs.ansible.com/ansible/latest/user_guide/playbooks_reuse_roles.html#role-directory-structure (besucht am 17.09.2022).
- [58] *Run Your First Command and Playbook — Ansible Documentation: Create and run your first network Ansible Playbook*. 2022-09-15. URL: https://docs.ansible.com/ansible/latest/network/getting_started/first_playbook.html#run-your-first-network-ansible-command (besucht am 17.09.2022).
- [59] *Run Your First Command and Playbook — Ansible Documentation: Run your first network Ansible command*. 2022-09-15. URL: https://docs.ansible.com/ansible/latest/network/getting_started/first_playbook.html#create-and-run-your-first-network-ansible-playbook (besucht am 17.09.2022).
- [60] Semi Koen. *Jupyter is the new Excel*. 2019. URL: <https://towardsdatascience.com/jupyter-is-the-new-excel-a7a22f2fc13a> (besucht am 23.09.2022).
- [61] *Set Up the Slave KDCs for Database Propagation - Kerberos V5 Installation Guide*. 2007. URL: <https://web.mit.edu/kerberos/krb5-1.5/krb5-1.5.4/doc/krb5-install/Set-Up-the-Slave-KDCs-for-Database-Propagation.html>.
- [62] Sunil Goyal. *Hadoop Vs Spark: Choosing the Right Big Data Framework*. 2019. URL: <https://www.netsolutions.com/insights/hadoop-vs-spark/> (besucht am 23.09.2022).
- [63] *Tags — Ansible Documentation: Selecting or skipping tags when you run a playbook*. 2022-09-15. URL: https://docs.ansible.com/ansible/latest/user_guide/playbooks_tags.html#selecting-or-skipping-tags-when-you-run-a-playbook (besucht am 17.09.2022).
- [64] Talend. *Was ist MapReduce?* 2021. URL: <https://www.talend.com/de/resources/what-is-mapreduce/> (besucht am 23.09.2022).
- [65] Anton Teslyuk u. a. „Architecture and deployment details of scalable Jupyter environment at Kurchatov Institute supercomputing centre“. In: *2020 Ivannikov Memorial Workshop (IVMEM)*. IEEE, 9/25/2020 - 9/26/2020, S. 59–61. ISBN: 978-1-7281-9088-4. DOI: 10.1109/IVMEM51402.2020.00017.

- [66] The Carpentries. *Getting Started with JupyterLab: Creating a Jupyter Notebook*. 2021. URL: <https://swcarpentry.github.io/python-novice-gapminder/01-run-quit/> (besucht am 23.09.2022).
- [67] The PostgreSQL Global Development Group. *PostgreSQL: The World's Most Advanced Open Source Relational Database*. 2022. URL: <https://www.postgresql.org/> (besucht am 17.09.2022).
- [68] The Presto Foundation. *presto: Distributed SQL Query Engine for Big Data*. 2022. URL: <https://prestodb.io/> (besucht am 17.09.2022).
- [69] Traefik.io. *Traefik Proxy Documentation - Traefik*. 2022. URL: <https://doc.traefik.io/traefik/>.
- [70] *Truly thrifty cloud hosting - Hetzner Online GmbH*. 9/21/2022. URL: <https://www.hetzner.com/cloud> (besucht am 23.09.2022).
- [71] *Use JupyterLab by default — JupyterHub on Hadoop 0.1.0 documentation*. 1/29/2021. URL: <https://jupyterhub-on-hadoop.readthedocs.io/en/latest/jupyterlab.html> (besucht am 23.09.2022).
- [72] *Using collections — Ansible Documentation*. 2022-09-15. URL: https://docs.ansible.com/ansible/latest/user_guide/collections_using.html#using-collections (besucht am 17.09.2022).
- [73] Jiawei Wang, Li Li und Andreas Zeller. „Better code, better sharing“. In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: New Ideas and Emerging Results*. Hrsg. von Gregg Rothermel und Doo-Hwan Bae. New York, NY, USA: ACM, 2020, S. 53–56. ISBN: 9781450371261. DOI: 10.1145/3377816.3381724.
- [74] „Was ist Middleware und wie funktioniert sie?“ In: *Talend ()*. URL: <https://www.talend.com/de/resources/what-is-middleware/> (besucht am 18.09.2022).
- [75] *What is a Kerberos Principal? - Kerberos V5 UNIX User's Guide*. 2007. URL: https://web.mit.edu/kerberos/krb5-1.5/krb5-1.5.4/doc/krb5-user/What-is-a-Kerberos-Principal_003f.html (besucht am 17.09.2022).
- [76] Ling Wu, Guangtai Liang und Qianxiang Wang. „Program Behavior Analysis and Control for Online IDE“. In: *2012 IEEE 36th Annual Computer Software and Applications Conference Workshops*. IEEE, 2012. DOI: 10.1109/compsacw.2012.42.
- [77] Ling Wu u. a. „CEclipse: An Online IDE for Programming in the Cloud“. In: *IEEE World Congress on Services (SERVICES), 2011*. Piscataway, NJ: IEEE, 2011, S. 45–52. ISBN: 978-1-4577-0879-4. DOI: 10.1109/SERVICES.2011.74.

Anhang A.

Ansible

A.1. ansible.cfg

```
[defaults]

host_key_checking = False
nocows = 1
gathering = smart
force_valid_group_names = silently
interpreter_python = /usr/bin/python3
retry_files_enabled = False
inventory = ./inventory
vault_password_file = ~/.vault_pass.txt
private_key_file = ~/.ssh/hetzner
collections_path = ./collections/ansible_collections
roles_path = ./roles

[inventory]

enable_plugins = yaml, ini
```

A.2. setup.yml

```
- hosts: ansible
tags:
  - hetzner
roles:
  - name: hetzner.cloud.servers_and_ips
    when: custom_inventory_file is not defined
post_tasks:
  - name: refresh dynamic changed inventory
    meta: refresh_inventory

- hosts: all,!ansible
become: yes
tags:
  - dns
roles:
  - name: hetzner.cloud.dns
    when: custom_inventory_file is not defined

- hosts: all,!ansible
become: yes
tags:
  - dns
tasks:
  - name: refresh dynamic changed inventory
    delegate_to: localhost
    become: no
    run_once: yes
    meta: refresh_inventory

- hosts: all,!ansible
become: yes
tags:
  - dns
  - reverse_dns
roles:
  - name: hetzner.cloud.reverse_dns
    when: custom_inventory_file is not defined
```

Anhang B.

Custom IDE

B.1. Preflight Check vor IDE-Installation

```
- name: check that group_vars/all.yml exists
  stat:
    path: "{{ playbook_dir }}/group_vars/all.yml"
  register: all_yml

- name: print message depending on group_vars/all.yml check
  assert:
    that: "all_yml.stat.exists"
    fail_msg: "File group_vars/all.yml is not present, please define it for IDE setup. See {{ playbook_dir }}/examples/all.yml for examples."
    success_msg: "File group_vars/all.yml is present."

- name: print message depending on variable check
  assert:
    that: "{{ item }} is defined"
    fail_msg: "Variable {{ item }} is not defined in group_vars/all.yml, please define it for IDE setup."
    success_msg: "Variable {{ item }} is defined."
loop:
  - ansible_private_key
  - ansible_public_key
  - domain
  - realm
  - realm_password
  - my_email
  - truststore_password
  - keystore_password
  - keytab_folder
  - hadoop_nameservice
  - haproxy_admin_password

- name: print message depending on variable check for Hetzner Cloud deployment
  assert:
    that: hetzner_api_token is defined
```

```
fail_msg: "Hetzner API token is not defined using variable hetzner_api_token  
in group_vars/all.yml, please define it for deployment in Hetzner Cloud."  
success_msg: "Hetzner API token is defined and will be used for deployment in  
Hetzner Cloud."  
when: custom_inventory_file is not defined  
  
- name: check if defined domain is reachable  
  wait_for:  
    host: "{{ domain }}"  
    port: 443  
    timeout: 3  
  register: domain_reachable  
  failed_when: false  
  
- name: print message depending on domain check  
  assert:  
    that: domain_reachable.state is defined and domain_reachable.state == "started"  
    "  
  fail_msg: "Configured domain {{ domain }} is not reachable, check your network  
  settings."  
  success_msg: "Configured domain {{ domain }} is reachable and will be used for  
  installation."
```

B.2. LDAP-Konfiguration des Hadoop-Ökosystems

```
<property>
  <name>hadoop.security.group.mapping.provider.ad4users.ldap.bind.user</name>
  <value>cn=admin,{{ ldap_organization }}</value>
</property>
<property>
  <name>hadoop.security.group.mapping.provider.ad4users.ldap.bind.password</name>
  <value>{{ ldap_password }}</value>
</property>
<property>
  <name>hadoop.security.group.mapping.provider.ad4users.ldap.base</name>
  <value>{{ ldap_organization }}</value>
</property>
<property>
  <name>hadoop.security.group.mapping.provider.ad4users.ldap.userbase</name>
  <value>ou=people,{{ ldap_organization }}</value>
</property>
<property>
  <name>hadoop.security.group.mapping.provider.ad4users.ldap.groupbase</name>
  <value>ou=groups,{{ ldap_organization }}</value>
</property>
<property>
  <name>hadoop.security.group.mapping.provider.ad4users.ldap.search.filter.user</name>
  <value>(&#38;(objectClass=posixAccount)(uid={0}))</value>
</property>
<property>
  <name>hadoop.security.group.mapping.provider.ad4users.ldap.search.filter.group</name>
  <value>(objectClass posixGroup)</value>
</property>
<property>
  <name>hadoop.security.group.mapping.provider.ad4users.ldap.search.attr.memberof</name>
  <value>memberOf </value>
</property>
```

B.3. Ansible Play Recap nach Installation über setup.yml

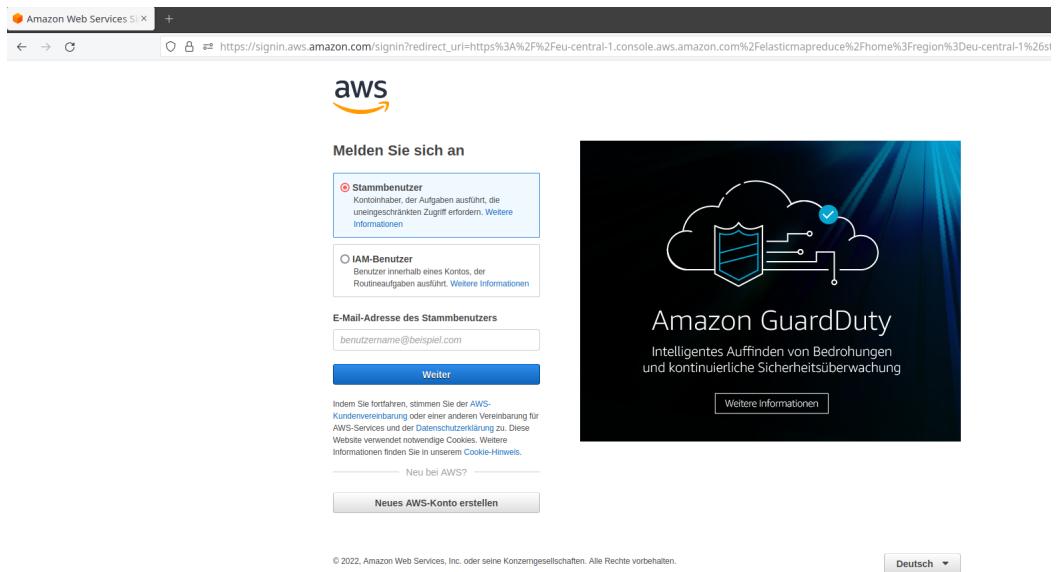
```
PLAY RECAP ****
135.181.110.167      : ok=6    changed=4    unreachable=0    failed=0     skipped=0    rescued=0
ignored=0
65.108.153.196      : ok=3    changed=2    unreachable=0    failed=0     skipped=0    rescued=0
ignored=0
65.108.250.170      : ok=3    changed=2    unreachable=0    failed=0     skipped=0    rescued=0
ignored=0
65.108.253.43       : ok=3    changed=2    unreachable=0    failed=0     skipped=0    rescued=0
ignored=0
65.108.49.219       : ok=3    changed=2    unreachable=0    failed=0     skipped=0    rescued=0
ignored=0
65.109.9.237        : ok=3    changed=2    unreachable=0    failed=0     skipped=0    rescued=0
ignored=0
65.21.249.131       : ok=3    changed=2    unreachable=0    failed=0     skipped=0    rescued=0
ignored=0
95.216.148.85       : ok=3    changed=2    unreachable=0    failed=0     skipped=0    rescued=0
ignored=0
95.216.174.250      : ok=3    changed=2    unreachable=0    failed=0     skipped=0    rescued=0
ignored=0
95.216.221.32       : ok=3    changed=2    unreachable=0    failed=0     skipped=0    rescued=0
ignored=0
hub1.click-your-it.de : ok=102   changed=77    unreachable=0    failed=0     skipped=6     rescued=0
ignored=0
hub2.click-your-it.de : ok=101   changed=77    unreachable=0    failed=0     skipped=6     rescued=0
ignored=0
localhost            : ok=20    changed=6     unreachable=0    failed=0     skipped=0     rescued=0
ignored=0
master1.click-your-it.de : ok=162   changed=117   unreachable=0    failed=0     skipped=17    rescued=0
ignored=0
master2.click-your-it.de : ok=146   changed=108   unreachable=0    failed=0     skipped=19    rescued=0
ignored=0
master3.click-your-it.de : ok=146   changed=108   unreachable=0    failed=0     skipped=19    rescued=0
ignored=0
security1.click-your-it.de : ok=163   changed=132   unreachable=0    failed=0     skipped=10    rescued=0
ignored=0
security2.click-your-it.de : ok=160   changed=128   unreachable=0    failed=0     skipped=13    rescued=0
ignored=0
worker1.click-your-it.de : ok=167   changed=129   unreachable=0    failed=0     skipped=18    rescued=0
ignored=0
worker2.click-your-it.de : ok=158   changed=120   unreachable=0    failed=0     skipped=15    rescued=0
ignored=0
worker3.click-your-it.de : ok=158   changed=120   unreachable=0    failed=0     skipped=15    rescued=0
ignored=0

Sonntag 25 September 2022 08:56:06 +0200 (0:00:03.197)          0:37:24.865 ****
```

Anhang C.

Amazon EMR

C.1. Anmeldung AWS



C.2. Startseite AWS-Konsole

The screenshot shows the AWS Management Console search results for the term 'EMR'. The search bar at the top contains 'EMR'. The results are categorized into 'Services' and 'Funktionen'.

Services (9)

- Funktionen (5)
 - Blogs (1)
 - Dokumentation (46.214)
 - Fachartikel (30)
 - Tutorials (1)
 - Marketplace (107)
- EMR ☆
Verwaltetes Hadoop-Framework
- AWS Glue DataBrew ☆
Visuelles Datenvorbereitungstool zum Bereinigen und Normalisieren von Daten für Ana...
- MediaStore ☆
Speichern und liefern Sie Videokomponenten für Live- oder On-Demand-Medien-Workf...
- MediaConvert ☆
Konvertieren datenbasierter Inhalte zur Übertragung und Multiscreen-Bereitstellung

Funktionen

- Private Registry
Elastic Container Registry-Funktion
- Repository
Elastic Container Registry-Funktion

C.3. EMR - AWS-Konsole

The screenshot shows the AWS EMR console interface. The top navigation bar includes the AWS logo, the service name "Amazon EMR", and a search bar. Below the navigation is a toolbar with four buttons: "Cluster erstellen" (highlighted in blue), "Details ansehen", "Klonen", and "Beenden". A dropdown menu labeled "Filter:" shows "Aktive Cluster". To the right, it says "0 Cluster (alle geladen)" and has a refresh icon. On the left, a sidebar lists various EMR features: EMR Studio, EMR on EC2, Cluster (which is selected and highlighted in orange), Notebooks, Git repositories, Sicherheitskonfiguration, Öffentlichen Zugriff blockieren, VPC-Subnetze, and Ereignisse.

C.4. EMR Cluster - Hauptkonfiguration

The screenshot shows the AWS EMR console interface for creating a new cluster. The URL is https://eu-central-1.console.aws.amazon.com/elasticmapreduce/home?region=eu-central-1#quick-create:

Allgemeine Konfiguration

- Cluster-Name: jupyterhub
- Protokollierung (checked)
- S3-Ordner: s3://aws-logs-864064620164-eu-central-1/elasticm
- Start-Modus: Cluster (selected)

Softwarekonfiguration

- Freigabe: emr-5.36.0
- Anwendungen:
 - Core Hadoop: Hadoop 2.10.1, Hive 2.3.9, Hue 4.10.0, Mahout 0.13.0, Pig 0.17.0, and Tez 0.9.2
 - HBase: HBase 1.4.13, Hadoop 2.10.1, Hive 2.3.9, Hue 4.10.0, Phoenix 4.14.3, and ZooKeeper 3.4.14
 - Presto: Presto 0.267 with Hadoop 2.10.1 HDFS and Hive 2.3.9 Metastore
 - Spark: Spark 2.4.8 on Hadoop 2.10.1 YARN and Zeppelin 0.10.0
- Verwenden des AWS Glue-Datenkatalogs für Tabellen-Metadaten

Hardwarekonfiguration

- Instance-Typ: m5.xlarge
- Anzahl der Instances: 1 (1 Master und 0 Core-Knoten)
- Cluster scaling: scale cluster nodes based on workload
- Automatische Beendigung: Automatische Beendigung [Weitere Informationen](#)
- Beenden Sie den Cluster, wenn er sich nach Stunden (0 Minuten) Minuten befindet

C.5. EMR Cluster - erweiterte Konfiguration

Cluster erstellen - Erweiterte Optionen für das Erstellen eines Clusters [Gehen Sie zu „Optionen für schnelles Erstellen“](#)

Step 1: Software und Schritte

- Step 2: Hardware
- Step 3: Allgemeine Cluster-Einstellungen
- Step 4: Sicherheit

Softwarekonfiguration

Freigabe: emr-5.36.0

<input checked="" type="checkbox"/> Hadoop 2.10.1	<input type="checkbox"/> Zeppelin 0.10.0	<input type="checkbox"/> Livy 0.7.1
<input type="checkbox"/> JupyterHub 1.4.1	<input type="checkbox"/> Tez 0.9.2	<input type="checkbox"/> Flink 1.14.2
<input type="checkbox"/> Ganglia 3.7.2	<input type="checkbox"/> HBase 1.4.13	<input checked="" type="checkbox"/> Pig 0.17.0
<input checked="" type="checkbox"/> Hive 2.3.9	<input type="checkbox"/> Presto 0.267	<input type="checkbox"/> ZooKeeper 3.4.14
<input type="checkbox"/> JupyterEnterpriseGateway 2.1.0	<input type="checkbox"/> MXNet 1.8.0	<input type="checkbox"/> Sqoop 1.4.7
<input type="checkbox"/> Mahout 0.13.0	<input checked="" type="checkbox"/> Hue 4.10.0	<input type="checkbox"/> Phoenix 4.14.3
<input type="checkbox"/> Oozie 5.2.1	<input type="checkbox"/> Spark 2.4.8	<input type="checkbox"/> HCatalog 2.3.9
<input type="checkbox"/> TensorFlow 2.4.1		

Mehrere Master-Knoten (optional)

Verwenden Sie mehrere Master-Knoten, um die Cluster-Verfügbarkeit zu verbessern. [Weitere Informationen](#)

Einstellungen für den AWS Glue-Datenkatalog (optional)

Verwendung für Hive-Tabellen-Metadaten

C.6. EMR Cluster - Maschinenanzahl

EC2 Subnetz: subnet-055a919938515fec5 | Standard in eu-central-1a

Cluster Nodes and Instances

Wählen Sie den Instance-Typ, die Anzahl der Instances und eine Kaufoption. Sie können On-Demand-Instances, Spot-Instances oder beides verwenden. Der Instance-Typ und die Kaufoption gelten für alle EC2-Instances in jeder Instance-Gruppe. Sie können diese Optionen nur bei der Erstellung einer Instance-Gruppe angeben. [Weitere Informationen zu Instance-Kaufoptionen](#)

Console options for automatic scaling have changed. [Learn more](#)

Knotentyp	Instance-Typ	Instance-Anzahl	Kaufoption
Master	m5.xlarge	1 Instances	<input checked="" type="radio"/> On-Demand <input type="radio"/> Spot <input type="checkbox"/> Use on-demand as max price
Core	m5.xlarge	0 Instances	<input checked="" type="radio"/> On-Demand <input type="radio"/> Spot <input type="checkbox"/> Use on-demand as max price
Aufgabe	m5.xlarge	0 Instances	<input checked="" type="radio"/> On-Demand <input type="radio"/> Spot <input type="checkbox"/> Use on-demand as max price

+ Aufgaben-Instance-Gruppe hinzufügen

Total core and task units: 0 Einheiten gesamt

C.7. EMR Cluster - Namen vergeben

Cluster erstellen - Erweiterte Optionen für das Erstellen eines Clusters [Gehen Sie zu „Optionen“](#)

Step 1: Software und Schritte
Step 2: Hardware
Step 3: Allgemeine Cluster-Einstellungen
Step 4: Sicherheit

Allgemeine Optionen

Cluster-Name:

Protokollierung i

S3-Ordner ...

Protokollverschlüsselung i

Debuggen i

Beendigungsschutz i

C.8. EMR Cluster - EC2-Schlüsselpaar

Cluster erstellen - Erweiterte Optionen für das Erstellen eines Clusters [Gehen Sie zu „Optionen für schnelles Erstellen“](#)

Step 1: Software und Schritte
Step 2: Hardware
Step 3: Allgemeine Cluster-Einstellungen
Step 4: Sicherheit

Sicherheitsoptionen

EC2-Schlüsselpaar: i

Cluster ist für alle IAM-Benutzer im Konto sichtbar i

Berechtigungen i

Standard Benutzerdefiniert
Verwenden Sie Standard-IAM-Rollen. Wenn keine Rollen vorhanden sind, werden diese automatisch für Sie mit den verwalteten Richtlinien für automatische Richtlinienaktualisierungen erstellt.

EMR-Rolle: ... Use EMR_DefaultRole_V2 i

EC2-Instance-Profil: ...

Auto Scaling-Rolle: ...

Sicherheitskonfiguration
 EC2-Sicherheitsgruppen

[Cancel](#) [Previous](#) [Cluster erstellen](#)

C.9. EMR Cluster - Anwendungsverlauf

The screenshot shows the AWS EMR console interface. On the left, there's a sidebar with navigation links for Amazon EMR, EMR Studio, EMR on EC2 (with 'Cluster' selected), Notebooks, Git repositories, Sicherheitskonfiguration, Öffentlichen Zugriff blockieren, VPC-Subnetze, Ereignisse, EMR in EKS, and Virtuelle Cluster. The main content area has tabs for 'Klonen', 'Beenden', and 'AWS-CLI-Export'. Below that, it says 'Cluster: jupyterhub' and 'Warten' (Cluster ready after last step completed). There are tabs for 'Übersicht', 'Anwendungsverlauf' (selected), 'Überwachung', 'Hardware', 'Konfigurationen', 'Ereignisse', 'Schritte', and 'Bootstrap-Aktionen'. Under 'Anwendungsverlauf', it lists 'Persistent application user interfaces' (YARN timeline server, Tez UI, Spark history server) and 'On-cluster application user interfaces' (HDFS-Namenknoten, Hue, JupyterHub, Spark-Verlaufsserver, Ressourcen-Manager). A note says 'On-cluster UI are available only while clusters are running. Because they are hosted on the master node, on-cluster UI require a connection via SSH tunneling.' At the bottom, there's a table with columns 'Application' and 'User interface URL'.

C.10. EMR Cluster - Security Groups

The screenshot shows the AWS EC2 Management Console. The left sidebar includes 'New EC2 Experience', 'EC2-Dashboard', 'Globale Ansicht von EC2', 'Ereignisse', 'Tags', 'Beschränkungen', and 'Instances' (selected). The main area shows a table titled 'Sicherheitsgruppen [2] Info' with columns: Name, Sicherheitsgruppen-ID, Name der Sicherheit..., VPC-ID, Beschreibung, Besitzer, Anzahl der Regeln ..., and Anzahl. It lists two security groups: 'sg-0d66bc4bcf7b27c15' (Master group for Elastic...) and 'sg-0fe10fb960339243' (Slave group for Elastic...).

C.11. EMR Cluster - Regeln für Datenverkehr bearbeiten

The screenshot shows the AWS EC2 Management Console interface for managing security groups. The URL is https://eu-central-1.console.aws.amazon.com/ec2/home?region=eu-central-1#SecurityGroup:groupId=sg-0d66bc4bcf7b27c15. The main page displays the details of the security group 'sg-0d66bc4bcf7b27c15 - ElasticMapReduce-master'. It shows the name, VPC ID (vpc-0c6deca2fbdd7a753), and creation date (2022-09-02T08:08:49.593Z). Below this, it lists the owner (864064620164) and the number of rules for incoming and outgoing traffic. The 'Regeln für eingehenden Datenverkehr' tab is selected, showing 8 rules. A button 'Regeln für eingehenden Datenverkehr bearbeiten' is visible at the bottom right.

C.12. EMR Cluster - JupyterHub-Port hinzufügen

The screenshot shows the 'Regeln für eingehenden Datenverkehr' (Incoming Traffic Rules) configuration page. The URL is https://eu-central-1.console.aws.amazon.com/ec2/home?region=eu-central-1#ModifyInboundSecurityGroupRules:securityGroupId=sg-0d66bc4bcf7b27c15. The table lists several existing rules, including ones for UDP ports 0-65535 and TCP port 8443. A new rule is being added at the bottom, defined by 'Alle UDP' (All UDP) as the protocol, '0 - 65535' as the port range, and 'Benutzerdefiniertes TCP' (Custom TCP) as the source. The destination is set to '9443' and the target is 'Anywhere'. A note indicates that the port 8443 is already defined in another rule. A 'Regel hinzufügen' (Add Rule) button is at the bottom left, and a 'Regeln speichern' (Save Rules) button is at the bottom right.

C.13. AWS-Konsole - S3

The screenshot shows the AWS Management Console search results for 'S3'. The search bar at the top contains 's3'. On the left, there is a sidebar titled 'Services (7)' with links to 'Funktionen (12)', 'Blogs (4)', 'Dokumentation (112.111)', 'Fachartikel (30)', 'Tutorials (7)', and 'Marketplace (815)'. The main area is titled 'Suchergebnisse für "S3"' and lists three services: 'S3' (Skalierbare Speicherkapazität in der Cloud), 'S3 Glacier' (Archivspeicherung in der Cloud), and 'Athena' (Daten in S3 mit SQL abfragen). A right sidebar shows 'standardlayout zurücksetzen' and icons for 'Willkommen', 'Start', 'Feedback', and 'Weitere Informationen'.

C.14. Amazon S3 - Bucket erstellen

The screenshot shows the AWS S3 Management Console. The left sidebar has 'Amazon S3' selected under 'Buckets'. It includes links for 'Access Points', 'Objekt-Lambda-Zugriffspunkte', 'Multi-Region-Zugriffspunkte', 'Batch-Operationen', and 'Zugriffs-Analyser für S3'. Below that, it says 'Einstellungen "Öffentlichen Zugriff beschränken" für dieses Konto'. The main area shows a 'Konto-Snapshot' with a note about storage usage and trends. The 'Buckets (2)' section shows two buckets: 'aws-eini-resources-064064620164-eu-central-1' and 'aws-logo-854044620164-eu-central-1'. A red button at the top right says 'Bucket erstellen' (Create Bucket).

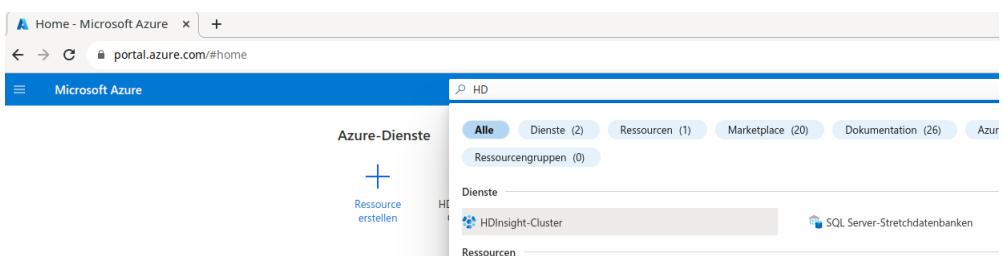
C.15. Amazon S3 - Blockade öffentlicher Zugriffe deaktivieren

The screenshot shows the AWS S3 Bucket creation interface. The URL is https://s3.console.aws.amazon.com/s3/bucket/create?region=eu-central-1. The page title is "Bucket erstellen". The "Allgemeine Konfiguration" tab is selected. In the "Bucket-Name" field, "jupyterlargefiles" is entered. The "AWS-Region" is set to "EU (Frankfurt) eu-central-1". Under "Einstellungen aus einem vorhandenen Bucket kopieren – optional", there is a note: "Nur die Bucket-Einstellungen in der folgenden Konfiguration werden kopiert." A "Bucket auswählen" button is present. The "Objekteigentümerschaft" section contains two radio buttons: "ACLs deaktiviert (empfohlen)" (selected) and "ACLs aktiviert". Below these, under "Einstellungen 'Öffentlichen Zugriff beschränken' für diesen Bucket", there is a note: "Der öffentliche Zugriff auf Buckets und Objekte wird durch Zugriffssteuerungslisten (ACLs, Access Control Lists), Bucket-Richtlinien, Zugriffspunktkontrolle und alle diese Elemente gewährt. Um sicherzustellen, dass der öffentliche Zugriff auf alle Ihre Buckets und Objekte blockiert wird, aktivieren Sie die Blockierung jeglichen öffentlichen Zugriffs. Diese Einstellungen gelten nur für diesen Bucket und die dazugehörigen Zugriffssprünge. AWS empfiehlt, dass Sie die Blockierung jeglichen öffentlichen Zugriffs aktivieren. Stellen Sie vor der Aktivierung dieser Einstellung sicher, dass Ihre Anwendung korrekt funktioniert. Sollten Sie einen bestimmten Grad des öffentlichen Zugriffs auf diesen Bucket oder seine enthaltenen Objekte benötigen, so können Sie unten die jeweiligen Einstellungen an Ihre spezifischen Speicherbenutzungsfälle anpassen. Weitere Informationen." A checkbox for "Blockieren des gesamten öffentlichen Zugriffs" is checked. The status message at the top right says: "Wir verbessern die S3-Konsole weiter, um sie schneller und einfacher verwendbar zu machen. Wenn Sie Feedback zum aktualisierten Erlebnis haben, wählen Sie Feedback geben aus."

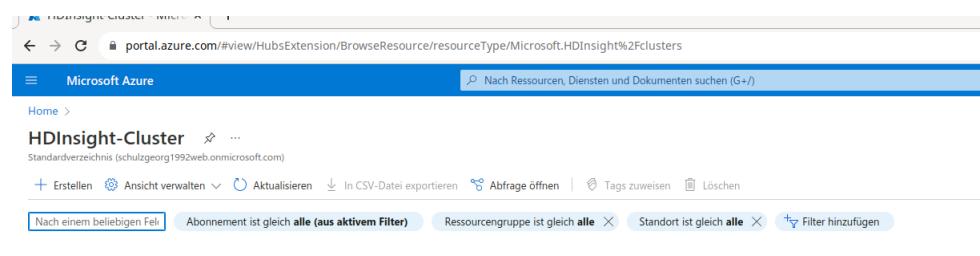
Anhang D.

Azure HDInsight

D.1. Azure-Portal - nach HDInsight suchen und erstellen



The screenshot shows the Microsoft Azure portal interface. The search bar at the top contains the text 'HD'. In the main results pane, under the heading 'Dienste', there is a single result: 'HDInsight-Cluster'. This result is highlighted with a gray background and has a small blue icon next to it. Other service categories like 'Ressourcen' and 'Marketplace' are also listed but are not highlighted.



The screenshot shows the Azure portal's 'HDInsight-Cluster' resource page. The URL in the address bar is 'portal.azure.com/#view/HubsExtension/BrowseResource/resourceType/Microsoft.HDInsight%2Clusters'. The page header says 'HDInsight-Cluster' and shows the user's name 'Standardverzeichnis (schulzgeorg1992web.onmicrosoft.com)'. Below the header are several buttons: '+ Erstellen', 'Ansicht verwalten', 'Aktualisieren', 'In CSV-Datei exportieren', 'Abfrage öffnen', 'Tags zuweisen', and 'Löschen'. There are also filter buttons for 'Nach einem beliebigen Feld', 'Abonnement ist gleich alle (aus aktivem Filter)', 'Ressourcengruppe ist gleich alle', 'Standort ist gleich alle', and 'Filter hinzufügen'. The main content area is a table with four columns: 'Name' (sorted by ascending), 'Clustertyp' (sorted by descending), 'Status' (sorted by ascending), and 'Ressourcengruppe'. The table is currently empty. At the bottom of the page, there is a message: 'Keine HDInsight-Cluster zum Anzeigen.' followed by a note: 'Erstellen Sie einen HDInsight-Cluster zur Verarbeitung großer Datenmengen mithilfe gängiger Ope...'. Below this is a button labeled 'HDInsight-Cluster erstellen'.

D.2. HDInsight - Cluster konfigurieren

The screenshot shows the 'HDInsight-Cluster erstellen' (Create HDInsight Cluster) page in the Microsoft Azure portal. The form is divided into several sections:

- Abonnement ***: Masterarbeit (selected)
- Ressourcengruppe ***: jupyter_azure (selected)
- Clusterdetails**:
 - Clustername ***: jupyter
 - Region ***: Germany West Central
 - Verfügbarkeitszone**: (dropdown menu)
 - Clustertyp ***: Spark (selected)
 - Version ***: Spark 2.4 (HDI 4.0) (selected)
- Clusteranmeldeinformationen**:
 - Benutzername für Clusteranmeldung *: admin
 - Kennwort für Clusteranmeldung *: (redacted)
 - Clusteranmeldekennwort bestätigen *: (redacted)
 - SSH-Benutzername (Secure Shell) *: sshuser
 - Verwenden Sie ein Clusteranmeldekennwort für SSH.
- Aktionen**:
 - Bewerten + erstellen
 - < Zurück
 - Weiter: Speicher »

D.3. HDInsight - Clustererstellung nicht möglich ohne Kontingenterhöhung

The screenshot shows the 'HDInsight-Cluster erstellen' (Create HDInsight Cluster) page in the Microsoft Azure portal. The 'Knotenkonfiguration' (Node Configuration) section displays a table of node types and their configurations. A red warning box highlights that there are not enough worker nodes (E8 V3) to support the selected number of worker nodes (4). It suggests increasing the number of worker nodes or selecting a different region.

Knotentyp	Knotengröße	Knotenzahl	Geschätzte Kosten...
Hauptknoten-K...	E8 V3 (8 Kerne, 64 GB RAM), 0.76 USD/Stunde	2	1.52 USD
Zookeeper-Knot...	A2 v2 (2 Kerne, 4 GB RAM), 0.12 USD/Stunde	3	0.00 (KOSTENLOS)
Worker-Knoten	E8 V3 (8 Kerne, 64 GB RAM), 0.76 USD/Stunde	4	3.04 USD

Aktivieren des verwalteten Datenträgers

Automatische Skalierung aktivieren [Weitere Informationen](#)

Geschätzte Gesamtkosten/Stunde 4.56 USD

Skriptaktionen
Verwenden Sie Skriptaktionen zum Ausführen benutzerdefinierter PowerShell- oder Bash-Skripts auf Clusterknoten während der Clusterbereitstellung. [Informationen zu Skriptaktionen](#)

[+ Skriptaktion hinzufügen](#)

D.4. Microsoft Azure - Kontingent für HDInsight

The screenshot shows the Microsoft Azure portal with the URL https://portal.azure.com/#view/Microsoft_Azure_Capacity/QuotaMenuBlade/-/myQuotas. The page title is "Kontingente - Microsoft". The main navigation bar includes "Home > Kontingente" and a search bar "Nach Ressourcen, Diensten und Dokumenten suchen (G+)". The left sidebar has sections for "Überblick", "Einstellungen", and "Meine Kontingente". The main content area displays a table with one row under "Keine Nutzung (1)". The table columns are "Kontingenztname", "Region", "Abonnement", and "Aktueller Verbrauch". The row details are "Cores", "Germany West Central", "Masterarbeit", and a progress bar at 0 %.

D.5. HDInsight - Erfolgreiche Clusterbereitstellung

The screenshot shows the Microsoft Azure portal with the URL <https://portal.azure.com/#view/HubsExtension/DeploymentDetailsBlade/-/overview/d/%2Fsubscriptions%2F7ac17151-5b03-417c-8423-523c7fba0e1>. The page title is "HDInsight_2022-09-15T18.08.17.320Z | Übersicht". The main content area shows a green checkmark icon and the message "Ihre Bereitstellung wurde abgeschlossen." Below this, it lists deployment details: "Bereitstellungsname: HDInsight_2022-09-15T18.08.17.320Z", "Startzeit: 15.9.2022, 20:08:19", "Abonnement: Masterarbeit", "Ressourcengruppe: jupyter_azure", and a "Korrelations-ID: c9cdc2e2-92cf-47e6-9c14-e446c57d0805". There are also sections for "Bereitstellungsdetails" and "Nächste Schritte". A blue button at the bottom says "Zu Ressource wechseln".

D.6. HDInsight Cluster - Speicherkonto

D.7. HDInsight Speicherkonto - Datei hochladen

D.8. HDInsight Speicherkonto - Dateiberechtigung anpassen

<input type="checkbox"/>	HdiSamples	6.11.2022, 07:44:42	Blockblob	0 B
<input type="checkbox"/>	hdp	6.11.2022, 07:33:42	Blockblob	0 B
<input type="checkbox"/>	hive	6.11.2022, 07:33:42	Blockblob	0 B
<input type="checkbox"/>	mapred	6.11.2022, 07:33:42	Blockblob	0 B
<input type="checkbox"/>	mr-history	6.11.2022, 07:33:42	Blockblob	0 B
<input type="checkbox"/>	tmp	6.11.2022, 07:33:42	Blockblob	0 B
<input type="checkbox"/>	user	6.11.2022, 07:33:42	Blockblob	0 B
<input type="checkbox"/>	warehouse	-	-	-
<input type="checkbox"/>	yaml	-	-	-
<input checked="" type="checkbox"/>	train_data.csv	6.11.2022, 10:59:26	Blockblob	15.27 GiB
<input type="checkbox"/>	user	6.11.2022, 07:33:42	Blockblob	0 B

D.9. HDInsight Speicherkonto - Dateilink kopieren

Eigenschaften

URL: <https://jupyterhdstorage...>

LETZTE ÄNDERUNG: 6.11.2022, 10:59:26 AM
ERSTELLUNGSZEIT: 6.11.2022, 10:59:26 AM
VERSIONS-ID: -
TYP: Blockblob
GROSSE: 15.27 GiB
ZUGRIFFESEBENE: N/V
LETZTE ÄNDERUNG DER ZUGRIFFESEBENE: N/V
ARCHIVSTATUS: -
AKTIVIERUNGSPRORITÄT: -
SERVER VERSCHLÜSSELT: true
ETAG: 0x8DABFDD968CC086
RICHTLINIE ZUR UNVERÄNDERLICHKEIT AUF VERSIONSEBENE: Deaktiviert
CACHESTEUERUNG:
INHALTSTYP: text/csv
CONTENT-MDS:
INHALTSCODIERUNG:
INHALTSSPRACHE:
INHALTSPOSITION:
LEASESTATUS: Entsperrt
LEASEZUSTAND: Verfügbar