h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbi
FACHBEREICH INFORMATIK

Hochschule Darmstadt
- Fachbereich Informatik -

Permissioned Blockchains für B2B

Prototypische Implementierung eines dezentralisierten Wartungsmarktes

Abschlussarbeit zur Erlangung des akademischen Grades
Bachelor of Science (B.Sc.)

vorgelegt von

Eric Nagel
Matrikelnummer

Referent:	Prof. Dr. Andreas Müller
Korreferent:	Björn Bär

Abstract

Traditionelle B2B-Anwendungen mit multiplen Geschäftspartnern als Teilnehmer bringen verschiedene Probleme mit sich. Wenn jedes Unternehmen seine eigenen Daten speichert, erfolgt der Zugriff auf diese, für Kooperationspartner, aufwändig über Schnittstellen. Die Daten könnten sich auch bei einer einzelnen, eventuell nicht vertrauenswürdigen Instanz befinden, welche die Kontrolle über diese hat. Um dieses Problem zu lösen wird eine dezentrale B2B-Anwendung mittels der Blockchain-Technologie entwickelt. Die bekanntesten Implementationen dieser, wie Bitcoin und Ethereum, bringen jedoch Nachteile hinsichtlich Datenschutz, Sicherheit und Transaktionsdurchsatz mit sich, welche im B2B-Bereich nicht wünschenswert sind. Diese werden analysiert, um anschließend eine Aussage über den Nutzen von B2B-Blockchain-Anwendungen zu treffen, und sinnvoll den dezentralen Wartungsmarkt zu entwickeln.

Inhaltsverzeichnis

Abbildungsverzeichnis **v**

Tabellenverzeichnis **vi**

1 Einführung und Motivation **1**

2 Blockchain-Grundlagen **3**

2.1 Funktionsweise . . . 3

2.1.1 Allgemein . . . 3

2.1.2 Konsensmechanismen . . . 4

2.1.3 Nichtangreifbarkeit/Immutability . . . 7

2.2 Blockchaintypen . . . 8

2.3 Exemplarische Anwendungsfälle . . . 9

3 Dezentraler Wartungsmarkt - Konzept **12**

3.1 Allgemein . . . 12

3.2 Anforderungen . . . 12

4 Aktueller Stand der Technik **15**

5 Evaluierung Permissioned Blockchains für B2B **16**

5.1 Skalierbarkeit . . . 16

5.1.1 Public Blockchains . . . 16

5.1.2 Ethereum . . . 18

5.1.3 Permissioned Blockchains . . . 20

5.2 Konsensmechanismen . . . 22

5.2.1 Proof of Stake . . . 22

5.2.2 Proof of Elapsed Time . . . 23

5.2.3 Practical Byzantine Fault Tolerance . . . 23

5.2.4 Tendermint . . . 24

5.2.5 Leader Based Consensus . . . 24

5.2.6 Federated Byzantine Agreement . . . 24

5.2.7 Diversity Mining Consensus . . . 24

5.2.8 Stellar Consensus . . . 24
5.2.9 Raft . . . 24
5.2.10 BFT-SMaRt . . . 24
5.3 Sonstige . . . 24
5.3.1 Private Transaktionen . . . 24
5.3.2 Code Execution . . . 24
5.3.3 Datenmenge . . . 24

6 Dezentraler Wartungsmarkt - Prototyp 25
6.1 Technologieauswahl . . . 25
6.2 Hyperledger Fabric und Composer . . . 25
6.2.1 Hyperledger Fabric . . . 25
6.2.2 Hyperledger Composer . . . 25
6.3 Modell . . . 25
6.4 Gerätesimulation durch Bosch XDK . . . 25
6.5 Programmlogik . . . 25
6.6 Benutzeroberflächen . . . 25
6.7 Konsensmechanismus . . . 26
6.8 Evaluierung . . . 26

7 Fazit und Ausblick 27

Literaturverzeichnis 28

Abbildungsverzeichnis

2.1 Verkettung von Blöcken durch Block Header Hashes 4
2.2 Signieren und Verifizieren von Nachrichten. Der Sender signiert die Nachricht mit seinen Private Key und der Empfänger kann diese mit den Public Key des Senders verifizieren. 5
2.3 Fork-Visualisierung - Vor dem Fork besitzen alle Nodes Block O als letzten Block [19]. 7
2.4 Fork-Visualisierung - 2 Nodes finden zur ungefähr gleichen Zeit einen Block und verbreiten ihn im Netzwerk, womit 2 Versionen der Blockchain bestehen [19]. . . 8
2.5 Fork-Visualisierung - Eine Node, welche Block A zuerst erhalten hat, hängt daran einen neuen Block C an [19]. 9
2.6 Fork-Visualisierung - Block C verbreitet sich im Netzwerk, rote Nodes sehen zwei Blockchains und akzeptieren die längere [19]. 10

5.1 Möglicher Transaktionsdurchsatz bei Bitcoin, Ethereum, Paypal und Visa [10]. . 17
5.2 Auswahl der gültigen Blockchain. In Bitcoin die längere Blockchain. In Ethereum die Blockchain 19
5.3 Vergleich des Transaktionsdurchsatzes von Ethereum und Hyperledger Fabric [41]. 21

Tabellenverzeichnis

Listingverzeichnis

Kapitel 1

Einführung und Motivation

Klassische B2B-Anwendungen bringen diverse Probleme hinsichtlich der Datenhaltung mit sich. Eigene Daten können bei jeden Geschäftspartner selber gespeichert werden, was jedoch den Zugriff auf diese, aufgrund von aufwendig einzurichtenden Schnittstellen und uneinheitlichen Datenformaten, erschwert. Eine andere Möglichkeit ist die Speicherung bei einem zentralen Unternehmen. Dieses hätte jedoch die Kontrolle über die Daten, womit alle anderen Parteien diesem vertrauen müssten. Diese Faktoren machen B2B-Anwendungen für die Teilnehmer unattraktiv und erschweren die Entwicklung [35] [54] [40].

Um diese Probleme zu lösen, wird ein Prototyp einer dezentralen B2B-Applikation basierend auf der Blockchain-Technologie entwickelt. Sie erlaubt es dezentrale Systeme aufzubauen, in welchen sich die Parteien nicht vertrauen. Alle Daten würden bei jedem Teilnehmer des Netzwerks gespeichert werden. Trotzdem sind diese nicht lösch- oder manipulierbar, alle Transaktionen sind lückenlos nachvollziehbar und es besteht ein gemeinsamer Konsens über den Datenbestand [26]. Bei der zu entwickelnden Applikation handelt es sich um einen automatisieren und dezentralisierten Wartungsmarkt. Teilnehmer an diesem sind Unternehmen und Wartungsdienstleister. Die Unternehmen besitzen IoT-Geräte, welche automatisch erkennen, dass sie eine Wartung benötigen. Sie legen für die Wartung einen Smart-Contract an, welcher von Wartungsdienstleistern angenommen werden kann. Diese melden sich an dem Gerät an und loggen die durchgeführten Wartungsschritte. Die Maschine schließt nach durchgeführter Wartung den Vertrag. Somit besteht ein automatisierter Wartungsmarkt zwischen mehreren Unternehmen, in welchen Wartungen verfolgbar und unveränderbar dokumentiert werden sowie kein Vertrauen zwischen den Parteien nötig ist.

Bekannte Blockchain-Implementationen, wie Bitcoin oder Ethereum, bringen jedoch Probleme mit sich, welche im B2B-Bereich von Nachteil sind. So sind alle Daten öffentlich einsehbar, der Transaktionsdurchsatz ist gering und die Konsensmechaniken sind unter bestimmten Umständen unsicher und resultieren in hohen Energieverbrauch [19][39][13].

Ziel dieser Arbeit ist es, die Probleme der Blockchain-Technologie für den B2B-Bereich zu analysieren und basierend auf den Ergebnissen die Entwicklung einer dezentralen B2B-Anwendung zu beschreiben sowie zu evaluieren. Dazu werden zunächst die grundlegenden Kon-

zepte der Blockchain-Technologie erklärt, um ein besseres Verständnis für die Vor- und Nachteile dieser zu erhalten. Anschließend werden die Probleme für B2B-Anwendungen anhand der Anforderungen an dem Wartungsmarkt genauer betrachtet und analysiert. Daraufhin erfolgt die Beschreibung der Anwendungsentwicklung. Zuletzt wird ein Fazit zur Lösung der Probleme und des entwickelten Systems gezogen.

Kapitel 2

Blockchain-Grundlagen

2.1 Funktionsweise

Die Funktionsweise der Blockchain wird hauptsächlich an Bitcoin erklärt. Als erste Blockchain-Anwendung [55] und aufgrund der relativ geringen Komplexität liefert es die Grundlage für die Funktion der Technologie. Andere Implementationen, wie Ethereum oder Ripple, funktionieren nach dem gleichen Prinzip.

2.1.1 Allgemein

Wenn der Begriff "Die Blockchain" auftaucht, ist damit meistens die Blockchain-Technologie gemeint. Es gibt nicht nur eine global bestehende Blockchain und auch nicht nur eine Implementation der Technologie, was man an Bitcoin oder Ethereum sehen kann.

Allgemein kann man die Blockchain als Datenstruktur bezeichnen, welche verteilt, nicht löschbar und unmanipulierbar gespeichert werden kann. Weiterhin verifizieren jegliche Teilnehmer am Netzwerk ausgeführte Transaktionen, womit ein gemeinsamer Konsens über den Datenbestand besteht [26].

In einer Blockchain werden Transaktionen in Blöcken gespeichert. Dabei handelt es sich um Operationen, welche Daten erstellen, verändern, oder löschen. Aus diesen lässt sich letztendlich der aktuelle Datenbestand ermitteln. So erfolgt z.B. bei Bitcoin keine Speicherung des aktuellen Guthabens der Teilnehmer. Es wird nur aus allen bestehenden Transaktionen berechnet [19]. Die Daten welche letztendlich bestehen, können z.B. Geldtransferinformationen (Bitcoin), Smart Contracts (Ethereum, selbstausführende Verträge mit selbst erstellter Programmlogik, siehe 2.2), oder simple Dokumente oder Informationen sein [13][39][51]. Die Blöcke setzen sich zusammen aus den Transaktionen sowie den Block Header, welcher verschiedene Metadaten, wie zum Beispiel den kombinierten Hash[1] aller Transaktionen, enthält [19].

Die Blöcke sind miteinander verkettet. Jeder Block Header enthält den Hash des vorherigen

[1] Hash: Ergebnis einer Operation, welche "eine Zeichenfolge beliebiger Länge auf eine Zeichenfolge mit fester Länge abbildet" [14].

Block Headers (Siehe Abb. 2.1). Dies ist ein wichtiges Feature zum Schutz der Blockchain vor Angriffen. Wenn ein Angreifer die Transaktionen eines Blocks zu seinen Gunsten verändern würde, würde sich der Hash des Block Headers ändern. Dieser müsste dann im darauffolgenden Block Header stehen, womit sich allerdings auch der Hash dieses Blocks ändert. Letztendlich müssten alle folgenden Blöcke manipuliert werden, um eine gültige Blockchain zu erhalten [39]. Diese Manipulation wird durch verschiedene Verfahren erschwert, welche genauer in den Kapiteln 2.1.1 und 2.1.2 erklärt werden.

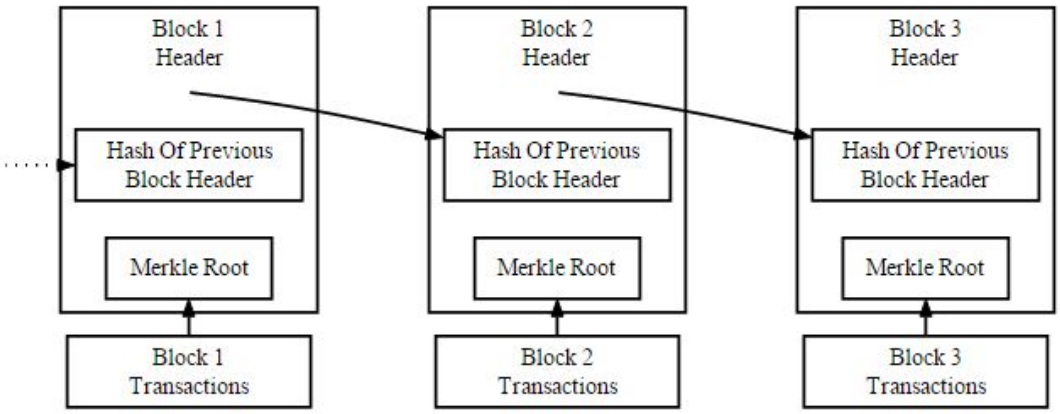

Abbildung 2.1: Verkettung von Blöcken durch Block Header Hashes

Die Blockchain ist verteilt gespeichert. Jeder Teilnehmer hat die Möglichkeit Sie auf seinen Rechner zu speichern. Somit besteht keine zentrale Instanz, welche die Kontrolle über die Daten hat. Weiterhin gibt es keinen Single Point of Failure[2], [26].

2.1.2 Konsensmechanismen

Aufgrund der verteilten Datenhaltung, muss es Verfahren geben, um die Daten synchron, und auf einen Stand, auf welchen sich alle Teilnehmer geeinigt haben, zu halten. Dazu gibt es die sogenannten Konsensmechanismen, welche gleichzeitig die Unmanipulierbarkeit der Daten sicherstellen. Bevor diese erklärt werden können, muss zunächst genauer auf die Funktion des Netzwerks eingegangen werden.

Wenn ein Teilnehmer eine Transaktion ausführt, wird diese, vorausgesetzt dass sie valide ist (Genauer im nächsten Absatz erklärt), an alle Nodes[3] (Teilnehmer, welche die Blockchain speichern) im Netzwerk weitergeleitet und im Transaktionspool aufgenommen. Dieser enthält alle noch nicht in Blöcken vorkommenden Transaktionen. Diese werden in einen neuen Block aufgenommen, und jede Node beginnt mit der Erstellung von diesem. Das Erstellen wird durch verschiedene Konsensmechaniken realisiert. Bei Bitcoin und Ethereum findet der Proof-of-Work

[2]Single Point of Failure: Komponente eines Systems, dessen Ausfall den Ausfall des gesamten Systems bewirkt [11].

[3]

Anwendung (Genauer im folgenden Absatz erklärt). Sobald eine Node einen Block erstellt, wird dieser im Netzwerk verteilt. Jede Node hängt ihn an ihre lokale Blockchain an, und beginnt mit der Erstellung des nächsten Blocks [**?**].

Damit eine Transaktion valide ist, muss sie bestimmte Voraussetzungen erfüllen. So muss sie unter anderen mit den Private Key des Senders signiert sein. Mittels seines Public Keys kann überprüft werden, ob wirklich er der Sender der Nachricht ist und ob die Transaktion manipuliert wurde. Dieses Verfahren wird auch in der Abbildung 2.2 visualisiert. Das Signieren trägt zur Sicherheit der Blockchain bei, da ein Angreifer somit keine Transaktionen manipulieren oder im Namen eines anderen ausführen kann. In Bitcoin ist eine weitere Kondition, dass der Transaktionsersteller die zu sendenen Bitcoins besitzt [19]. In Systemen wie Ethereum und Hyperledger Fabric, in welchen eigene Programmlogik abgebildet werden kann, können weitere Konditionen festgelegt werden. So muss z.B. ein Teilnehmer die nötigen Rechte haben um eine Transaktion auszuführen [1].

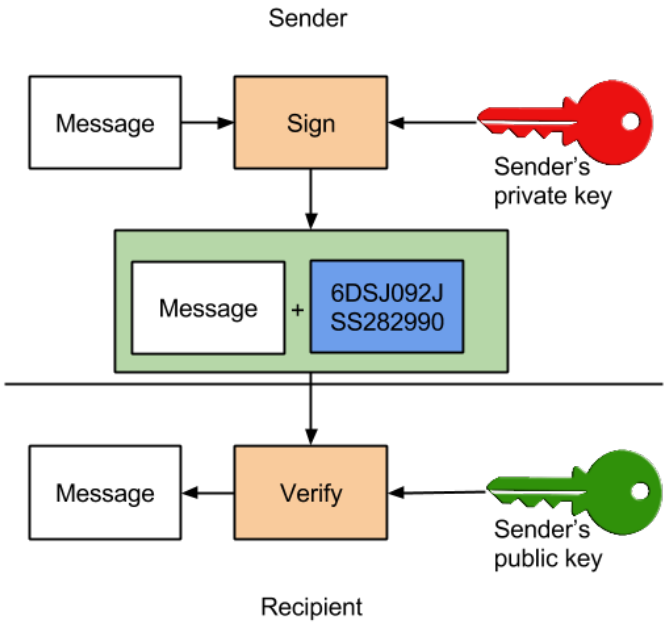

Abbildung 2.2: Signieren und Verifizieren von Nachrichten. Der Sender signiert die Nachricht mit seinen Private Key und der Empfänger kann diese mit den Public Key des Senders verifizieren.

Der Proof-of-Work ist nur einer der zur Verfügung stehenden Konsensmechanismen (Siehe Kapitel 5.1.3). Er bedarf jedoch genauerer Erklärung, da er in aktuellen Blockchain-Implementationen vorwiegend genutzt wird. Der Proof-of-Work ist eine Art Rätsel, welches mit der Rechenleistung von Nodes gelöst werden muss, um einen Block zu erschaffen. Genauer gesagt, muss für einen Block ein Hash gefunden werden, welcher einen bestimmten Wert unterschreitet. Desto kleiner dieser ist, desto höher ist die Schwierigkeit. Um wachsender Rechenleistung und Teil-

nehmerzahl entgegen zu wirken, also die Zeit für die Erstellung eines Blockes ungefähr gleich zu halten, kann die Schwierigkeit angepasst werden. Dies ist aufgrund verschiedener Faktoren nötig, welche genauer im Kapitel 5 erläutert werden. Um unterschiedliche Hashwerte für gleiche Blöcke zu erhalten, gibt es im Block Header eine Nonce[4], welche verändert wird [39]. Alle Nodes im Bitcoin-Netzwerk benötigen im Durchschnitt 10 Minuten um einen Proof-of-Work zu erbringen [19], bei einer Hash Rate[5] von ca. 13.000.000 TH/s (Terrahashes pro Sekunde) [9]. Bei Ethereum beträgt die Zeit ungefähr 15 Sekunden [6], bei einer Hash Rate von ca. 150 TH/s [7]. Damit die Nodes eine Motivation haben, Rechenleistung für das Erstellen von Blöcken zu nutzen, erhalten sie bei Erbringung des Proof-of-Work eine Belohnung in Form von Währung [39] [13].

Um vollständig zu verstehen, wie der Proof-of-Work funktioniert, muss das Forking erklärt werden. Wenn eine Node einen Proof-of-Work erbringt, also einen Block erstellt, wird dieser an alle anderen Nodes weitergeleitet. Im Bitcoin-Netzwerk dauert es bei einer maximalen Blockgröße von 1MB [19], zwischen 6 und 20 Sekunden, bis ein Block mindestens 90% aller Nodes erreicht hat [3]. Dies stimmt auch mit den Paper von Decker und Wattenhofer überein, wo eine durchschnittliche Zeit von 12,6 Sekunden angegeben wird, bis ein Block 95% aller Nodes erreicht [28]. In dieser Zeit kann es vorkommen, dass eine weitere Node einen Block erstellt. Auch dieser wird im Netzwerk verteilt, womit 2 Versionen der Blockchain existieren: Eine endet mit Block A, und die andere mit Block B. Dies ist der sogenannte Fork. Das Netzwerk muss sich nun darauf einigen, welche der beiden Versionen beibehalten werden soll. Deshalb gilt: Die Blockchain in welche mehr Arbeit eingeflossen ist, ist die gültige. Im Falle von Bitcoin wäre dies die längere Blockchain. Die Nodes probieren an den zuerst erhaltenen Block (A oder B) einen neuen anzuhängen. Gelingt dies, ist eine der beiden Blockchains länger als die andere. Diese wird dann von allen Nodes als die richtige akzeptiert. Dieser Vorgang wird auch in den Abbildungen 2.3 bis 2.6 dargestellt. Theoretisch ist es möglich, dass ein Fork über mehrere Blöcke besteht. Die Wahrscheinlichkeit dafür ist jedoch gering, da mehrmals nacheinander mindestens 2 Nodes zur ungefähr gleichen Zeit einen Block erstellen müssen. Auch zu erwähnen ist, dass in einem Fork-Branch weitere Forks entstehen können. Diese Forks sind der Grund, warum Transaktionen erst als bestätigt gelten, sobald sie in einem Block stehen, welcher eine gewisse Anzahl an Nachfolgern hat. Denn erst dann ist die Sicherheit gegeben, dass die Transaktion nicht in einem Fork vorhanden ist, welcher eventuell verworfen wird [19]. Wie genau der Proof-of-Work das Netzwerk absichert, wird im Kapitel 2.1.2 erklärt.

Neben dem Proof-of-Work gibt es noch weitere Konsensmechanismen, wie Proof-of-Stake, Proof-of-Authority oder Practical Byzantine Fault Tolerance [48], [27] . Diese werden im Kapitel 5.1.3 genauer beschrieben und analysiert.

[4]Nonce: Eine "Zahlen- oder Buchstabenkombination, [...] die nur ein einziges Mal in dem jeweiligen Kontext verwendet wird"[15].

[5]Hash Rate: Anzahl der in einer Zeiteinheit berechneten Hashwerte [8].

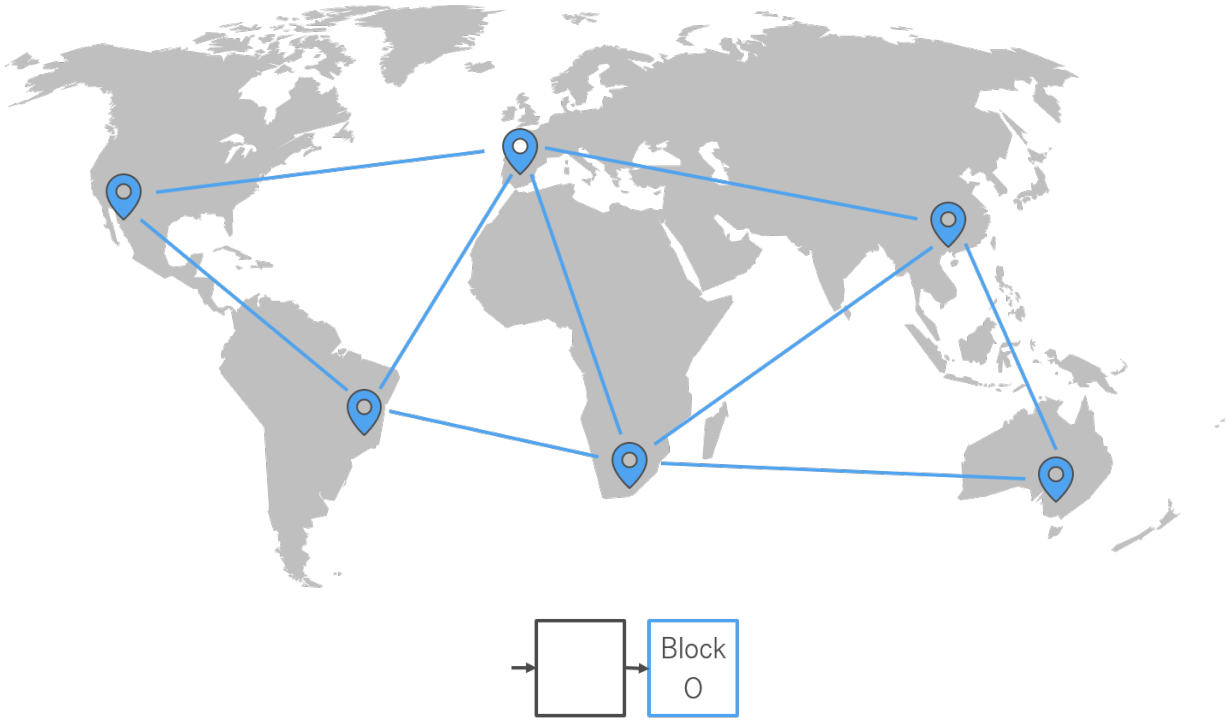

Abbildung 2.3: Fork-Visualisierung - Vor dem Fork besitzen alle Nodes Block O als letzten Block [19].

2.1.3 Nichtangreifbarkeit/Immutability

Viele Faktoren tragen zur Nichtangreifbarkeit und Unveränderlichkeit der Blockchain bei. Da alle Nodes die ausgeführten Transaktionen auf Validität prüfen, können diese nicht ohne Berechtigung, im Namen einer anderen Identität, oder mit unzureichenden Konditionen ausgeführt werden. Der wichtigste Faktor ist jedoch der genutzte Konsensmechanismus in Verbindung mit den verketteten Blöcken. Durch ihm wird sichergestellt, dass bestehende Daten nicht gelöscht oder manipuliert werden können.

Ein Beispiel dafür kann am Proof-of-Work gezeigt werden. Ein Angreifer probiert eine Transaktion aus einen bestehenden Block zu entfernen. Dazu würde er die Transaktion bei seiner lokalen Blockchain entfernen. Nun ist jedoch der Hash des Blockes sowie der Block selber nicht valide und würde von keiner Node akzeptiert werden. Der Angreifer muss also erneut einen Proof-of-Work für den manipulierten Block erbringen. Dies wäre für eine Einzelperson jedoch Zeitaufwändig, wenn man bedenkt, das extra für diesen Zweck produzierte Hardware eine Hash Rate von bis zu 13,5 TH/s erreicht [38]. Dies Wenn der manipulierte Block nun noch Nachfolger hat, muss aufgrund des neuen Hashes auch für diese der Proof-of-Work erbracht werden. Hinzu kommt, dass die Blockchain des Angreifers erst von allen Nodes akzeptiert wird, wenn sie länger ist. Er müsste also schneller als das gesamte Bitcoin-Netzwerk Blöcke erschaffen können. Dies ist nur möglich, wenn er 51% der Rechenleistung des Netzwerks besitzt. Deshalb wird dieser Angriff auch 51%-Angriff genannt [49] [13].

An dieser Stelle sollte erähnt werden, dass auch wenn ein 51%-Angriff erfolgt, die Angriffs-

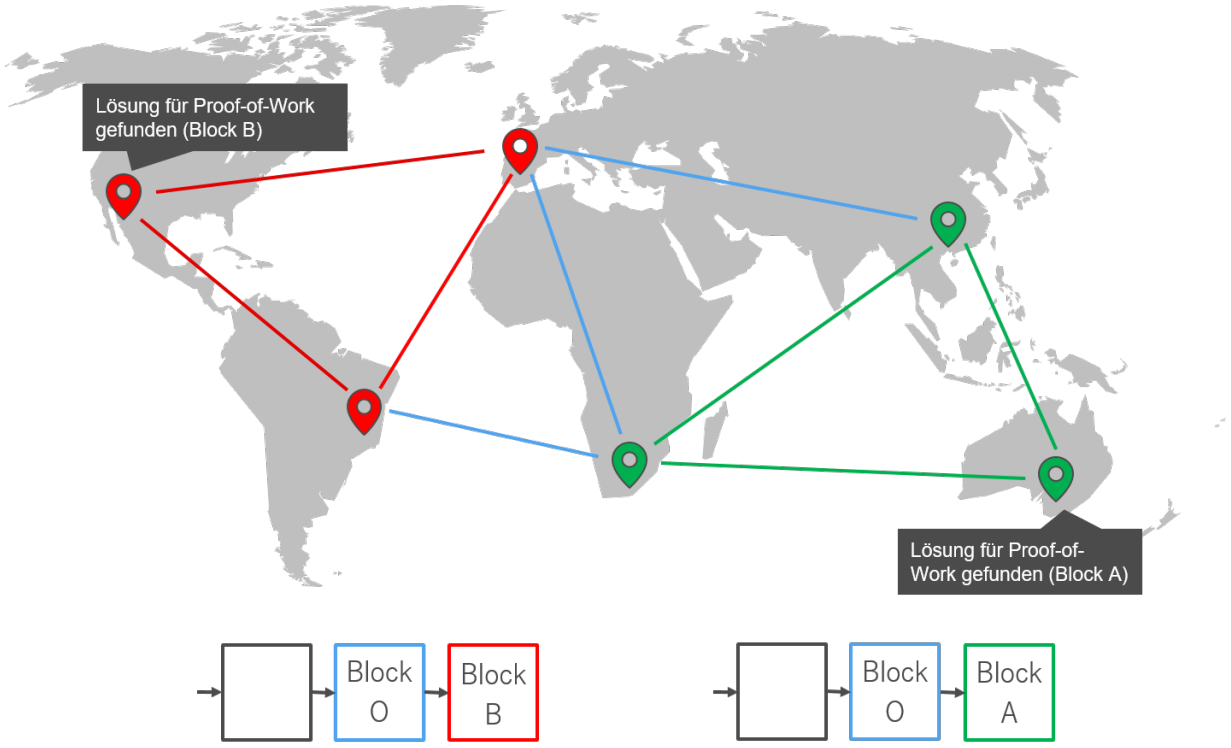

Abbildung 2.4: Fork-Visualisierung - 2 Nodes finden zur ungefähr gleichen Zeit einen Block und verbreiten ihn im Netzwerk, womit 2 Versionen der Blockchain bestehen [19].

möglichkeiten beschränkt sind. Der Angreifer kann keine unvaliden Transaktionen sowie Blöcke erstellen. Ihm ist es möglich DoS-Angriffe auszuführen indem er verhindert das bestimmte Transaktionen in Blöcke aufgenommen werden. Genau so kanner die Historie der Daten verändern, indem er eine Transaktion aus einem Block entfernt und diese nicht erneut in einen Block aufnimmt, oder dafür sorgt, dass sie ungültig werden. So gibt es z.B. im Falle von Kryptowährungen folgende Form eines Double-Spending-Angriffs: Ein Angreifer sendet z.B. Bitcoins an einen Händler. Dieser wartet auf die Bestätigung der Transaktion in einen Block sowie auf nachfolgende Blöcke. So stellt er sicher, dass die Transaktion nicht in einem eventuell verworfenen Fork stand. Erst dann versendet er die Ware. Anschließend ersetzt der Angreifer die Transaktion durch eine Zahlung an sich selber und erstellt die längere Blockchain, womit der Händler letztendlich kein Geld erhalten hat [13]. Auch zu bedenken ist, dass ein Nutzer mit 51% der Rechenleistung wenig Motivation hat Angriffe auszuführen, da er für jeden erstellten Block Kryptowährung als Belohnung erhält. Der Wert der Kryptowährung würde sinken, wenn Angriffe auf die Blockchain entdeckt werden. Deshalb besteht für die sogenannten Miner eine Motivation, ehrlich zu arbeiten [19].

2.2 Blockchaintypen

Es gibt 3 Typen von Blockchains, welche die zugelassenen Teilnehmer bestimmen. Bisher wurden nur Public Blockchain-Anwendungen, wie Bitcoin und Ethereum erwähnt. In diesen gibt

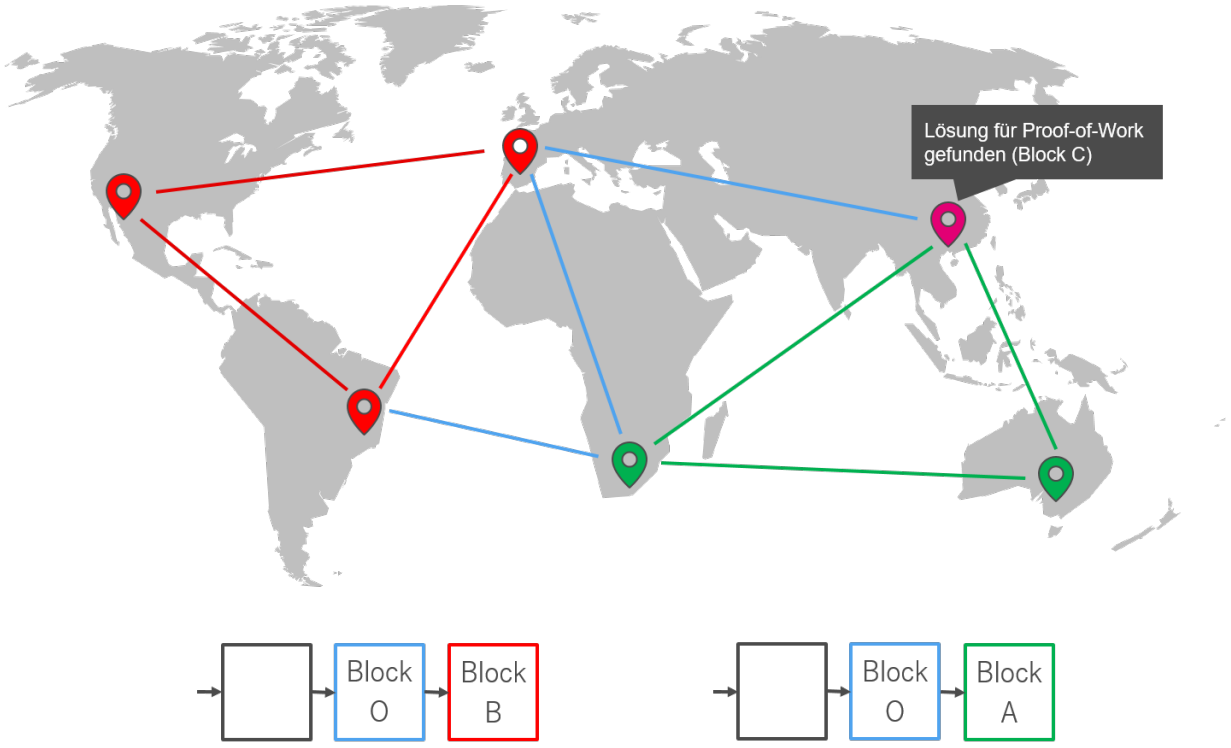

Abbildung 2.5: Fork-Visualisierung - Eine Node, welche Block A zuerst erhalten hat, hängt daran einen neuen Block C an [19].

es keine Teilnehmerbeschränkungen, jeder kann am Netzwerk teilnehmen und die Blockchain öffentlich einsehen. Anders ist dies bei Permissioned (oder auch Consortium [20]) und Private Blockchains. Die beiden Begriffe werden in einigen wissenschaftlichen Arbeiten gleichgesetzt (Siehe [32], [41], [37]). Hier folgt jedoch eine Unterscheidung. Dabei ist eine Private Blockchain, eine Blockchain welche nur von einen Nutzer verwendet wird. Da eine solche Anwendung keinen Sinn macht, da keine Vorteile der Blockchain genutzt werden können, wird darauf nicht genauer eingegangen. Interessanter sind Permissioned Blockchains, an welchen nur zugelassene Nutzer teilnehmen dürfen. Nur diese sind berechtigt, Transaktionen auszuführen und die Daten einzusehen [37]. Dies bietet sich vor allem bei B2B-Anwendungen an, welche von verschiedenen Unternehmen genutzt werden sollen. In diesen kann es aufgrund von z.B. sensiblen Daten nötig sein, das nur bestimmte Parteien Zugriff auf die Blockchain haben. An dieser Stelle sollte auch erwähnt werden, dass es möglich ist Blockchain-Implementationen wie z.B. Ethereum als Permissioned Blockchain zu nutzen [42].

2.3 Exemplarische Anwendungsfälle

Die Blockchain wird als revolutionäre Technologie angepriesen (Siehe [50]). Trotzdem ist es wichtig zu wissen, für welche Zwecke sie wirklich geeignet ist. Grundsätzlich macht eine Blockchain Sinn, wenn mehrere Parteien, welche sich nicht vertrauen, mit einen System interagieren wollen, welches von keiner dritten zentralen Instanz verwaltet wird [54]. Um eine bessere Vor-

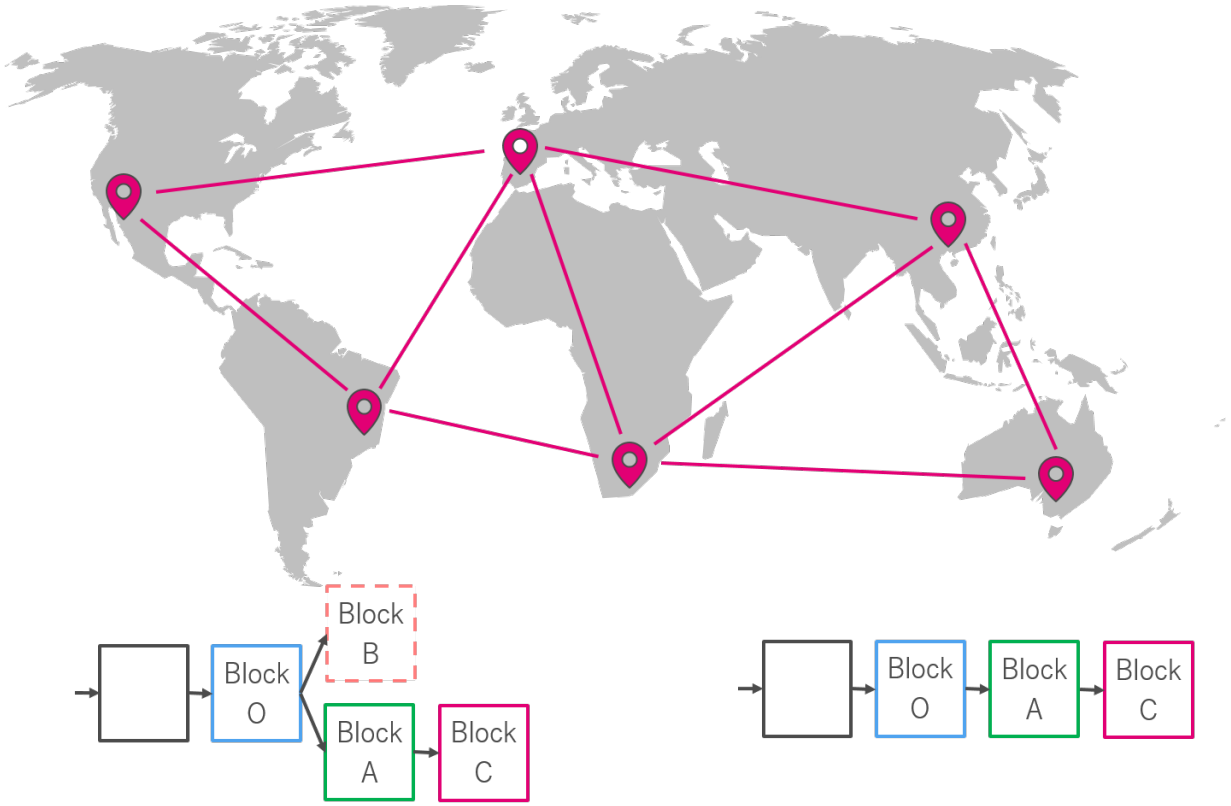

Abbildung 2.6: Fork-Visualisierung - Block C verbreitet sich im Netzwerk, rote Nodes sehen zwei Blockchains und akzeptieren die längere [19].

stellung zu solchen Anwendungen zu erhalten, werden im Folgenden verschiedene Exemplarische Anwendungsfälle genannt und beschrieben.

Der erste Anwendungsfall, mit welchen die Blockchain-Technologie auch entstanden ist, sind Kryptowährungen. Mit Ihnen ist es möglich Geld zwischen beliebigen Parteien zu übertragen, ohne dass die Transaktionen von einer eventuell nicht vertrauenswürdigen Bank oder ähnlichem kontrolliert und verwaltet werden [49].

Weitere Anwendungsfälle ergeben sich mit der Möglichkeit Programmlogik auf der Blockchain abzubilden. So können beispielsweise dezentrale Online-Wahlen realisiert werden. Die Stimmen würden in der Blockchain gesammelt werden, und können so letztendlich nicht mehr von z.B. einer korrupten Regierung manipuliert werden [23].

Ein weiterer Anwendungsfall, insbesondere für den B2B-Bereich, wäre Supply Chain Management. Über eine digitale Lieferkette sollen Material- und Informationsflüsse zu Produkten und Dienstleistungen aufgebaut und verwaltet werden [36]. Dies erlaubt Unternehmen das automatisieren von Prozessen und das verbesserte reagieren auf Ereignisse (z.B. Lieferverspätungen). Weiterhin ist es dem Unternehmen möglich, dem Kunden exakt aufzuzeigen wo es und seine Unterprodukte produziert wurden. In klassischen B2B-Anwendungen müsste jedes Unternehmen, welches ein Teil der Supply Chain ist, die relevanten Daten durch z.B. APIs[6] bereitzustellen. Dies bedeutet Aufwand, da diese erstmal entwickelt werden müssen. Weiterhin müsste ein Sys-

[6] API: Schnittstelle zur Anwendungsprogrammierung[16]

tem von all diesen unterschiedlichen Schnittstellen die Daten abfragen und in einem System zusammenführen um die Supply Chain zu erstellen. Diese müssten dann abgefragt werden um die Supply Chain zu erstellen. Aufgrund dieses Aufwandes werden oft Dritte eingestellt, welche sich um den Aufbau und um die Datenintegration der Supply Chain kümmern. Den Aufwand, sowie die eventuell nicht vertrauenswürdige dritte Partei könnte man durch die Nutzung der Blockchain überspringen. In dieser könnte jedes Unternehmen die relevanten Daten speichern, ohne das aufwändige Erstellen von Schnittstellen. Die Supply Chain wäre direkt in der Blockchain vorhanden, und kein Unternehmen muss Datenmanipulation oder ähnliches befürchten [35].

Auch dezentrale Märkte sind für B2B-Anwendungen interessant. Der zentrale Marktplattformbetreiber, wie z.B. Ebay oder Amazon, welcher persönliche Informationen speichert und Gebühren für den Verkauf von Artikeln verlangt, wäre hinfällig. Nutzer könnten Waren untereinander verkaufen, während die Blockchain als Notar für den Warenaustausch dient [20].

Ein weiteres Beispiel wären Blockchain Sharing-Systeme. So könnte ein dezentrales Fahrradleihsystem aufgebaut werden. Nutzer würden mit ihrem Smartphone, über ihre in der Blockchain hinterlegte Identität (Im Falle von z.B. Ethereum die Wallet-Adresse[7]), das Fahrrad entsperren. Dieser erkennt automatisch die gefahrene Distanz sowie die Nutzungsdauer. Über einen Smart Contract würde anschließend die automatische Zahlung erfolgen. Neben der Automatisierung besteht der Vorteil, dass mehrere Unternehmen oder auch Privatpersonen Leihfahrräder anbieten können, ohne dass sie einer zentralen Instanz mit der Verwaltung vertrauen müssen [4], [30].

[7]Wallet: Speichert z.B. bei Ethereum und Bitcoin den Private Key des Nutzers und wird z.B. als Adresse für Zahlungen genutzt [5].

Kapitel 3

Dezentraler Wartungsmarkt - Konzept

3.1 Allgemein

Ziel dieser Arbeit ist die Entwicklung einer prototypischen B2B-Applikation in Form eines automatisierten sowie dezentralisierten Wartungsmarktes. Teilnehmer an diesem sind multiple Unternehmen und Wartungsanbieter. Erstere besitzen IoT-Geräte, welche erkennen können, dass sie eine Wartung benötigen. Die Wartungsanbieter erhalten die Informationen zur Wartung, und können sich für diese anmelden. Anschließend würden sie diese durchführen und dabei die Wartungsschritte loggen.

In klassischen B2B-Anwendungen wäre die Realisierung dieses Systems auf 2 Arten erfolgt. Bei ersterer gäbe es eine dritte Partei, welche den Markt verwaltet, und bei welcher sich alle Unternehmen und Wartungsanbieter anmelden müssen (z.B. Ebay). Die andere Möglichkeit wäre, dass eines der teilnehmenden Unternehmen den Markt verwaltet. Bei beiden Optionen müssten die Teilnehmer am Markt ihre Daten einer eventuell nicht vertrauenswürdigen zentralen Instanz zur Verfügung stellen. Weiterhin würde bei jeden Unternehmen die Notwendigkeit bestehen, API's für den Datenzugriff zu erstellen.

Um dies zu verhindern, wird der Wartungsmarkt auf Basis der Blockchain-Technologie implementiert. Somit können beliebig viele Unternehmen und Wartungsanbieter an dem System teilnehmen, ohne dass eine Datenmanipulation durch die Teilnehmer oder eine zentrale Instanz befürchtet werden muss.

3.2 Anforderungen

Es ergeben sich verschiedene Anforderungen an das zu entwickelnde System. Die Spezifizierung dieser ist wichtig, denn auf Basis von diesen wird eine Blockchain-Implementation ausgewählt und auf verschiedene Probleme analysiert. Dieser werden hier zunächst einmal aufgelistet um einen Überblick zu erhalten.

Es ergeben sich verschiedene funktionale Anforderungen:

- Registrieren und Identifizieren von Unternehmen, Wartungsanbietern und Geräten in der Blockchain
- Wartungsgeräte kündigen Wartungen in Form eines Smart Contracts in der Blockchain an
- Wartungsanbieter können den Smart Contract unter bestimmten Konditionen annehmen
- Wartungsanbieter loggen Wartungsschritte in der Blockchain
- Gerät überprüft ob die Wartung erfolgt ist, und schließt den Contract
- Nur bestimmte Teilnehmer können bestimmte Transaktionen ausführen
- Private Transaktionen sollen zwischen Teilnehmern möglich sein

Folgende nicht-funktionale Anforderungen existieren:

- Hoher Transaktionsdurchsatz und geringe Transaktionszeiten
- Nichtangreifbarkeit der Daten

Einige Anforderungen sollten genauer erklärt werden:

Registrieren und Identifizieren von Unternehmen, Wartungsanbietern und Geräten in der Blockchain Die zu entstehende B2B-Anwendung soll zwischen verschiedenen Unternehmen bestehen. Diese müssen Berechtigungen erhalten um am Netzwerk teilzunehmen, und ausgeführte Transaktionen sollen ihnen zugeordnet werden können.

Wartungsgeräte kündigen Wartungen in Form eines Smart Contracts in der Blockchain an Es kann verschiedene Gründe für die Wartung geben. So kann z.B. ein Wartungsdatum erreicht werden, oder Sensorwerte weisen auf einen Fehler hin.

Wartungsanbieter können den Smart Contract unter bestimmten Konditionen annehmen Eine dieser Konditionen könnte sein, dass der Wartungsanbieter bereits Erfahrungen mit der Wartung von bestimmten Geräten hat. Diese Information könnte ebenfalls aus der Blockchain abgefragt werden. Weiterhin darf der Vertrag z.B. noch nicht von einen anderen Anbieter akzeptiert worden sein.

Gerät überprüft ob Wartung erfolgt ist, und schließt den Contract Die Überprüfung kann anhand der geloggten Schritte sowie Sensorwerten erfolgen, welche vor und nach der Wartung existiert haben.

Nur bestimmte Teilnehmer können bestimmte Transaktionen ausführen Die verschiedenen Teilnehmer haben unterschiedliche Rechte. So soll es z.B. einen Unternehmen nicht möglich sein, Wartungsverträge zu bearbeiten oder zu akzeptieren.

Private Transaktionen sollen zwischen Teilnehmern möglich sein Bei Blockchains wie Bitcoin und Ethereum sind alle Daten in der Blockchain für alle Teilnehmer einsichtbar. Aufgrund von sensiblen Daten kann es allerdings vorkommen, das nicht alle Transaktionen für alle Teilnehmer sichtbar sein sollen. Im Falle des Wartungsmarktes sollen z.B. Preisabsprachen zwischen Unternehmen und Wartungsdienstleistern privat erfolgen.

Hoher Transaktionsdurchsatz und geringe Transaktionszeiten In Bitcoin ist lediglich ein Transaktionsdurchsatz von 7 Transaktionen pro Sekunde möglich [55]. Hinzu kommt, dass es ca. zwischen 30 Minuten und 16 Stunden dauern kann, bis eine Transaktion bestätigt ist [21]. Darauf wird auch genauer im Kapitel 5.1 eingegangen. In dem zu entwickelnden System ist die Skalierbarkeit wichtig. Je nach der Anzahl der am Netzwerk teilnehmenden Unternehmen und Wartungsanbieter wird eine höherer Transaktionsdurchsatz benötigt. Insbesondere wenn tausende von Geräten Transaktionen in der Blockchain ausführen.

Nichtangreifbarkeit der Daten Die Nichtangreifbarkeit wird durch die genutzte Konsensmechanik realisiert. Der am häufigsten genutzte Proof-of-Work ist in einem Netzwerk mit wenig Teilnehmern allerdings unsicher, da es einfach ist 51% der Rechenleistung zu erreichen. Weiterhin führt er zu einen hohen Stromverbrauch (Im Bitcoin-Netzwerk der Verbrauch von ca. 3.500.000 US-Haushalten [2]), welcher nicht erwünscht ist.

An dieser Stelle muss darauf hingewiesen werden, dass das System um viele nützliche Features erweiterbar ist. So könnte zum Beispiel eine Bewertung der Wartungsanbieter anhand bestimmter Faktoren erfolgen. Da es sich jedoch nur um eine prototypische Implementation handelt, werden nur die Features implementiert, welche für einen Proof-of-Concept eines solchen Systems benötigt werden.

Kapitel 4

Aktueller Stand der Technik

Die Blockchain wird seit 2008 erfolgreich für Kryptowährungen eingesetzt. Mit Ethereum wird das Konzept von Smart Contracts implementiert, womit es möglich ist eigene Programmlogik in der Blockchain abzubilden und so dezentrale Anwendungen zu entwickeln. Weitere weniger bekannte Blockchain-Implementationen für Public Blockchains sind Monero, Dashcoin und Litecoin [33].

Die Technologie bringt durch ihre Architektur jedoch auch Limitationen mit sich, weshalb sie nicht für alle Anwendungszwecke geeignet ist (Siehe 5. Die Probleme werden in vielen wissenschaftlichen Arbeiten analysiert und Lösungen für diese vorgeschlagen. Trotzdem bestehen gewisse Limitationen weiterhin [55][49][44].

Permissioned Blockchains, wie Hyperledger Fabric, Quorum, Sawtooth Lake oder Quorum bieten Vorteile gegenüber dem Public Blockchains, sie bringen allerdings auch neue Herausforderungen mit sich. So muss z.B. eine Alternative zum Proof-of-Work gefunden werden, um je nach Use-Case, Performance, Skalierbarkeit und Nichtangreiffbarkeit sicher zu stellen [37].

Dezentrale Märkte sind eine der am meisten mit Blockchain in Verbindung erwähnten Use-Cases, wie man an Quellen wie [20] und [43] sehen kann. Neben Konzepten für diese (Siehe [34]), gibt es auch Live-Systeme, wie Syscoin [45].

Dezentrale Wartungsmärkte hingegen werden nur in Verbindung mit der Supply Chain erwähnt, wie zum Beispiel in [46] oder [31]. Implementationen oder Konzeptentwürfe konnten nicht gefunden werden.

Kapitel 5

Evaluierung Permissioned Blockchains für B2B

Die Blockchain-Technologie bringt diverse Probleme mit sich, welche je nach Anwendungszweck und Blockchaintyp verschieden große Auswirkungen haben. Für den B2B-Bereich gilt es vor allem die Skalierbarkeit sowie die Konsensmechanismen zu analysieren.

5.1 Skalierbarkeit

Das CAP-Theorem besagt, dass es in einem verteilten System nur möglich ist, 2 von den 3 folgenden Eigenschaften zu erfüllen: Konsistenz, Verfügbarkeit und Ausfalltoleranz. Bei der Blockchain wären dies: Dezentralisierung, Skalierbarkeit und Nichtangreifbarkeit [44]. Im Bezug auf die Skalierbarkeit wird vor allem auf den Transaktionsdurchsatz sowie die Bestätigungszeiten von Transaktionen eingegangen. Dazu erfolgt zunächst eine Analyse an aktuellen Public Blockchains, und letztendlich an Permissioned Blockchains. Die Ergebnisse werden ebenfalls auf das CAP-Theorem angewandt.

5.1.1 Public Blockchains

5.1.1.1 Bitcoin

Das Bitcoin-Netzwerk erreicht aktuell einen maximalen Transaktionsdurchsatz von 7 Transaktionen (Unterschiedlich je nach Größe der Transaktionen) pro Sekunde (TPS), bei einer Blockgröße von 1MB. Hingegen erreicht Paypal 115 TPS, und Visa 2000 TPS (Siehe auch Abb. 5.1) [10]. Hinzu kommt, dass ungefähr 170000 unbestätigte Transaktionen[1] bestehen [29]. Berechnungen von Scherer zeigen, dass bei 11,8 Millionen Nutzern im Bitcoin-Netzwerk, sowie einem Transaktionsdurchsatz von 4 TPS, jeder Nutzer nur nur ca. 10 Transaktionen im Jahr senden kann [44].

[1] Unbestätigte Transaktion: Eine Transaktion, welche noch nicht in einen Block vorkommt [19].

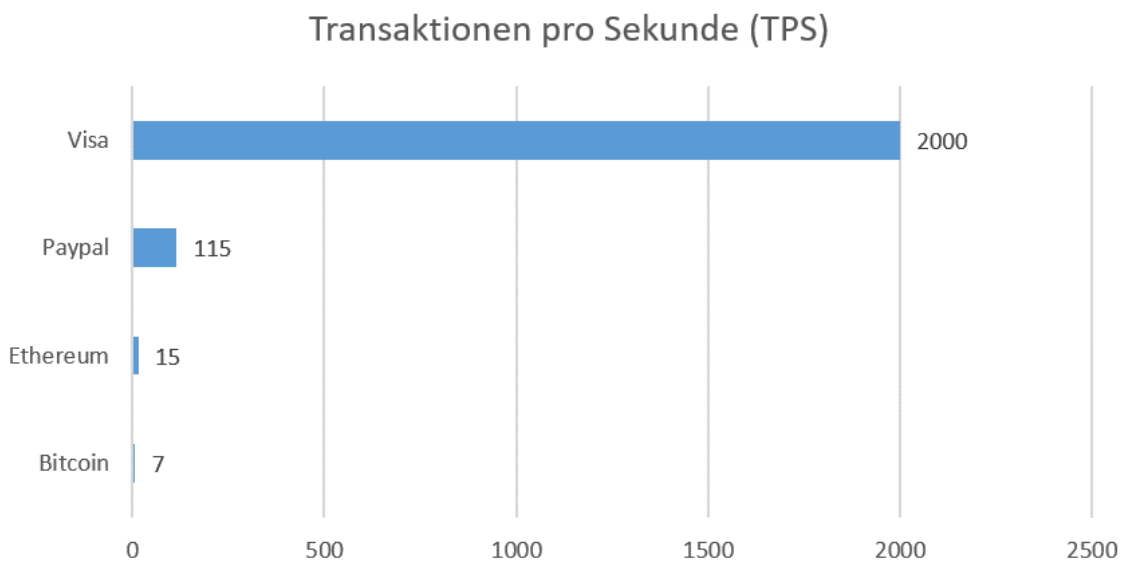

Abbildung 5.1: Möglicher Transaktionsdurchsatz bei Bitcoin, Ethereum, Paypal und Visa [10].

Der Transaktionsdurchsatz ist durch verschiedene Faktoren limitiert. Hauptsächlich durch die limitierte Blockgröße von 1MB, und dem Proof-of-Work: Nur eine bestimmte Anzahl an Transaktionen passt in einen Block, und nur alle 10 Minuten wird einer erstellt. Es gäbe also die Möglichkeit, die Blockgröße zu vergrößern, oder die Zeit für den Proof-of-Work zu verringern, indem die Schwierigkeit angepasst wird. Es gibt jedoch diverse Nachteile, welche dadurch entstehen würden. Bei einer größeren Blockgröße würde es länger dauern, bis ein Block beim Propagieren durch das Netzwerk alle Nodes erreicht. Dies würde zu öfter vorkommenden und längeren Forks führen und somit die Sicherheit des Netzwerks beeinträchtigen. Den gleichen Effekt hätte eine kürzere Proof-of-Work Zeit, da die Wahrscheinlichkeit höher ist, dass zwei Nodes zur ungefähr gleichen Zeit einen Block erstellen [44] [13] [47].

Entsteht ein Fork, probieren Nodes die längere und somit gültige Blockchain zu erschaffen. Gelingt dies, wird die kürzere Blockchain mit den nun sogenannten Stale Blocks verworfen. Die gesamte Rechenleistung, welche in die Stale Blocks und seine Nachfolger geflossen ist, trägt nicht zur Sicherheit des Netzwerks bei. Dies lässt sich auch anhand der Abbildung 5.2 erläutern. Innerhalb der Blockchain bestehen durch mehrere Forks 5 Branches. Das bedeutet, dass die Rechenleistung des Netzwerks auf diese aufgespalten ist. Es wird davon ausgegangen, dass 20% der Rechenleistung in den obersten Branch geflossen ist, welcher der längste ist. Die restlichen 4 Branches erhalten je 10% der Rechenleistung. Wenn es nun einen Angreifer mit 40% der Rechenleistung probiert eine eigene Blockchain zu erstellen, gelingt ihm dies, da er schneller die längere Blockchain erstellen kann [47]. Zusammenfassend lässt sich sagen, dass ein Angreifer nicht 51% der Rechenleistung für einen Fork benötigt, wenn das Netzwerk diese bei Forks verschwendet [22].

Ein weiteres Problem der Forks ist, dass Miner keine Belohnung für die Arbeit an verworfenen Blöcken erhalten. Dadurch kann es zur Zentralisierung durch wachsende Mining Pools

kommen. Dies wird an folgenden Beispiel ersichtlich: Ein Mining Pool A besitzt 30% der Rechenleistung, ein Mining Pool B 10%. In dem genannten Beispiel würde Mining Pool A in 70% aller Fälle einen Stale Block erzeugen, und B in 90% aller Fälle. Kein Miner würde dem Mining Pool B beitreten, da die Wahrscheinlichkeit geringer ist, dass B gültige Blöcke erschafft. A hingegen würde immer mehr Miner, und somit mehr Rechenleistung erhalten [13].

An dieser Stelle sollte auch darauf hingewiesen werden, dass schnellere Blockerstellungszeiten nicht zwingend zu schnelleren Transaktionsbetätigungen führen. Transaktionen werden zwar schneller in Blöcken aufgenommen, aber es muss auf mehr Nachfolger gewartet werden, um sicher zu gehen, dass die Transaktion nicht in einem Fork vorkommt [44].

5.1.2 Ethereum

Bessere Skalierbarkeit durch GHOST Das Ethereum Netzwerk nutzt das sogenannte GHOST-Protokoll, und erreicht damit eine Transaktionsdurchsatz von 15 TPS, bei einer durchschnittlichen Zeit von 15 Sekunden um den Proof-of-Work zu erbringen. Dieses löst Probleme des Forkings und der Benachteiligung von Minern. Ersteres wird dadurch gelöst, dass Stale Blocks in die Berechnung der gültigen Blockchain einfließen. Anders als bei Bitcoin, wo lediglich die Parents und deren Nachfolger eine Rolle spielen. Die Stale Blocks werden in Ethereum "Uncles" genannt. Kurz gesagt, ist ein Uncle ein alternativer gefundener Block welcher auf der gleichen Höhe wie der Parent bestehen würde [13].

Die Bestimmung der gültigen Blockchain wird an der Abbildung 5.2 ersichtlich. In Ethereum ist die Blockchain die gültige, für welche die meiste Arbeit aufgebracht wurde, unter Einbezug der Uncles. Das führt dazu, dass der Branch mit den meisten Uncles bestehen bleibt. Das bedeutet letztendlich, dass die gesamte Rechenleistung das Netzwerk absichert, auch wenn diese sich auf die Branches aufteilt. Ein Angreifer braucht somit weiterhin 51% der Rechenleistung um einen Angriff auszuführen [47].

Damit ist allerdings noch nicht das Problem der Zentralisierung durch Mining Pools gelöst. Es besteht weiterhin keine Motivation für Miner, Uncles zu minen. Deswegen ist das GHOST-Protokoll in Ethereum so erweitert, dass Miner Ether[2] als Belohnung für das Erstellen von Uncles erhalten (Allerdings weniger als bei vollwertigen Blöcken). Somit besteht ebenfalls die Motivation, kleineren Mining Pools beizutreten [13]. An dieser Stelle sollte auch erwähnt werden, dass Miner entscheiden können, an welchen Branch sie arbeiten [55].

Während Ethereum die Probleme löst, welche durch Forks entstehen, ist der Transaktionsdurchsatz trotzdem limitiert. Die Blockgröße muss klein genug bleiben, damit das Propagieren im Netzwerk effizient bleibt [44]. Ansonsten würden Miner unter Umständen so benachteiligt werden, dass sie nur sehr selten bis garnicht den aktuellen Block der gültigen Blockchain erhalten würden. Dies wiederum würde dazu führen, dass sie nur Uncles minen können, und so nie die volle Belohnung erhalten können. Hinzu kommt, das Uncles nur gültig sind, wenn sie maximal eine bestimmte Anzahl an Generationen vom aktuellen Block in der gültigen Chain

[2]Kryptowährung von Ethereum [13].

entfernt sind. Ansonsten hätten die Miner auch weniger Motivation ehrlich zu bleiben, da sie ohne Nachteile an der Chain eines Angreifers arbeiten könnten [13].

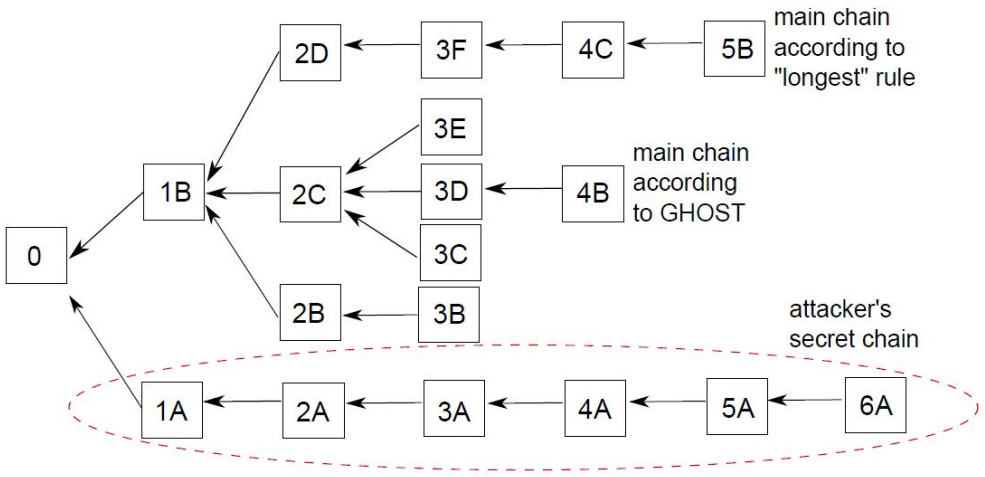

Abbildung 5.2: Auswahl der gültigen Blockchain. In Bitcoin die längere Blockchain. In Ethereum die Blockchain

Schlechtere Skalierbarkeit durch Smart Contracts Ethereum löst die Probleme von häufig auftretenden Forks und erlaubt so einen höheren Transaktionsdurchsatz sowie schnellere Transaktionsbestätigungszeiten. Weitere Probleme entstehen jedoch, wenn eine Blockchain nicht nur Geldtransfertransaktionen verarbeitet. Ethereum erlaubt das speichern und ausführen von eigenem Code durch Smart Contracts. Dadurch steigt die Komplexität der auszuführenden Transaktionen. Dadurch nimmt die Skalierbarkeit ab, da die entstehenden größeren Blöcke eine längere Propagationszeit verursachen. Ebenfalls verschlechtert sich die Performance des Netzwerks, da die Daten schwieriger zu verarbeiten sind. So muss jede Node alle Transaktionen verifizieren, Smart Contract-Code ausführen, und die Ergebnisse speichern. [44].

In Ethereum werden Transaktionen sequentiell bei allen Nodes ausgeführt. Dazu gehört das ausführen von Smart Contract-Code sowie das verifizieren der Ergebnisse. Nur so können in Konflikt stehende Transaktionen, wie zum Beispiel beim Double-Spend) erkannt werden. Eine Parallelausführung ist nicht möglich. Dies verschlechtert letztendlich die Performance des Netzwerks, da es länger dauert Transaktionen auszuführen [44]. Dies wird auch durch ein Beispiel klar. Ein Angreifer kann DoS-Attacken ausführen, indem er komplex auszuführende Smart Contracts schreiben. Die Ausführung von diesem bei jeder Node führt dazu, dass keine anderen Operationen ausgeführt werden können. Ethereum löst dieses Problem, indem der Transaktionssender für jeden Berechnungschritt eine Gebühr zahlen muss. Dies funktioniert jedoch nur, wenn in der Blockchain-Anwendung eine Kryptowährung genutzt wird [53].

Ebenfalls behauptet Vukolic, dass der Code der Smart Contracts nicht bei allen Nodes ausgeführt werden muss. Um Konsens zu erreichen genügt es, dass alle Nodes den gleichen Stand der Daten erhalten. Deshalb könnte die Codeausführung von nur von bestimmten Nodes

ausgeführt werden. Das Problem dabei ist, dass man eine genügend große Anzahl an vertrauenswürdige Teilnehmer festlegen muss [53]. Damit geht allerdings auch das vertrauenslose Modell der Blockchain verloren.

Letztendlich lässt sich sagen, dass Public Blockchains nicht skalieren. Um dies zu lösen, müsste die Netzwerktopologie verbessert werden um schnelle Blockpropagrationszeiten zu erlauben [44]. Weitere Schwierigkeiten bestehen sobald nicht nur Geldtransferaktionen verarbeitet werden müssen. Betrachtet man das CAP-Theorem wird ersichtlich, dass nur die Eigenschaften Dezentralisierbarkeit und Sicherheit gegeben sind. Es ist jedoch zu bedenken, dass viele Probleme der Skalierbarkeit aufgrund der genutzten Konsensmechanik bestehen. Auch wenn es teilweise Lösungsvorschläge für diese gibt, genügen sie bisher nicht um Skalierbarkeit herzustellen. Deshalb gilt es, die Limitationen von Permissioned Blockchains sowie alternative Konsensmechaniken für diese zu analysieren.

5.1.3 Permissioned Blockchains

Permissioned Blockchains werden eingesetzt, wenn nur bestimmte Teilnehmer an der Blockchain teilnehmen sollen. Dadurch entsteht eine stärkere Zentralisierung als bei Public Blockchains. Bezieht man sich auf das CAP-Theorem, müssten sich dadurch die Sicherheit und/oder Skalierbarkeit verbessern. Dies führt allerdings auch dazu, dass ein größeres Maß an Vertrauen zwischen den Teilnehmern gegeben sein muss. Dies wird dadurch sichergestellt, dass jeder Teilnehmer die Rechte zur Teilnahme am Netzwerk erhalten hat und die Identitäten dieser bekannt sind. Duch letzteres ist nachverfolgbar, welche Teilnehmer welche Transaktionen ausführt [44].

Scherer behauptes, dass das größere Vertrauen es erlaubt den Nodes verschiedene Aufgaben zuzuteilen. Dies beschreibt er am Beispiel von Hyperledger Fabric, einer Permissioned-Blockchain. In dieser gibt es Peer und Ordering Nodes. Erstere übernehmen das ausführen von Code, während letztere die Reihenfolge der auszuführenden Transaktionen in den Blöcken bestimmten und dabei ebenfalls überprüfen ob es in Konflikt stehende Transaktionen gibt (Genauer im Kapitel 6.1. Die Ordering Nodes sind also letztendlich für den Konsens verantwortlich. Im Gegensatz zu Ethereum können Peer Nodes so parallel Transaktionen verarbeiten. Sie müssen sich nicht um eventuelle Konflikte oder die Reihenfolge der Transaktionen kümmern. Letztendlich würde die Skalierbarkeit, im Rahmen des Verarbeitens von Transaktionen, nur von der Hardware der Peers abhängen [44].

Scherer führt ebenfalls Tests durch um die Performance von Hyperledger Fabric zu analysieren. Dazu nutzt er eine frühe und unstabile Version 1.0. Das Netzwerk besteht aus einer Ordering und einer Peer Node. Es wird kein Konsensmechanismus genutztDie Anwendung selber unterstützt die Zahlung mittels digitaler Assets (z.B. Tokens bzw. Coins) zwischen 2 Accounts. Dabei erreicht er einen maximalen Transaktionsdurchsatz von 350 TPS. Dabei ist allerdings zu bedenken, dass der Test auf einer Maschine mit limitierten Ressourcen ausgeführt wird. Um einen maximalen Transaktionsdurchsatz zu erreichen, müssten mehrere leistungsstarke Computer für den Test eingesetzt werden. Scherer stellt ebenfalls fest, dass der Transaktionsdurchsatz

abnimmt, desto mehr Peers es gibt, welche Transaktionen bestätigen. Dies liegt daran, dass diese sogenannten Endorser untereinander kommunizieren müssen. Pro Node müssten $O(n^2)$ Nachrichten gesendet werden, wobei n die Anzahl an Nodes ist. Die Anzahl an effizient nutzbaren Endorsern ist also beschränkt. Auch hier ist jedoch zu bedenken, dass ein Test mit leistungsstarken Computern ausgeführt werden muss um die Skalierbarkeit dieser festzustellen [44].

Ein Paper von Pongnumkul vergleicht die Leistung von Hyperledger Fabric mit Ethereum. Er nutzt dazu die stabile Version 0.6. Er führt die Test ebenfalls mit nur einer Peer Node durch. Zur Ordering Node macht er keine Angabe. Die Anwendung ist die gleiche wie bei Scherer und es wird ebenfalls kein Konsensmechanismus genutzt. Pongnumkul stellt fest, dass die Performance von Fabric in allen Kriterien besser ist als bei Ethereum. So betrug die Zeit, bis eine Beispieltransaktion verarbeitet wurde bei Ethereum 41 Sekunden und bei Ethereum 478 Sekunden. Tests zum maximalen Transaktionsdurchsatz haben ergeben, dass Ethereum 40 TPS und Hyperledger Fabric 300 TPS erreicht hat. Die dazugehörige Abbildung 5.3 zeigt auch, dass die Unterschiede zwischen Ethereum und Fabric signifikanter sind, desto mehr Transaktionen verarbeitet werden müssen [41].

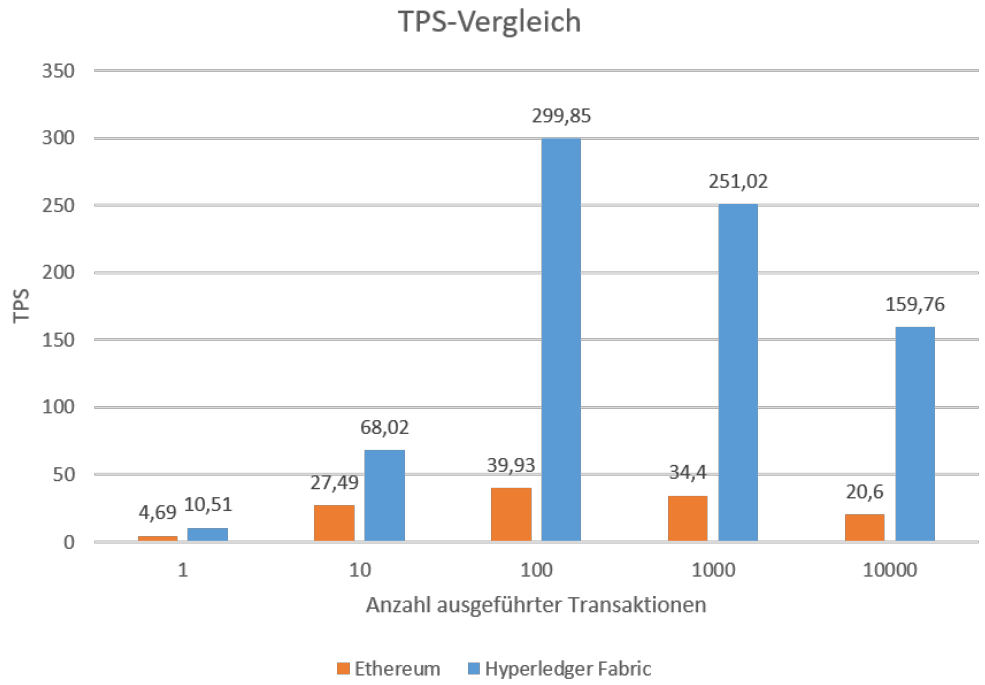

Abbildung 5.3: Vergleich des Transaktionsdurchsatzes von Ethereum und Hyperledger Fabric [41].

Bei beiden Tests ist zu bedenken, dass sie mit nicht aktuellen Versionen von Hyperledger Fabric ausgeführt wurden. Mittlerweile gibt es eine stabile Version 1.0, sowie eine Preview von Version 1.1.0 [17]. Es ist also möglich das die Performance sich mittlerweile verbessert hat.

Letztendlich lässt sich sagen, dass Permissioned Blockchains, was die Verabeitung von Transaktionen betrifft, eine bessere Performance erzielen. Damit bestätigt sich auch das CAP-

Theorem bezüglich der Skalierbarkeit. Die Performance wurde allerdings noch nicht unter der Nutzung verschiedener Konsensmechanismen betrachtet. Weiterhin muss das CAP-Theorem noch auf die Sicherheit untersucht werden. Deshalb erfolgt im nächsten Kapitel die Analyse der Skalierbarkeit und Sicherheit von Konsensmechanismen.

5.2 Konsensmechanismen

Das Erreichen von Konsens in einer Blockchain, ist eine Abwandlung des Byzantine Generals Problem. In diesem gibt es Generäle, welche Armeen kommandieren welche eine Stadt umzingeln. Ein Angriff auf diese ist nur erfolgreich, wenn alle Armeen gleichzeitig angreifen. Die Generäle müssen untereinander kommunizieren und Konsens darüber herstellen, ob ein Angriff erfolgen soll. Allerdings gibt es Verräter unter diesen. Es handelt sich also um ein vertrauensloses Umfeld. Genau so kann es in einer Blockchain zu den sogenannten Byzantine Faults kommen, wenn nicht vertrauenswürdige Nodes Daten manipulieren können. Deshalb müssen verteilte Systeme Byzantine Fault Tolerance (BFT) herstellen, um eine gewisse Anzahl an nicht vertrauenswürdigen Teilnehmern zu tolerieren. Dies geschieht in Blockchains über die Konsensmechanismen [55] [12].

Eine Blockchain, welche den Proof-of-Work als Konsensmechanismus nutzt, ist nicht skalierbar. Ebenfalls würde er in Netzwerken mit relativ wenig Teilnehmern die Sicherheit beeinträchtigen, da ein Teilnehmer einfacher 51% der Rechenleistung erreichen kann. Für Permissioned Blockchains muss also ein Konsensmechanismus gefunden werden, welche Skalierbarkeit und Sicherheit herstellt. Aufgrund des höheren Vertrauens in Permissioned Blockchains, behauptet Scherer, dass ein Konsensmechanismus genutzt werden kann, welcher Vertrauen in geringerem Maße als der Proof-of-Work garantiert. Somit könnten Skalierbarkeit und Sicherheit hergestellt werden [44]. Im folgenden werden verschiedene Konsensmechanismen verschiedener Blockchain-Technologien miteinander verglichen. Dabei ist zu bedenken, dass nur auf die Konsensmechanismen, welche Sicherheit und Skalierbarkeit in Permissioned Blockchains sicherstellen, im Detail eingegangen wird.

5.2.1 Proof of Stake

Beim PoS hängt die Wahrscheinlichkeit, einen Block zu minen, von der Menge des Eigentums (z.B. Kryptowährung) eines Nutzers ab. Desto mehr Bitcoins man beispielsweise hat, desto höher ist die Wahrscheinlichkeit für das Mining ausgewählt zu werden. Ähnlich wie beim PoW könnte ein Teilnehmer mit 51% aller Bitcoins das Netzwerk angreifen, da er die längere Blockchain erstellen kann. Aber selbst wenn es ein Teilnehmer schafft, 51% der Bitcoins zu besitzen, hätte er keine Motivation dazu. Denn letztendlich würde ein Angriff den Kurs von Bitcoin senken, und somit würde der Miner sich selber schaden [55]. Da der PoS hauptsächlich nur bei Kryptowährungen genutzt werden kann, ist er für Permissioned Blockchains uninteressant.

5.2.2 Proof of Elapsed Time

Der PoET wird in Intels Blockchain-Technologie Hyperledger Sawtooth genutzt. Die grundlegende Idee ist, dass eine Node eine zufällige Zeit generiert, welche Sie warten muss um einen Block zu erstellen. Um sicherzustellen, dass die generierte Zeit nicht verfälscht wurde, wird Trusted Computing [3] genutzt. So stellen Intels Software Guard Extensions (SGX) sicher, dass Code nicht modifiziert werden kann. Eine Node muss also über solchen unmodifizierbaren Code eine Zeit generieren. Weiterhin erfolgen statistische Tests, um zu verhindern dass eine Node Blöcke zu schnell und somit zu oft erstellt. Letztendlich ist die Blockerstellung damit fair verteilt, und kein Teilnehmer kann die Blockchain kontrollieren. Im Prinzip funktioniert der Mechanismus wie der PoW. Dort wird eine Wartezeit durch das Finden eines Hashes sichergestellt, während dies beim PoET durch die Hardware sichergestellt wird. Dadurch, dass es keine rechenintensiven Aufgaben gibt, ist die Skalierbarkeit sicher gestellt. Es ist jedoch zu bedenken, dass es bisher wenige Analysen zu der Sicherheit des PoET gibt. Ein Paper von Chen stellt fest das der Konsensmechanismus unter bestimmten Umständen unsicher ist, schlägt aber auch Lösungen dafür vor. Ebenfalls kommt hinzu, dass man der Hardware von Intel für das Trusted Computing vertrauen muss. [24].

5.2.3 Practical Byzantine Fault Tolerance

Der PBFT gehört zu der Familie der BFT-Protokolle. Beim PBFT wählen stimmen die Teilnehmer für eine Node, welcher die auszuführenden Transaktionen in einen neuen Block festlegt. Dieser wird an alle anderen Nodes weitergeleitet. Anschließend wird der Konsens hergestellt. 2/3 der Nodes müssen dem Block zustimmen, damit er gültig ist. Erst dann werden die Transaktionen bei jeden Teilnehmer ausgeführt. Deshalb können bis zu 1/3 der Nodes unvertrauenswürdig sein. Ein Angreifer müsste für einen Angriff die Kontrolle über 2/3 der Nodes haben [48][55].

Vukolic behauptet, dass es BFT-Protokolle gibt, welche einen Transaktionsdurchsatz von mehreren 10000 TPS unterstützen. Die Skalierbarkeit dieser bezüglich der Anzahl an Nodes ist jedoch begrenzt [52]. Croman erzielt bei seinen Tests mit dem PBFT, bei 8 Nodes und 8192 auszuführenden Transaktionen einen Transaktionsdurchsatz von 14000 TPS. Weiterhin wird ersichtlich wie die Performance mit der Anzahl an Nodes abnimmt. Mit 64 Nodes und 8192 auszuführenden Transaktionen wird ein Transaktionsdurchsatz von 4500 TPS erreicht [25]. Im Gegensatz zum PoW besteht hier eine bessere Skalierbarkeit bezüglich des Transaktionsdurchsatzes, allerding ist sie bezüglich der Anzahl an Teilnehmern begrenzt [52].

Ein weiterer Vorteil von BFT-Protokollen ist, dass es Consensus Finality gibt. Das bedeutet, dass es nicht zu Forks kommen kann. Somit müssten Nutzer nicht darauf warten, dass mehrere Blöcke nach einer Transaktion erstellt werden, damit die Sicherheit gegeben ist, dass diese endgültig bestehen wird. Somit entfällt auch die Gefahr von Double-Spend-Attacken [52].

[3] Trusted Computing: Soft- und Hardware stellen sicher, dass ein Computer sich wie erwartet verhält [18]

5.2.4 Tendermint

5.2.5 Leader Based Consensus

5.2.6 Federated Byzantine Agreement

5.2.7 Diversity Mining Consensus

5.2.8 Stellar Consensus

5.2.9 Raft

5.2.10 BFT-SMaRt

5.3 Sonstige

5.3.1 Private Transaktionen

5.3.2 Code Execution

5.3.3 Datenmenge

Kapitel 6

Dezentraler Wartungsmarkt - Prototyp

6.1 Technologieauswahl

- Welche Blockchain wird genutzt um die Anforderungen zu erfüllen ?

6.2 Hyperledger Fabric und Composer

6.2.1 Hyperledger Fabric

6.2.2 Hyperledger Composer

6.3 Modell

- Architekturen, Sequenzdiagramme, Workflows etc.
- Datenmodell (Participants, Assets, Transaktionen)
- Netzwerkarchitektur

6.4 Gerätesimulation durch Bosch XDK

- Simulation eines IOT-Geräts durch einen Bosch XDK

6.5 Programmlogik

- Funktion der Transaktionen

6.6 Benutzeroberflächen

- UIs für die Interaktion mit der Blockchain

6.7 Konsensmechanismus

6.8 Evaluierung

- Analyse des Systems in Bezug auf Anforderungen und Blockchain-Probleme

Kapitel 7

Fazit und Ausblick

- Kurze Zusammenfassung
- Ausblick geben/Erweiterbarkeit des Systems beschreiben
- Ausblick zu Problemen von B2B-Blockchains geben

Literaturverzeichnis

[1] Access Control Language | Hyperledger Composer. https://hyperledger.github.io/composer/unstable/reference/acl_language.

[2] Bitcoin Energy Consumption Index. https://digiconomist.net/bitcoin-energy-consumption.

[3] BitcoinStats. http://bitcoinstats.com/network/propagation/.

[4] Blockchain Bikes. http://futurefluxfestival.nl/en/program/blockchain-bikes/.

[5] Blockchain Wallet. https://wirexapp.com/guides/mobile-wallets/blockchain-wallet/.

[6] Ethereum Average BlockTime Chart. https://etherscan.io/chart/blocktime.

[7] Ethereum Network HashRate Growth Chart. https://etherscan.io/chart/hashrate.

[8] Glossar - Bitcoin. https://bitcoin.org/de/glossar.

[9] Hash Rate. https://blockchain.info/hash-rate.

[10] Scalability - Bitcoin Wiki. https://en.bitcoin.it/wiki/Scalability.

[11] Single Point of Failure. *Wikipedia*, July 2016. Page Version ID: 156306981.

[12] Byzantine Fault Tolerance: The Key for Blockchains. http://www.nasdaq.com/article/byzantine-fault-tolerance-the-key-for-blockchains-cm810058, June 2017.

[13] Ethereum White Paper, December 2017.

[14] Kryptologische Hashfunktion. *Wikipedia*, November 2017. Page Version ID: 170625494.

[15] Nonce. *Wikipedia*, August 2017. Page Version ID: 167799632.

[16] Programmierschnittstelle. *Wikipedia*, November 2017. Page Version ID: 170840602.

[17] Hyperledger Fabric Releases, January 2018.

[18] Trusted Computing. *Wikipedia*, January 2018. Page Version ID: 819361025.

[19] Andreas M. Antonopoulos. *Mastering Bitcoin*. O'Reilly, Sebastopol CA, first edition edition, 2015. OCLC: ocn876351095.

[20] Elyes Ben Hamida, Kei Leo Brousmiche, Hugo Levard, and Eric Thea. Blockchain for Enterprise: Overview, Opportunities and Challenges. In *The Thirteenth International Conference on Wireless and Mobile Communications (ICWMC 2017)*, Nice, France, July 2017.

[21] Steven Buchko. How Long Do Bitcoin Transactions Take? https://coincentral.com/how-long-do-bitcoin-transfers-take/, December 2017.

[22] Vitalik Buterin. Toward a 12-second Block Time, July 2014.

[23] Amy Castor. An Ethereum Voting Scheme That Doesn't Give Away Your Vote. https://www.coindesk.com/voting-scheme-ethereum-doesnt-give-away-vote/, April 2017.

[24] Lin Chen, Lei Xu, Nolan Shah, Zhimin Gao, Yang Lu, and Weidong Shi. On Security Analysis of Proof-of-Elapsed-Time (PoET). In *International Symposium on Stabilization, Safety, and Security of Distributed Systems*, pages 282–297. Springer, 2017.

[25] Kyle Croman, Christian Decker, Ittay Eyal, Adem Efe Gencer, Ari Juels, Ahmed Kosba, Andrew Miller, Prateek Saxena, Elaine Shi, Emin Gün Sirer, Dawn Song, and Roger Wattenhofer. On Scaling Decentralized Blockchains. In *Financial Cryptography and Data Security*, Lecture Notes in Computer Science, pages 106–125. Springer, Berlin, Heidelberg, February 2016.

[26] Michael Crosby. BlockChain Technology: Beyond Bitcoin. Technical report, 2016.

[27] Stefano De Angelis, Leonardo Aniello, Roberto Baldoni, Federico Lombardi, Andrea Margheri, and Vladimiro Sassone. PBFT vs proof-of-authority: Applying the CAP theorem to permissioned blockchain. 2017.

[28] Christian Decker and Roger Wattenhofer. Information propagation in the bitcoin network. In *Peer-to-Peer Computing (P2P), 2013 IEEE Thirteenth International Conference On*, pages 1–10. IEEE, 2013.

[29] Blockchain (Firma). Blockchain Charts: Unbestätigte Transaktionen. https://blockchain.info/de/unconfirmed-transactions.

[30] Andreas Fischer. Das IoT in der Blockchain. https://www.com-magazin.de/praxis/internet-dinge/iot-in-blockchain-1228562.html.

[31] Patrick Götze. Lufthansa Industry Solutions - Mit Blockchain zu mehr Transparenz in der Luftfahrt. https://www.lufthansa-industry-solutions.com/de-de/loesungen-produkte/luftfahrt/mit-blockchain-zu-mehr-transparenz-in-der-luftfahrt/.

[32] Vincent Gramoli. On the danger of private blockchains. In *Workshop on Distributed Cryptocurrencies and Consensus Ledgers (DCCL'16)*, 2016.

[33] Blockchain Hub. Blockchains & Distributed Ledger Technologies.

[34] Ido Kaiser. A Decentralized Private Marketplace: DRAFT 0.1.

[35] Kari Korpela, Jukka Hallikas, and Tomi Dahlberg. Digital Supply Chain Transformation toward Blockchain Integration. January 2017.

[36] Winfried Krieger. Definition » Supply Chain Management (SCM) « | Gabler Wirtschaftslexikon. http://wirtschaftslexikon.gabler.de/Definition/supply-chain-management-scm.html.

[37] Wenting Li, Alessandro Sforzin, Sergey Fedorov, and Ghassan O. Karame. Towards Scalable and Private Industrial Blockchains. In *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*, BCC '17, pages 9–14, New York, NY, USA, 2017. ACM.

[38] What is Bitcoin Mining? Learn about Bitcoin mining hardware. https://www.bitcoinmining.com/bitcoin-mining-hardware/.

[39] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. 2008.

[40] NetEDI. How API's are shaping B2B Data Integration- NetEDI®, September 2017.

[41] S. Pongnumkul, C. Siripanpornchana, and S. Thajchayapong. Performance Analysis of Private Blockchain Platforms in Varying Workloads. In *2017 26th International Conference on Computer Communication and Networks (ICCCN)*, pages 1–6, July 2017.

[42] Mercury Protocol. How To: Create Your Own Private Ethereum Blockchain, December 2017.

[43] Sirajd Raval. *Decentralized Applications: Harnessing Bitcoin's Blockchain Technology*. O'Reilly Media, Inc., 2016.

[44] Mattias Scherer. *Performance and Scalability of Blockchain Networks and Smart Contracts*. 2017.

[45] Jagdeep Sidhu. Syscoin: A Peer-to-Peer Electronic Cash System with Blockchain-Based Services for E-Business. In *Computer Communication and Networks (ICCCN), 2017 26th International Conference On*, pages 1–6. IEEE, 2017.

[46] John Soldatos. What Does Blockchain Technology Have to Do with Enterprise Maintenance Activities? https://www.solufy.com/blog/what-does-blockchain-technology-have-to-do-with-enterprise-maintenance-activities.

[47] Yonatan Sompolinsky and Aviv Zohar. Accelerating Bitcoin's Transaction Processing. Fast Money Grows on Trees, Not Chains. Technical Report 881, 2013.

[48] H. Sukhwani, J. M. Martínez, X. Chang, K. S. Trivedi, and A. Rindos. Performance Modeling of PBFT Consensus Process for Permissioned Blockchain Network (Hyperledger Fabric). In *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*, pages 253–255, September 2017.

[49] Melanie Swan. *Blockchain: Blueprint for a New Economy*. O'Reilly, Beijing : Sebastopol, CA, first edition edition, 2015. OCLC: ocn898924255.

[50] Don Tapscott and Alex Tapscott. *Die Blockchain-Revolution: Wie die Technologie hinter Bitcoin nicht nur das Finanzsystem, sondern die ganze Welt verändert*. Plassen Verlag, Kulmbach, 1 edition, October 2016.

[51] Hyperledger Fabric Team. Hyperledger Whitepaper. https://docs.google.com/document/d/1Z4M_qwILLRehPbVRUsJ3OF8Iir-gqS-ZYe7W-LE9gnE/edit?usp=embed_facebook, 2016.

[52] Marko Vukolić. The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication. In *International Workshop on Open Problems in Network Security*, pages 112–125. Springer, 2015.

[53] Marko Vukolić. Rethinking Permissioned Blockchains. pages 3–7. ACM Press, 2017.

[54] Karl Wüst and Arthur Gervais. Do you need a Blockchain? Technical Report 375, 2017.

[55] Zibin Zheng, Shaoan Xie, Hong-Ning Dai, Xiangping Chen, and Huaimin Wang. Blockchain Challenges and Opportunities: A Survey. *International Journal of Web and Grid Services*, December 2017.