http://dx.doi.org/10.25673/39552

Vergleichende IT-Sicherheitsanalyse aktueller Datenplattformen

Sebastian Tebbje, Karl-Heinz Niemann, Björn Nickel,

Fakultät 1
Hochschule Hannover
Ricklinger Stadtweg 120, 30459 Hannover
Sebastian.Tebbje@Hs-Hannover.de
Karl-Heinz.Niemann@Hs-Hannover.de
Björn.Nickel@stud.Hs-Hannover.de

Abstract: Big-Data-Datenplattformen werden immer beliebter, um große Datenmengen bei Bedarf analysieren zu können. Zu den fünf gängigsten Big-Data-Verarbeitungsframeworks gehören Apache Hadoop, Apache Storm, Apache Samza, Apache Spark, und Apache Flink. Zwar unterstützen alle fünf Plattformen die Verarbeitung großer Datenmengen, doch unterscheiden sich diese Frameworks in ihren Anwendungsbereichen und der zugrunde liegenden Architektur. Eine Reihe von Studien hat sich bereits mit dem Vergleich dieser Big-Data-Frameworks befasst, indem sie sie anhand eines bestimmten Leistungsindikators bewertet haben. Die IT-Sicherheit dieser Frameworks wurde dabei jedoch nicht betrachtet. In diesem Beitrag werden zunächst allgemeine Anforderungen und Anforderungen an die IT-Sicherheit der Datenplattformen definiert. Anschließend werden die Datenplattform-Konzepte unter Berücksichtigung der aufgestellten Anforderungen analysiert und gegenübergestellt.

1 Einleitung

Seit dem Aufkommen der digitalen Transformation in fast allen Bereichen sind Daten zu einem der wichtigsten Aktivposten geworden, wenn es darum geht, Wettbewerbsvorteile für Unternehmen zu erlangen. Mit dem technologischen Fortschritt ist die Geschwindigkeit der Datengenerierung weltweit exponentiell angestiegen. Besonders in der Industrie fallen täglich riesige Mengen an heterogenen Daten und Informationen an, die die Verarbeitungskapazität herkömmlicher Datenbanksysteme übersteigen. Um einen Nutzen aus diesen Daten zu ziehen, muss ein alternativer Weg für ihre Verarbeitung gewählt werden [1]. In den letzten Jahren konnten sich Datenplattformen unter dem Begriffen Big Data und Cloud-Technologien am Markt etablieren, um diese Datenflut über verschiedene Unternehmensbereiche zugänglich zu machen. Mit dem Trend zu IoT und Industrie 4.0 müssen diese Daten zukünftig nicht nur innerhalb eines Unternehmens, sondern auch über dessen Unternehmensgrenzen hinweg innerhalb des Wertschöpfungsnetzwerks bereitgestellt werden können. Wertschöpfungsnetzwerke sind eine Organisationsform bestehend aus rechtlich selbständigen und wirtschaftlich operierenden Unternehmen, die über Geschäftsbeziehungen miteinander verbunden sind. Die kooperative Zusammenarbeit ermöglicht es diesen Unternehmen, ihre Planungs-, Produkt- und Prozessdaten durch alle Stufen der Wertschöpfungskette miteinander zu verknüpfen. Im Detail bedeutet dies, durch die technische Anbindung diverser Systeme, Komponenten oder Sensoren mittels passender Hard- und Software, ein gezieltes Sammeln tatsächlich notwendiger Daten zu ermöglichen. Weiterführend besteht die Möglichkeit diese Daten mit eingebundenen Wissensdatenbanken und unter Einbeziehung von KI auszuwerten. Danach werden diese Daten visualisiert und analysiert, in Benachrichtigungen und Aktionen umgesetzt, um gegebenenfalls direkt notwendige Prozesse anstoßen zu können. Beispielweise das Auslösen eines Wartungs- oder Serviceeinsatzes oder einer Teilelieferung. Hersteller, Anwender und weitere Unternehmenspartner können auf derselben Plattform zusammenarbeiten, Daten teilen und ggf. auch direkte Fernzugriffe auf Anlagendaten vornehmen. Die Daten, die von Maschinen im Verlauf der Produktion oder Wartung erzeugt werden, bergen großes Potenzial. Zum einen, um die Effizienz des Produktionsprozesses zu steigern, zum anderen, um die Daten mit Dritten zu tauschen, woraus im Idealfall neue Services oder Produkte entstehen können. Mit der zunehmenden Öffnung der Unternehmensnetzwerke wachsen, neben den Potentialen, auch die Herausforderungen und Risiken in Bezug auf die IT-Sicherheit und die rechtlichen Grundlagen, beispielsweise bei unrechtmäßiger Datennutzung. Die Schwachstellen in der OT-Sicherheit steigen jährlich exponentiell an [2]. Je umfangreicher die erfasste Datenmenge ist, desto höher ist die Wahrscheinlichkeit, dass auch sensible Informationen enthalten sind. Das Offenlegen vertraulicher Informationen könnte dem Unternehmen schaden, weshalb sich die Unternehmen darauf konzentrieren müssen, die Integrität und den Schutz der gesammelten Daten zu wahren. Der Schutz von Daten geht über die Beschränkung des Zugriffs auf bestimmte Ressourcen (Identitäts- und Zugriffskontrolle) hinaus: Es muss auch kontrolliert werden, wie die Daten nach dem Zugriff behandelt werden, was als Data Usage Control [3] bezeichnet wird. Data Usage Control bietet einen gemeinsamen und vertrauenswürdigen Sicherheitsrahmen, um die Einhaltung von Data-Governance-Regeln

und die verantwortungsvolle Nutzung von Unternehmensdaten durch Drittparteien zu gewährleisten und die sichere gemeinsame Nutzung von Daten in Ökosystemen wie Industrie 4.0 zu erleichtern und sicherzustellen. Dies wird über entsprechende Verträge zwischen den Unternehmen realisiert, was nicht Bestandteil dieses Beitrags ist. Aufgrund der Vielzahl von verfügbaren Datenplattformlösungen bedarf es einer Übersicht, die die Funktionsweisen der unterschiedlichen Ansätze gegenüberstellt. Weiterhin müssen die Plattformkandidaten klassifiziert und differenziert werden, da die Ansätze auf unterschiedliche Anwendungsfälle zielen. Des Weiteren sollen die IT-Sicherheitseigenschaften der verschiedenen Implementierungen anhand von zuvor definierten Anforderungen evaluiert werden. In diesem Beitrag werden daher die für Industrie 4.0 relevanten Datenplattformkandidaten bezüglich der allgemeinen- und IT-Sicherheitseigenschaften differenziert und vergleichend gegenübergestellt werden.

2 Forschungsstand

Es gibt bereits eine Reihe von Veröffentlichungen, in denen etablierte Frameworks und Tools zur Erstellung und zum Betrieb von Datenplattformen verglichen und evaluiert werden. Jedoch liegt das Hauptaugenmerk dieser Vergleiche entweder auf einer sehr geringen Auswahl von Frameworks oder es werden ausschließlich die funktionellen Anforderungen an eine Datenplattform berücksichtigt. So werden in [4] zwei Frameworks der Apache Foundation auf ihre Streaming-Performance untersucht. In [5] erfolgt die Untersuchung etwas allgemeiner, es werden vier Frameworks der Apache Foundation anhand von zuvor definierten „Key Performance Indicators“ (KPI) miteinander verglichen. Während [6] primär auf die Skalierbarkeit zweier vergleichbarer Frameworks abzielt. In [7] werden verschiedene Big-Data-Tools auf ihre „graph processing“-Fähigkeiten evaluiert. Außerdem liefert [8] einen Vergleich einer kommerziellen Hadoop Distribution in Kombination mit den Cloud-Service-Provider Amazon Web Services (AWS) und Microsoft Azure. Bei all diesen Veröffentlichungen fällt auf, dass die IT-Sicherheitsanforderungen an eine Datenplattform im Allgemeinen nicht berücksichtigt werden. Dieser Aspekt soll im Weiteren näher betrachtet werden.

3 Klassifizierung von Datenplattformen

Eine Datenplattform stellt eine Komplettlösung für die durchgängige Datenverarbeitung dar. Auf der Plattform, werden Daten aus verschiedenen Datenquellen erfasst, verarbeitet, kontrolliert und für Benutzer oder Datenanwendungen für weiterführende Analysen bereitgestellt. Datenplattformen werden in der Regel für die Verarbeitung sehr großer Datenmengen unterschiedlichen Ursprungs eingesetzt. Eine Datenplattform berücksichtigt möglicherweise jeden einzelnen Datensatz, den ein Unternehmen im Rahmen seiner Geschäftstätigkeit generiert. Dazu zählen sowohl IT-Daten aus dem Bürosektor, als auch Maschinendaten aus der Feldebene. In diesem Beitrag wird zwischen drei verschiedenen Klassen von Datenplattformen unterschieden. Die Klassen werden durch die Verarbeitungsweise (engl. processing) der Daten innerhalb der Plattformen definiert [9].

1) Stapelverarbeitung (batch-only processing):

Bei der Stapelverarbeitung werden Daten zunächst über einen bestimmten Zeitraum gespeichert und blockweise gruppiert. Anschließend werden die Datenblöcke gemeinsam verarbeitet.

2) Streamverarbeitung (stream-only processing):

Bei der Streamverarbeitung werden Daten kontinuierlich erfasst, verteilt und analysiert.

3) Hybridverarbeitung (hybrid processing):

Bei der Hybridverarbeitung kann sowohl Stream-, als auch Stapelverarbeitung betrieben werden.

Der beim Data Warehousing verwendete Schritt des Extrahierens, Transformierens und Ladens (ETL) ist in der Regel ein Batch-Prozess. Zu den wichtigsten Vorteilen der Stapelverarbeitung gehört die Möglichkeit, die Verarbeitung von Aufträgen auf andere Ressourcen mit größerer Kapazität zu verteilen, sowie Computerressourcen unter Programmen und Benutzern aufzuteilen. Während bei der Stapelverarbeitung Daten(-gruppen) verarbeitet werden, die über einen bestimmten Zeitraum gespeichert wurden, und die zu regelmäßig geplanten Zeiten oder nach Bedarf ausgeführt werden, kann der Benutzer bei der Stream-Verarbeitung Daten in Analysetools einspeisen, sobald sie generiert wurden. Dies ermöglicht eine Datenverarbeitung quasi in Echtzeit und sofortige Analyseergebnisse. Zu den Vorteilen gehört die sofortige Erkennung von Bedingungen und Anomalien innerhalb eines sehr kurzen Zeitraums. Die Streamverarbeitung

ist demnach bei Anwendungsfällen, die sehr geringe Verarbeitungslatenzen erfordern, von Vorteil, während die Stapelverarbeitung auf Anwendungsfälle mit sehr großen Datenmengen zielt.

4 Anforderungen an die Datenplattformen

Bei der Wahl eines Frameworks für eine Datenplattform im Kontext von Industrie 4.0 spielen diverse Faktoren eine Rolle. In diesem Abschnitt werden einige grundlegende Anforderungen an ein Framework definiert. In Abschnitt 4.1 werden generelle und in 4.2 sicherheitsspezifische Anforderungen beschrieben.

4.1 Generelle Anforderungen an eine Datenplattform

1) Betriebsmodell:

Beim Betriebsmodell wird zwischen On- und Off-Premise unterschieden. On-Premise stellt den klassischen Weg zur Realisierung einer Datenplattform dar. Dabei erfolgt die Installation auf vorhandener Hardware lokal innerhalb eines Unternehmens. Während Off-Premise die Bereitstellung einer Datenplattform an anderer Stelle, z. B. bei einem Cloud-Anbieter, bedeutet. Möglich ist auch ein hybrider Betrieb der Datenplattform, bei dem vorhandenen unternehmensinterne Ressourcen mit einer oder mehreren Clouds kombiniert werden können. Abhängig vom Betriebsmodell ergeben sich für entsprechende Anwendungsfälle verschiedene Vor- und Nachteile für den Betreiber der Datenplattform. Zu den wichtigsten Fragestellungen dabei gehören die Kosten-Nutzen-Effizienz und die Datenhoheit [10]. Um allen Anwendungsfällen gerecht zu werden, sollte eine Datenplattform in allen Betriebsmodellen betrieben werden können, oder über entsprechende Verbindungsmöglichkeiten verfügen.

2) Skalierbarkeit:

Skalierbarkeit ist die Fähigkeit eines Systems, auf eine zunehmende Last zu reagieren. Es gibt zwei Arten: Skalierung nach oben (vertikal) und Skalierung nach außen (horizontal). Bei der vertikalen Skalierung wird die Hardwarekonfiguration in Bezug auf ihre Leistungsfähigkeit aufgerüstet, während bei der horizontalen Skalierung zusätzliche Hardware-Einheiten hinzugefügt wird. Eine Datenplattform muss in der Lage sein, die Volumenanforderungen des jeweiligen Anwendungsfalls zu erfüllen und sich an das Wachstum der zu bearbeitenden Datenmenge anzupassen.

3) Maschinelles Lernen und Künstliche Intelligenz

Technologische Innovationen, insbesondere in den Bereichen Machine Learning (ML) und künstliche Intelligenz (KI), haben neue Möglichkeiten für Unternehmen jeder Größe geschaffen, von datengestützten Erkenntnissen zu profitieren. Die profitable Nutzung der anfallenden Daten liefert einen wesentlichen Grund für die Erstellung einer Datenplattform, daher muss die Datenplattform über entsprechende Integrationsmöglichkeiten für skalierbare Software-Bibliotheken verfügen, um Algorithmen des maschinellen Lernens ausführen zu können.

4) Echtzeitfähigkeit:

Echtzeitfähigkeit ist für Anwendungen mit zeitkritischen Übertragungsfenstern relevant. In diesem Beitrag wird daher zwischen harter, weicher und keiner Echtzeitfähigkeit unterschieden. Harte Echtzeitfähigkeit bedeutet, dass Deadlines immer eingehalten werden. Bei weicher Echtzeitfähigkeit liegt die durchschnittliche Übertagungszeit innerhalb der Deadline aber vereinzelte Überschreitungen sind tolerabel. Bei keiner Echtzeitfähigkeit sind keinerlei Garantien für die rechtzeitige Verarbeitung der Information gegeben.

5) Resilienz:

Resilienz bezeichnet die Fähigkeit von IT-Systemen, hinsichtlich Störungen und Problemen wie Ausfällen einzelner Komponenten robust zu reagieren und die Anwender weiterhin mit den benötigten Services zu versorgen. Typische Maßnahmen zur Sicherstellung der Resilienz sind redundante, verteilte Systeme und Datensicherungen. Eine Datenplattform bildet die Grundlage für die bereitgestellten Dienste und sollte entsprechend resilient sein. Gleichzeitig wird mit dieser Anforderung auch die Verfügbarkeit adressiert, welches eins der primären Schutzziele der IT-Sicherheit darstellt.

6) Verfügbare Implementierung:

Die Verfügbarkeit von Open-Source und/oder kommerzieller Implementierungen ist eine grundsätzliche Voraussetzung bei der Auswahl geeigneter Lösungen, um den Betreibern anwendungsfallspezifische Konfigurations- und Gestaltungsspielraum zu ermöglichen.

7) Nachrichtenzustellungsgarantie

Garantien für die Zustellung von Nachrichten werden im Falle eines Fehlers benötigt. Die Datensenke bestätigt der Datenquelle den Empfang der Nachricht. Es lassen sich zwei Arten von Garantien unterscheiden: genau einmalige Zustellung und mindestens einmalige Zustellung. Exakte einmalige Zustellung bedeutet, dass die Nachricht weder dupliziert wird noch verloren geht und dem Empfänger genau einmal zugestellt wird. Mindestens eine Zustellung bedeutet, dass mehrere Versuche unternommen werden, die Nachricht zuzustellen, und dass mindestens einer dieser Versuche erfolgreich war. Darüber hinaus kann die Nachricht dupliziert werden, ohne dass sie verloren geht.

8) Datenspeicher und Berechnungsmodus (engl. computation mode)

Der Berechnungsmodus der Datenbank kann in-memory oder disk-based sein. Eine In-Memory-Datenbank speichert alle Daten im Hauptspeicher (RAM) eines Computers. Eine herkömmliche Datenbank ruft die Daten von Festplattenlaufwerken ab. In-Memory-Computing ist schneller, hat aber den Nachteil, dass der Inhalt verloren geht, wenn der Rechner ausfällt. Hierbei handelt es sich um eine nicht funktionale Anforderung.

4.2 Anforderungen an die IT-Sicherheit der Datenplattform

Das grundsätzliche Ziel der IT-Sicherheit ist es, einen Schutz aller Informationen und Prozesse in einem informationstechnischen System zu gewährleisten. Die allgemeinen übergeordneten Schutzziele der IT-Sicherheit sind: Verfügbarkeit, Integrität, Vertraulichkeit, Authentizität und Nichtabstreitbarkeit [11]. In [12] wird eine Übersicht über Normen und Standards zur IT-Sicherheit gegeben und auf dieser Grundlage verschiedene Anforderungen an die IT-Sicherheit definiert. Während unter 4.1 bereits Anforderungen bezüglich der Verfügbarkeit (Resilienz, Nachrichtenzustellungsgarantie) erläutert wurden, werden im Folgenden weitere technische Anforderungen an die IT-Sicherheit einer Datenplattform unter Berücksichtigung der IT-Schutzziele vorgestellt.

1) Authentifizierung

Die Authentizität gewährleistet, dass ein Kommunikationspartner tatsächlich derjenige ist, der er vorgibt zu sein. Besonders bei verteilten Systemen stellt dies eine Herausforderung dar. Eine Datenplattform muss die Fähigkeit haben, alle Nutzer im Netzwerk eindeutig identifizieren und authentifizieren zu können.

2) Autorisierung

Neben der Authentifizierung eines Nutzers, muss es auch eine Möglichkeit zur Autorisierung geben. Eine restriktive Berechtigungsvergabe kann beispielsweise durch Verfahren wie Role Based Access Control (RBAC) oder Discretionary Access Control (DAC) erfolgen.

3) Schutz der Daten vor unautorisiertem Zugriff

Daten können sowohl bei der Übertragung (data-in-transit) als auch im Ruhezustand (data-at-rest) Risiken ausgesetzt sein und müssen in beiden Zuständen geschützt werden. Es gibt verschiedene Verschlüsselungsmethoden um sensible Daten im Sinne der Vertraulichkeit zu schützen. Dabei werden die Daten vor der Übertragung verschlüsselt oder es werden verschlüsselte Verbindungen (wie z.B. HTTPS, FTPS usw.) verwendet. Für den Schutz von Daten im Ruhezustand können die Daten vor der Speicherung oder das Speicherlaufwerk verschlüsselt werden.

4) Audit

Um die Nichtabstreitbarkeit zu gewährleisten, muss die Datenplattform über entsprechende Protokollierungsfunktionen verfügen. Es müssen alle Aktivitäten innerhalb der Datenplattform zu jeder Zeit nachvollziehbar sein. Mit Hilfe von Audits kann sichergestellt werden, dass keine audit-relevanten Ereignisse aufgetreten sind.

5 Übersicht der Datenplattformen

Nachfolgend werden fünf Datenplattformen im Hinblick auf die generellen Anforderungen aus Abschnitt 4.1 beschrieben. Die Auswahl beruht auf den zur Zeit am häufigsten genutzten Frameworks [13]. Alle genannten Kandidaten sind über freie Open-Source-Implementierungen verfügbar und horizontal skalierbar. Ursprünglich für den On-Premise-Betrieb entworfen, sind alle Frameworks auch in der Cloud betreibbar und es bestehen entsprechende Migrationsmöglichkeiten. Die Kandidaten sind den in Abschnitt 3 definierten Klassen zugeordnet.

A) Stapelverarbeitung (batch-only processing):

A.1) Apache Hadoop [14]

Apache Hadoop wurde 2008 als Batch-Processing-Framework definiert, das verteilte Daten über eine Gruppe von Host-Rechnern, die Cluster oder Knoten genannt werden, sammelt und verarbeitet. Hadoop besteht aus drei Kernkomponenten. Die Komponente zum Sammeln von Daten nennt sich Hadoop Distributed File System (HDFS). Die Speicherung der Daten erfolgt dabei auf der Festplatte. HDFS erstellt Replikate den Hosts innerhalb eines Clusters und ist dadurch fehlertolerant. Wenn also ein Rechner im Cluster ausfällt, kann auf die Daten von anderen Rechnern zugegriffen werden, auf denen die gleiche Kopie der Daten erstellt wurde. Die zweite Komponente ist YARN, welche arithmetische Ressourcen für die Auftragsplanung wie CPU und Speicher bereitstellt. Die dritte Komponente ist MapReduce, welches für die Verarbeitung von Daten (Softwareschicht) mit anderen Prozessen verwendet wird. Über die Jahre hat sich ein komplettes Ökosystem mit zahlreichen Erweiterungsmöglichkeiten um Hadoop herum gebildet, darunter auch Apache Mahout. Mahout ist ein Framework zur Erstellung skalierbarer, leistungsstarker Anwendungen für maschinelles Lernen.

B) Streamverarbeitung (stream-only processing):

B.1) Apache Storm [15]

Apache Storm ist ein Framework für die Verarbeitung großer strukturierter und unstrukturierter Daten in Echtzeit. Es ist ausschließlich auf die Stream-Verarbeitung spezialisiert. Storm besitzt zwei Schnittstellen: die normale Storm API und Storm Trident. Erstere unterstützt keine höheren Abstraktionen, sondern lehnt seine API an das reine Stream Processing Model an. Die API bearbeitet jedes Tupel einzeln und nutzt die At-Least-Once-Semantik. Storm Trident bietet eine höhere Abstraktionsebene an und arbeitet mit Micro-Batches. Es stellt eine Exactly-Once-Semantik bereit. Die Streamverarbeitung wird direkt auf den Hosts durchgeführt. Storm überwacht dabei die Datenverarbeitung und wiederholt die Verarbeitung bei Fehlern. Die Verwaltung des Clusters und der Ressourcen erfolgen durch Nimbus. Storm kann im Standalone-Modus oder auf YARN betrieben werden. Anwendungen werden über den Nimbus bereitgestellt und per Thrift, einem Datenaustauschformat, zu den Hosts übertragen. Storm verfügt über eine SAMOA API. Apache SAMOA bietet eine Sammlung verteilter Streaming-Algorithmen für die gängigsten Data-Mining- und Machine-Learning-Aufgaben wie Klassifizierung, Clustering und Regression sowie Programmierabstraktionen zur Entwicklung neuer Algorithmen, die auf verteilten Stream-Processing-Engines (DSPEs) laufen.

B.1) Apache Samza [16]

Apache Samza wurde ursprünglich von LinkedIn entwickelt, um verschiedene Arten von Stream-Processing-Anforderungen zu erfüllen, wie z. B. die Verfolgung von Daten, die Protokollierung von Daten durch Dienste und Dateneingabepipelines für Echtzeitdienste. Samza ist wie Storm ausschließlich auf die Stream-Verarbeitung ausgelegt. Im Gegensatz zu den anderen Frameworks hat es kein eigenes Transport- und Verarbeitungssystem. Es nutzt Apache Kafka als Transportsystem und Apache Hadoop YARN als Verarbeitungssystem. Es verwendet eine In-Memory-Verarbeitungstechnik und bietet die Samza API zur Anwendungsentwicklung und die SAMOA API. Kafka sorgt zusammen mit ZooKeeper für die nötige Fehlertoleranz, indem der Verarbeitungsstand dort gesichert wird. Außerdem wird garantiert, dass die Daten immer mindestens einmal gesendet werden.

C) Hybridverarbeitung (hybrid processing):

C.1) Apache Spark [17]

Apache Spark ging aus einem Forschungsprojekt der University of California, Berkeley hervor. Es war ursprünglich als Ersatz für das Apache Hadoop MapReduce Framework konzipiert. Spark nutzt eine Abstraktion namens Resilient Distributed Dataset (RDD). RDDs sind schreibgeschützte Daten-Partitionen, die auf den Rechenknoten vorliegen. Sie können im Fehlerfall einfach aus den Ausgangsdaten wiederhergestellt werden. Dieses Batch-orientierte Konzept wurde später für Streaming-Daten angepasst. Es nutzt kein natives Streaming, sondern Micro-Batching, das heißt, es sammelt eingehende Daten für einen gewissen Zeitraum und

verarbeitet diese in bestimmten Zeitabständen. Bei jedem Verarbeitungsschritt werden die Operatoren neu auf die Hosts verteilt. Dadurch wird weiche Echtzeit ermöglicht. Es stehen eine umfangreiches API und einige Bibliotheken bereit, darunter eine Machine-Learning-Bibliothek (MLlib). Außerdem kann, genau wie Hadoop, Apache Mahout genutzt werden. Spark bietet mehrere Schnittstellen zum Erstellen von Anwendungen an. Für die Streamverarbeitung ist nur die DStream API relevant. Beim Datentransport werden Checkpoints genutzt, um Fehlertoleranz herzustellen. Als Ablage für Checkpoints wird das verteilte Dateisystem HDFS genutzt. Ein Master verwaltet die Ressourcen und ist beim Starten der Anwendung beteiligt. Mithilfe von ZooKeeper können redundante Master eingesetzt werden, um eine höhere Ausfallsicherheit zu erreichen. Die Clusterverwaltung kann entweder direkt durch Spark im Standalone-Betrieb erfolgen oder durch ein externes Werkzeug wie Hadoop YARN.

C.2) Apache Flink [18]

Apache Flink ist ein Open-Source-Framework, das 2010 entwickelt wurde und sich für die Datenverarbeitung sowohl im Echtzeit- als auch im Batch-Modus bewährt hat. Es verwendet eine In-Memory-Verarbeitungstechnik und bietet eine Reihe von APIs wie Stream Processing API (Datenstrom), Batch-Processing API (Datensatz) und Table API, die für Abfragen verwendet werden. Es verfügt auch über Bibliotheken für maschinelles Lernen (FlinkML) und Graphverarbeitung (Gelly). Flink führt Fehlertoleranzmaßnahmen wie periodisches Sichern von Zwischenständen (Checkpointing) durch. Außerdem wird garantiert, dass die Daten genau einmal gesendet werden. Für die Clusterverwaltung nutzt Flink entweder den JobManager im Standalone-Betrieb, also ohne ein externes Werkzeug, oder beispielsweise Apache Hadoop YARN. Das verteilte Dateisystem HDFS dient der sicheren Ablage von Checkpoints. Der lokale Betriebsmodus erleichtert das Debuggen, da alle Komponenten lokal ausgeführt werden. Zum Starten, Beenden und Verwalten der Anwendungen kann eine CLI oder eine Web-GUI verwendet werden.

6 Sicherheitsanalyse der betrachteten Datenplattformen

1) Authentifizierung

Während Hadoop, Storm, Samza und Flink über keine integrierte Authentifizierungsmechanismen verfügen, bietet Spark die Authentifizierung für Remote Procedure Calls (RPC) und Web UI´s (Servlet Filter) an. Trotzdem ist eine Authentifizierung der Netzwerkteilnehmer in allen Frameworks möglich. Da alle Frameworks über TCP/IP kommunizieren, kann auf der Netzwerkschicht über Transport Layer Security (TLS) eine Authentifizierung erfolgen. Außerdem sind alle Frameworks mit Kerberos kompatibel. Kerberos ist ein TCP/IP-basiertes Authentifizierungssystem das speziell für Authentifizierung auf der Darstellungsschicht in einer verteilten Umgebung konzipiert ist. Durch entsprechende Erweiterungen der Frameworks können weitere Authentifizerungsmöglichkeiten genutzt werden: Für Storm gibt es Plugins für Apache Thrift und SASL, bei Samza kann die Authentifizierung über Kafka erfolgen und Kubernetes liefert Authentifikationsoptionen bei der Verwendung mit Spark. Flink kann theoretisch Erst- oder Drittanbieter-Konnektoren (Kafka, HDFS, Cassandra, Flume, Kinesis usw.) verwenden, die beliebige Authentifizierungsmethoden (Kerberos, SSL/TLS, Benutzername/Passwort usw.) erfordern. Praktisch wird jedoch nur Kerberos für die Authentifizierung sicher unterstützt.

2) Autorisierung

Flink bietet derzeit keinerlei Autorisierungsmöglichkeiten. Hadoop, Samza und Storm nutzen YARN als Ressourcenmanager, welcher umfangreiche Access Control Lists (ACLs) zur Autorisierung unterstützt. ACLs können entweder für Benutzer oder für Gruppen konfiguriert werden. Für Storm kann außerdem das Autorisierungs-Plugin SimpleACLAuthorizer genutzt werden. Die Berechtigungen können dabei an den Benutzernamen oder an das Kerberos-Ticket des Nutzers gebunden werden. Auf diese Weise kann jeder Benutzer mit gültigem Kerberos-Ticket Operationen durchführen, wie z.B. aktivieren, deaktivieren oder auf Clusterinformationen zugreifen. Spark unterstützt die Zugriffskontrolle auf die Web UI, sofern ein Servlet Filter vorhanden ist. Dadurch kann jede Anwendung mit ihren eigenen separaten ACLs konfiguriert werden. Spark unterscheidet dabei jedoch ausschließlich zwischen "Anzeige"-Berechtigungen und "Änderungs"-Berechtigungen. Sofern Spark YARN als Ressourcenmanager nutzt, kann auch darüber die Berechtigungsvergabe zurückgegriffen werden. Bei Hadoop und Storm gibt es die Möglichkeit über Apache Ranger oder Apache Sentry auch rollenbasierte Zugriffsberechtigungen (RBAC) zu erteilen. Sowohl Ranger als auch Sentry bilden ein Framework für die Definition von Richtlinien zur Kontrolle des Zugriffs auf Dateien, Ordner, Datenbanken, Tabellen oder Spalten.

3) Schutz der Daten vor unautorisiertem Zugriff

Da alle Frameworks die Möglichkeit bieten TLS zu nutzen, kann auf der Netzwerkschicht ebenfalls die Ende-zu-Ende Verschlüsselung bei der Datenübertragung durch TLS erfolgen. Oberhalb der Transportschicht implementiert Hadoop´s HDFS eine transparente Ende-zu-Ende-Verschlüsselung. Einmal konfiguriert, werden Daten, die aus speziellen HDFS-Verzeichnissen gelesen und in diese geschrieben werden, transparent ver- und entschlüsselt, ohne dass Änderungen am Code der Benutzeranwendung erforderlich sind. Diese Verschlüsselung erfolgt ebenfalls Ende-zu-Ende, was bedeutet, dass die Daten nur vom Client ver- und entschlüsselt werden können. Außerdem speichert HDFS niemals unverschlüsselte Daten. Apache Storm und Samza verfügen über keine eigenen Verschlüsselungsmethoden. Apache Spark bietet nur die Verschlüsselung von Steuerungsverbindungen (RPC). Zusätzlich unterstützt Spark die Verschlüsselung von temporären Daten, die auf lokale Festplatten geschrieben werden. Für Apache Flink bietet sich ausschließlich TLS an, da es keine eigenen Verschlüsselungsmethoden besitzt.

4) Auditing

Hadoop, Storm und Spark nutzen Apache Log4j als Logging-Framework. Darüber lassen sich sowohl Benutzeraktivitäten als auch Dienstaktivitäten protokollieren. Samza und Flink verwenden die SLF4J-Logging-Schnittstelle. Dadurch kann jedes Logging-Framework, das SLF4J unterstützt, verwendet werden. Standardmäßig wird Log4j 2 als das zugrundeliegende Logging-Framework verwendet. Auf die Log-Dateien kann über die Job-/TaskManager-Seiten der WebUI zugegriffen werden. Der verwendete Resource Provider (z.B. YARN) kann zusätzliche Zugriffsmöglichkeiten bieten.

7 Zusammenfassung und Fazit

In den folgenden Tabellen 1 und 2 sind die Ergebnisse der Evaluierung anhand der aufgestellten Anforderungen zusammenfassend dargestellt.

	Hadoop	**Storm**	**Samza**	**Spark**	**Flink**
Datenverarbeitungsweise	Batch	Stream	Stream	Hybrid	Hybrid
Betriebsmodell	On-/Off-Prem	On-/Off-Prem	On-/Off-Prem	On-/Off-Prem	On-/Off-Prem
Skalierbarkeit	Horizontal	Horizontal	Horizontal	Horizontal	Horizontal
Machine Learning	Mahout	SAMOA API	SAMOA API	SparkMLlib, Mahout	FlinkML
Echtzeit	✗	✓	✓	Weiche Echtzeit	✓
Resilienz	✓	✓	✓	✓	✓
Open Source	✓	✓	✓	✓	✓
Nachrichtenzustellungsgarantie	Exakt eine	Mindestens eine, exakt eine (Trident API)	Mindestens eine	Exakt eine	Exakt eine
Datenspeicher	Disk-based	In memory	In memory	In memory	In memory

Tabelle 1: Übersicht der evaluierten Datenplattformkandidaten

	Hadoop	Storm	Samza	Spark	Flink
Authentifizierungs-Möglichkeiten	Kerberos, TLS	Kerberos, Thrift, SASL, TLS	Kerberos (YARN), SSL+SASL (Kafka), TLS	Kerberos (YARN), TLS, Servlet Filter (WEB UI +RPC)	Kerberos, TLS
Autorisierungs-Möglichkeiten	ACL, RBAC (Sentry, Ranger)	ACL, SimpleACLAuthorizer RBAC (Ranger)	ACL (YARN)	Servlet Filter, ACL (YARN)	✗
Verschlüsselung (in-transit)	HDFS, TLS	(HDFS), TLS	TLS	AES (RPC), TLS	(HDFS), TLS
Verschlüsselung (at-rest)	✓	✗	✗	✓	✗
Auditing	✓	✓	✓	✓	✓

Tabelle 2: Übersicht der IT-Sicherheitsfeatures

In diesem Beitrag wurden die Datenplattformkandidaten Hadoop, Storm, Samza, Spark und Flink anhand verschiedener Anforderungen evaluiert. Die Anforderungen adressieren die allgemeine Funktionalität einer Datenplattform und die vorhanden IT-Sicherheitsfeatures für den sicheren Betrieb. Die Definitionen der Anforderungen beruhen auf kontextspezifische Standards und Normen. Abschließend wurden die Ergebnisse für die einzelnen Kriterien ausgeführt und in Tabellen zusammengefasst dargestellt.

Tabelle 1 zeigt, dass alle Plattformkandidaten die funktionalen Anforderungen weitestgehend erfüllen. Ein Unterschied besteht im Wesentlichen auf der Verarbeitung in Echtzeit, was wiederum vom übergeordneten Verarbeitungsmodus abhängt. Jede dieser Datenplattformen bietet demnach verschiedene Vor- und Nachteile, sodass eine konkrete Auswahl von dem jeweiligen Anwendungsfall abhängig gemacht werden muss. Um möglichst viele Anwendungsfälle abdecken zu können, empfiehlt sich eine Kombination verschiedener Frameworks. In Tabelle 2 werden die Ergebnisse der IT-Sicherheitsanalyse dargestellt. Wobei die Analyse auf den jeweiligen Dokumentationen beruht. Anhand der Spezifikationen erfüllt keine der Datenplattformen alle Anforderungen. Beispielsweise muss bei der Authentifizierung Kerberos oder TLS genutzt werden. Als Autorisierungsmöglichkeiten können lediglich ACLs und Filter genutzt werden, was eine feingranulare restriktive Berechtigungsvergabe im Kontext von Industrie 4.0 erschwert. Hadoop und Strom bieten die Möglichkeit durch Apache Ranger RBAC zu ermöglichen. Daher ist eine finale Auswahl für eine präferierte Plattform nicht möglich. Vielmehr muss man die einzelnen Features bewerten und dann an Hand der Detail-Information eine Auswahl treffen.

8 Literaturverzeichnis

[1] Dumbill, E. 2013. Making Sense of Big Data. *Big data* 1, vol. 1 no. 1, 1–2.

[2] Claroty. 2021. *Claroty Biannual ICS Risk & Vulnerability*. *REPORT: 1H 2021*. https://security.claroty.com/1H-vulnerability-report-2021. Accessed 20 September 2021.

[3] Kelbert, F. and Pretschner, A. 2018. *Data Usage Control for Distributed Systems*.

[4] Samadi, Y., Zbakh, M., and Tadonki, C. 2018. Performance comparison between Hadoop and Spark frameworks using HiBench benchmarks. *Concurrency Computat Pract Exper* 30, 12.

[5] Safaa Alkatheri, Samah Abbas, and Muazzam Ahmed Siddiqui. 2019. A Comparative Study of Big Data Frameworks. *International Journal of Computer Science and Information Security (IJCSIS)* 17, 1.

[6] García-Gil, D., Ramírez-Gallego, S., García, S., and Herrera, F. 2017. A comparison on scalability for batch big data processing on Apache Spark and Apache Flink. *Big Data Anal* 2, 1.

[7] Kaepke, M. and Zukunft, O. A Comparative Evaluation of Big Data Frameworks for Graph Processing, 30–37.
[8] Ahmed Kawser and Kawser Ahmed Pinto. 2020. A Comparative study one of the Hadoop distribution Hortonworks with Amazon Web Service (AWS) and Microsoft Azure (2020).
[9] Bhupender singh thakur and Dr. Kishori Lal Bansal. 2020. Comparative Analysis and Evaluation of Big Data Processing Frameworks and Tools. *Mukt Shabd Journal,* 9, 1338–1348.
[10] Bundesministerium für Wirtschaft und Energie (BMWi). 2019. Sichere unternehmensübergreifende Kommunikation mit OPC UA. Diskussionspapier.
[11] Bundesamt für Sicherheit in der Informationstechnik. 2016. *IT-Grundschutz-Katalog*. https://www.bsi.bund.de/DE/Themen/ITGrundschutz/ITGrundschutzDownloads/Download_Archiv/download_node.html.
[12] Tebbje, S. and Niemann, K.-H. 2020. IT-Security-Anforderungen bei firmenübergreifenden Wertschöpfungsketten im Kontext von Industrie 4.0. *Anwendungsorientierte Forschung für die digitale Transformation von KMU - Schriften des Forschungsclusters Industrie 4.0*, 79–93.
[13] Inoubli, W., Aridhi, S., Mezni, H., Maddouri, M., and Mephu Nguifo, E. 2018. An experimental survey on big data frameworks. *Future Generation Computer Systems* 86, 546–564.
[14] Apache Software Foundation. 2005. *Apache Hadoop. Open-source software for reliable, scalabe and distributed computing*. https://hadoop.apache.org/.
[15] Apache Software Foundation. 2011. *Apache Storm. Distributed and fault-tolerant realtime computation*. https://storm.apache.org/.
[16] Apache Software Foundation. 2018. *Apache Samza. A distributed stream processing framework*. http://samza.apache.org/.
[17] Apache Software Foundation. 2014. *Apache Spark. Lightning-fast unified analytics engine*. https://spark.apache.org/.
[18] Apache Software Foundation. 2015. *Apache Flink. Stateful Computations over Data Streams*. https://flink.apache.org/.

Gefördert vom Niedersächsischen Ministerium für Wissenschaft und Kultur unter Fördernummer ZN3489 im Niedersächsischen Vorab der Volkswagen Stiftung und betreut vom Zentrum für digitale Innovationen (ZDIN).