Blockchain

als Grundlage

disruptiver Geschäftsmodelle

Hausarbeit

im Studiengang Wirtschaftsinformatik

Modul
Mobile Anwendungen und Systeme

vorgelegt von

Marius Golombeck
magol003@stud.fh-dortmund.de
Mat. Nr.: 7092542

am 08.01.2018
an der Fachhochschule Dortmund

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

CBC	cipher block chain
XOR	exclusive or
P2P	peer-to-peer
MDC	modification detection codes
MAC	message authentication codes
OWHF	one-way hash function
CRHF	collision resistant hash function
FIPS	Federal Information Processing Standard
ECDSA	Elliptic Curve Digital Signature Algorithm
ECC	elliptic curve cryptography
SEC	Standards for Efficient Cryptography
PoW	Proof of Work
PoS	Proof of Stake
ICO	Initial Coin Offering
IPO	Initial Public Offering
XRP	Ripple
SCP	Stellar Consensus Protocol

1 Einleitung

Die Blockchain Technologie gilt als eine der vielversprechendsten Zukunftstechnologien unserer Zeit [1]. Sie verspricht eine fundamentale Revolution in der Art und Weise wie Informationen ausgetauscht, gespeichert und archiviert werden können [2].

Bei einer Blockchain handelt es sich um eine effiziente, dezentrale Datenstruktur, bestehend aus chronologisch geordneten Datenblöcken. Diese Blöcke werden mit Hilfe von kryptographischen Verfahren verkettet. Hierbei entsteht eine lückenlos geführte, transparente, unveränderliche Kette von Informationen, die problemlos von allen beteiligten Parteien nachvollzogen werden kann [3].

Durch Nutzung einer Blockchain als Basis in der Informationsinfrastruktur von Unternehmen entstehen Möglichkeiten zur Disruption traditioneller Geschäftsmodelle [1].

1.1 Problemstellung / Ziel der Arbeit

In dieser Arbeit gilt es einleitend zu klären, wie sich die technische Funktionsweise einer Blockchain gestaltet.

Danach gilt es das Konzept der Smart Contracts vorzustellen.

Anschließend werden Umsetzungsmöglichkeiten dieser Technologie vorgestellt und mögliche Disruptionspotenziale hergeleitet.

2 Blockchain Grundlagen

Die Übersetzung des Begriffes Blockchain lässt mehr auf die Funktionsweise schließen, als zu erahnen ist. Es handelt sich dabei wortwörtlich um eine Kette aus Blöcken.

Eine einheitliche Definition existiert jedoch nicht. Es bietet sich also an, die Bedeutung des Wortes zurückzuverfolgen, um so ein umfassendes Begriffsverständnis zu erlangen [4].

Bereits im Jahr 1976 taucht der Begriff block chaining in einem Patentantrag von William F. Ehrsam, Carl H. W. Meyer, John L. Smith und Walter L. Tuchman mit dem Titel „Message verification and transmission error detection by block chaining“ auf [5].

Die Erfinder beschreiben in ihrem Patent ein Verfahren zur sicheren Übertragung von Nachrichten durch deren Verkettung in Blöcken. Dieses Verfahren trägt den Namen Cipher Block Chaining (CBC) [5].

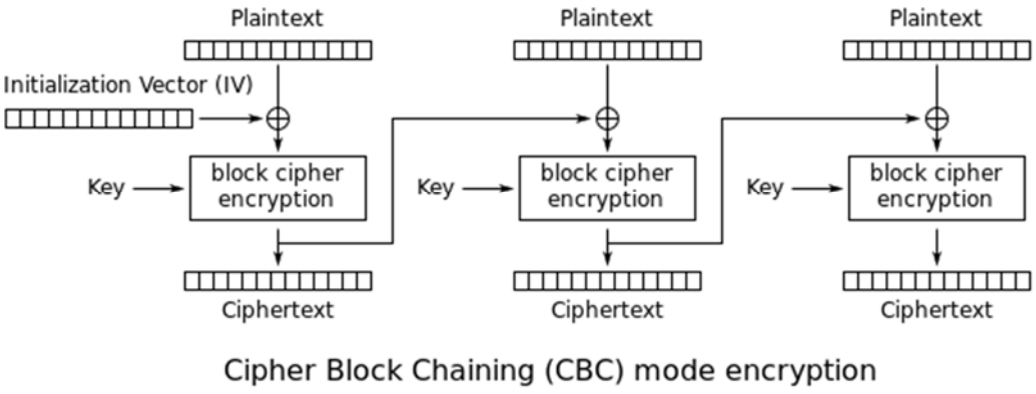

Abbildung 1 - Verschlüsselung mit Hilfe des CBC Verfahrens [6]

In diesem Verfahren werden Blöcke mit Klartext durch einen XOR-Operator miteinander verknüpft und im Anschluss verschlüsselt [7].

Die fortlaufende Anwendung des Verfahrens bildet so eine Kette aus logisch voneinander abhängigen Blöcken. Zur eindeutigen Identifikation jeder Kette, wird bei der Erstellung des ersten Blocks ein einzigartiger Initialisierungsvektor verwendet [5].

Dass dieses grundlegende kryptographische Verfahren erheblichen Einfluss auf das heutige technische Verständnis einer Blockchain haben wird, zeigt sich im Verlauf der Ausarbeitung.

Die ersten Bezeichnungen einer Ereignisverarbeitung von Blöcken mit digitalen Signaturen als Blockchain fanden sich erst über 20 Jahre später, im Jahr 1997, bei Diskussionen über HashCash[1], wie aus den Cypherpunk Cryptography Mailing Lists hervorgeht [4].

Im Jahr 2005 wiederum, beschrieb Nick Szabo, in der Ausarbeitung zu seiner Kryptowährung Bit Gold ein dezentrales System von proof-of-work chains, deren Bestandteile mit Hilfe von öffentlichen Schlüsseln, Zeitstempeln und digitalen Signaturen verknüpft wurden [8].

Satoshi Nakamoto greift das Konzept der proof-of-work chains [9] bei der Implementierung eines verteilten Zeitstempelservers auf peer-to-peer-Basis [10] für seine Kryptowährung Bitcoin im Jahr 2008 auf [11].

Die cost-function von HashCash wird als Mining Funktion[2] der Kryptowährung Bitcoin adaptiert [11]. Der Quellcode und die Sicherheitsprotokolle wurden gemeinsam von Hal Finney und Satoshi Nakamoto entwickelt [12].

Finney, der quasi durch Zufall auf die Arbeiten von Nakamoto stieß, leistete durch seine Programmierarbeit einen wesentlichen Beitrag zur Funktionsweise von Bitcoin

[1] HashCash wurde von Adam Back ursprünglich als proof-of-work Algorithmus entwickelt um denial-of-service Attacken zu bekämpfen [70].

[2] Hier: bezogen auf die Transaktionsverarbeitung, eine Erklärung folgt in Kapitel 2.1.4

und ist bis heute neben Nakamoto der einzige namentlich bekannte Entwickler von Bitcoin in seiner Urform [13].

Obwohl Nakamoto in seinem Paper zu Bitcoin nicht einmal die Terminologie einer Blockchain verwandte, wurde die von ihm entwickelte Datenstruktur in E-Mails zwischen Nakamoto und Finney im Folgenden wiederholt als Blockchain bezeichnet, was den Begriff nachhaltig prägen sollte [14].

Im Hinblick auf die historischen Gegebenheiten lässt sich folgende Definition bilden.

Definition[3]**:** Eine Blockchain ist eine Art Datenbank, deren Daten in Blöcken gespeichert und mit Hilfe von kryptographischen Methoden zu einer Kette verknüpft werden.

[3] In Anlehnung an [69]

2.1 Technische Funktionsweise

Die technische Funktionsweise einer Blockchain allgemeingültig zu erklären gelingt nur, indem man sich auf ihre grundlegenden Konzepte stützt.

Hierzu gehören neben kryptographischen Verfahren wie Hashing und digitalen Signaturen auch ein (P2P) Netzwerk mit verteiltem Konsens [2].

Konzept	Funktion
Kryptographische Hashfunktion	Identifikation Integrität
Digitale Signaturverfahren	Authentifikation Nachweisbarkeit Vertraulichkeit
P2P Netzwerk	System of Record[4] Verfügbarkeit Buchführung der Transaktionen
Verteilter Konsens	Aktualität Einheitlicher Datenstand

Tabelle 1 - Konzepte und ihre Funktionen in einer Blockchain[5]

Außerdem ist es möglich drei Kernanforderungen an eine Blockchain festzulegen, welche die Vorteile, die sich aus dieser Technologie ergeben, garantieren soll [15].

1. Integrität
2. Vertraulichkeit
3. Verfügbarkeit

[4] Vgl. "A system of record is the authoritative data source for a given data element or piece of information." [71]

[5] (eigene Darstellung)

Im Verlauf der nächsten Abschnitte 2.1.1 bis 2.1.4 werden die drei o.g. Konzepte, sowie die Kernanforderungen erläutert und in den Kontext eingeordnet.

2.1.1 Kryptographische Hashfunktion

Eine Hash Funktion h bildet einen Input x auf einen Hashwert z ab, sodass gilt:

$$h: x \rightarrow z$$

Diese Funktion muss zumindest die folgenden zwei Eigenschaften erfüllen [16].

1. Kompression *(compression)*: Die Funktion h bildet einen Input x beliebiger Länge auf einen Hashwert z fester Länge ab, sodass gilt: $h: \{0,1\}^* \rightarrow \{0,1\}^n$
2. Einfachheit der Berechnung *(ease of computation)*: Seien die Funktion h und der Input x gegeben, so ist z mit geringem Aufwand zu berechnen.

Hashfunktionen lassen sich prinzipiell in die Kategorien Schlüsselbehaftet *(keyed)* und Nicht-Schlüsselbehaftet *(unkeyed)* einteilen [16].

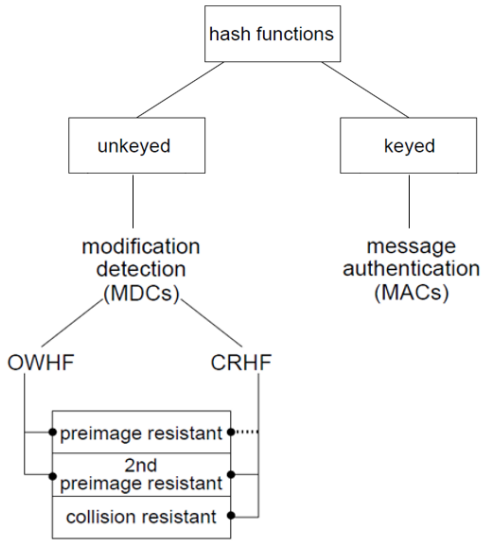

Abbildung 2 - Vereinfachte Darstellung der Klassifikation von Hash Funktionen[6]

[6] In Anlehnung an [16]

- MAC Hashfunktionen haben als Eingabeparameter den zu kodierenden Input x sowie darüber hinaus einen privaten Schlüssel k , sodass gilt: $h(x,k) = z$
- MDC Hashfunktionen haben als Eingabeparameter lediglich den zu kodierenden Input x, sodass gilt: $h(x) = z$.

  MDC Hashfunktionen lassen sich in zwei weitere Kategorien unterteilen [16]:

  - Ein-Wege Hashfunktionen (OWHF, *one way hash functions*) [17]
  - Kollisionsresistente Hashfunktionen (CRHF, *collision resistant hash functions*) [18]

Für die o.g. MDC Hashfunktionen lassen neben den Eigenschaften der Kompression und der Einfachheit der Berechnung drei weitere Eigenschaften festlegen [16] [19].

3. Urbildbeständigkeit (OWHF, CRHF): Eine Funktion h ist urbildbeständig, wenn es schwierig[7] ist, für sie einen gegebenen Hashwert z, mit einem korrelierenden Inputwert x' zu finden, sodass gilt: $h(x') = z$
4. Abbildbeständigkeit (OWHF, CRHF): Eine Funktion h ist abbildbeständig, wenn es schwierig ist, zu einem gegebenen Input x einen zweiten Input x' zu finden, sodass für alle $x \neq x'$gilt: $h(x) = h(x')$
5. Kollisionsresistenz (CRHF): Eine Funktion h ist kollisionsresistent, wenn es schwierig ist, zwei Inputwerte $x \; und \; x'$ zu finden, sodass gilt: $h(x) = h(x')$

Hash Funktionen können in vielen Gebieten zum Einsatz kommen, wie zum Beispiel in digitalen Signaturverfahren, als Prüfsummen oder in Datenbanken [20].

Kryptographische Hashfunktionen sind Hash Funktionen, die zumindest über die fünf o.g. Eigenschaften verfügen [16].

[7] Hier: nicht in polynomieller Zeit lösbar

Darüber hinaus sollten gute kryptographische Hashfunktionen die folgenden drei Eigenschaften besitzen, um die Sicherheit weiter zu erhöhen [16].

- Korrelationsfreiheit *(non correlation)*: Input Bits und Output Bits sollten nicht miteinander korrelieren. Idealerweise sollte ein sog. Lawineneffekt *(avalanche property)* erreicht werden, bei dem eine Änderung an nur einem Input Bit eine mehrheitliche Änderung der Output Bits nach sich zieht.
- Kollisionsnäheresistenz *(near-collision resistance)*: Es soll schwierig sein, zwei Inputs x und x' zu finden, für die $h(x)$ und $h(x')$ sich nur in wenigen Bits unterscheiden.
- Partielle Urbildresistenz *(partial-preimage resistance* bzw. *local one-wayness)*: Es soll genau so schwierig sein einen Teil des Inputs zu finden, wie den gesamten Input. Des Weiteren soll es selbst bei Kenntnis über einige Teile des Inputs schwierig sein, die verbleibenden Teile zu finden.

Betrachten wir nun den weit verbreiteten SHA-256 Algorithmus [21]. Ein deterministischer Hashing Algorithmus, der sämtliche vorgehend definierten Eigenschaften besitzt. Er wurde durch den Federal Information Processing Standard (FIPS) für die Nutzung auf bundesstaatlicher Ebene der Vereinigten Staaten von Amerika zertifiziert und wird weitreichend in der Industrie eingesetzt [21].

Wie der Name bereits vermuten lässt, sind die Hashes die mit SHA-256 erzeugt werden 256 Bits lang [22].

Erzeugen wir zum Beispiel einen Hashwert w aus dem Text „Hello World“:

$$h_{SHA256}: w_1 \mapsto z_1, \text{mit } w_1 = Hello\ World$$

$$z_1 = a591a6d40bf420404a011733cfb7b190d62c65bf0bcda32b57b277d9ad9f146e$$

Dieser Hash repräsentiert die folgenden 32 Bytes:

$$a5\ 91\ a6\ d4\ 0b\ f4\ 20\ 40\ 4a\ 01\ 17\ 33\ cf\ b7\ b1\ 90$$

$$d6\ 2c\ 65\ bf\ 0b\ cd\ a3\ 2b\ 57\ b2\ 77\ d9\ ad\ 9f\ 14\ 6e$$

Sowie die folgenden 256 Bits:

10100101 10010001 10100110 11010100 00001011 11110100 00100000 01000000
01001010 00000001 00010111 00110011 11001111 10110111 10110001 10010000
11010110 00101100 01100101 10111111 00001011 11001101 10100011 00101011
01010111 10110010 01110111 11011001 10101101 10011111 00010100 01101110

Prüfen wir nun unser Hashing Verfahren mit einem zweiten Beispiel auf Korrelationsfreiheit - speziell auf die Existenz des Lawineneffekts.

Ändern wir den Input String $Hello\ World$ so gering wie möglich, indem wir das letzte Bit von 0 auf 1 kippen, ergibt sich daraus der String $Hello\ Worle$. Wenn wir diesen String nun hashen resultiert dies in:

$$h_{SHA256}: w_2 \mapsto z_2, \text{mit } w_2 = Hello\ Worle$$

$$z_2 = 03eea90a6021267e951388b7c5bf2873a6c12e38a4c9671fa3c4a52db6c767b8$$

Wir stellen bereits auf den ersten Blick fest, dass sich dieser Hash grundlegend von unserem Ausgangshash unterscheidet, obwohl sich der Input nur um ein Bit geändert hat. Doch wie stark sich die Hashes wirklich voneinander unterscheiden, lässt sich nur in einem Bitvergleich feststellen.

Stellen wir den 2. Hash nun wieder als seine 256 Bits dar, so ergibt sich:

00000011 11101110 10101001 00001010 01100000 00100001 00100110 01111110
10010101 00010011 10001000 10110111 11000101 10111111 00101000 01110011
10100110 11000001 00101110 00111000 10100100 11001001 01100111 00011111
10100011 11000100 10100101 00101101 10110110 11000111 01100111 10111000

Tatsächlich unterscheiden sich die zwei 256-Bit großen Hashwerte in 133 Bits, was einer Differenz der Hashes von $\frac{133}{256} \approx 51{,}95\%$ entspricht.

Wir haben also für unser Beispiel eine mehrheitliche Änderung aller Output Bits bei einer Änderung von nur einem Input Bit nachgewiesen.

Der für die Veranschaulichung notwendige Code findet sich im Anhang unter Abbildung 9 bis Abbildung 11.

Grundsätzlich empfiehlt sich Hashing überall dort, wo ein schnell zu erstellender, eindeutiger Identifizierer für Daten benötigt wird [20].

So wird Hashing auch in Blockchains dazu genutzt, Transaktionen eindeutig zu beschreiben. Eine gute kryptographische Hashfunktion, die alle Kernprinzipien von kryptographischen Hashfunktionen beachtet, stellt ein Kernelement von Blockchain Technologie dar und erfüllt die Anforderung an Integrität, die in 2.1 aufgestellt wurde [2] [23].

2.1.2 Digitale Signatur

Eine Signatur ist ein Zeichen, welches eigenhändig unter ein Schriftstück gesetzt wird. Zu meist nutzen wir sie in Form einer Unterschrift, um den Inhalt eines Schriftstücks zu bestätigen oder unser Einverständnis zu geben [24].

Das Ziel von digitalen Signaturen ist vergleichbar. Sie bestätigen, dass eine Nachricht vom Sender anerkannt und während der Transaktion nicht verändert wurde. Außerdem garantieren sie, dass der Absender ihr Absenden nicht leugnen kann.

Digitale Signaturverfahren sollten drei grundsätzlichen Paradigmen folgen [25]:

1. Authentizität (authentication): Die Unterschrift ist nur vom Sender (Unterzeichner) zu leisten. Der Empfänger, bzw. ein Dritter, ist nicht in der Lage die Unterschrift zu fälschen.

2. Integrität (integrity): Die Unterschrift ist mit dem zugehörigen Dokument untrennbar verbunden.
3. Beweisbarkeit (non-repudiability): Der Empfänger kann beweisen, dass die Unterschrift von A stammt.

Digitale Signaturverfahren nutzen das Prinzip der asymmetrischen Verschlüsselung[8], jedoch gilt es sie klar davon abzugrenzen [26].

Die Theorie der asymmetrischen Verschlüsselung wurde von Whitfield Diffie und Martin E. Hellman begründet und ist als Diffie-Hellman-Schlüsseltausch bekannt [27].

Asymmetrische Verschlüsselungsverfahren basieren grundsätzlich auf einem Schlüsseltausch mit einem Schlüsselpaar, bestehend aus öffentlichem und privatem Schlüssel [26].

Asymmetrische Verschlüsselungssysteme und Digitale Signaturverfahren implementieren die folgenden Kernfunktionen [26]:

- Asymmetrische Verschlüsselungssysteme[9]
    - Verschlüsseln: Benötigt eine Nachricht und einen öffentlichen Schlüssel als Eingabeparameter und gibt einen Wert in einem gegebenen Bereich von möglichen Werten zurück
    - Entschlüsseln: Benötigt eine verschlüsselte Nachricht und einen privaten Schlüssel als Eingabeparameter und reproduziert damit die originale Nachricht (Bei Verwendung des korrekten privaten Schlüssels)

[8] Auch als Public Key Kryptographie Verfahren bekannt

[9] Im Anhang findet sich in Abbildung 12, Abbildung 13 und Abbildung 14 ein Code-Beispiel zur Generierung eines Schlüsselpaars sowie zur Ver- und Entschlüsselung

- Digitale Signaturverfahren[10]
    - Signieren: Benötigt eine Nachricht und einen privaten Schlüssel als Eingabeparameter und gibt einen Wert in einem gegebenen Bereich von möglichen Werten zurück
    - Verifizieren: Benötigt eine Nachricht und eine Signatur und liefert einen booleschen Wert zurück („wahr“ wenn verifiziert, sonst „falsch“)

Zur Erläuterung wird im Folgenden das von Ronald L. Rivest, Adi Shamir und Leonard Adleman im Jahr 1977 entwickelte RSA Verfahren verwendet [28].

Abbildung 3 zeigt den Prozess der digitalen Signatur. Betrachten wir zwei Parteien namens Alice und Bob.

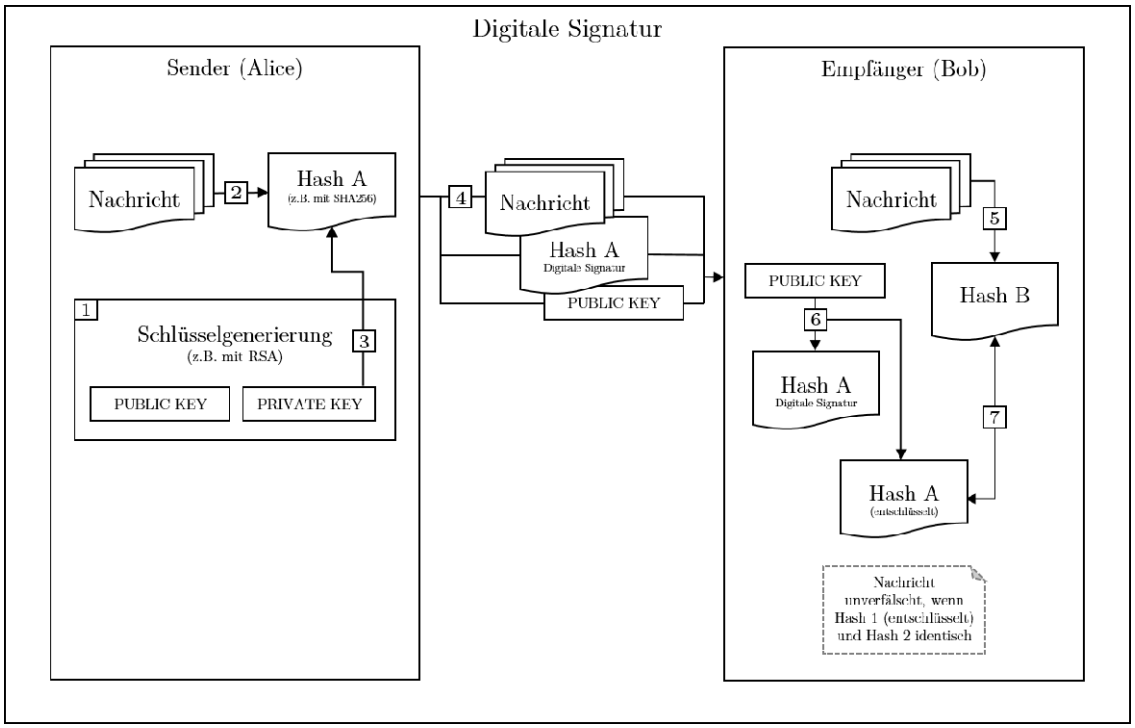

Abbildung 3 - Ablauf des Prozesses digitaler Signatur[11]

[10] Im Anhang findet sich in Abbildung 15 und Abbildung 16 ein Code-Beispiel zur Signatur und Verifizierung

[11] (eigene Darstellung)

Zuerst wählt Alice das Verschlüsselungsverfahren und generiert ihr Schlüsselpaar (1). Danach hasht sie die zu übertragende Nachricht[12] mit einem Hashingverfahren ihrer Wahl (zum Beispiel mit dem in 2.1.1 vorgestellten SHA-256 Verfahren) (2). Alice nutzt nun ihren privaten Schlüssel, um den soeben erzeugten Hash zu verschlüsseln und erzeugt damit die digitale Signatur dieser Nachricht (3).

Man beachte hier, dass Alice nicht die Nachricht selbst, sondern den Hash ihrer Nachricht signiert.

Gehe man zum Beispiel von einer Verwendung einer 2048 Bit RSA Verschlüsselung aus, so ist diese auch nur in der Lage Nachrichten bis 2047 Bit zu verarbeiten.

Da aber auch die Übertragung von größeren Daten sicher gewährleistet werden soll, wird die Nachricht gehasht, um sie durch die feste Größe des Hashes problemlos signieren zu können. (Es wäre auch möglich die Nachricht in verarbeitungsfähige Blöcke aufzuteilen, was allerdings offensichtliche Performancenachteile mit sich bringt).

Im Anschluss übersendet Alice die Nachricht, die Signatur und ihren öffentlichen Schlüssel (4). Bob hasht nun die Nachricht erneut (5) und entschlüsselt den übermittelten Hashwert mit Hilfe des öffentlichen Schlüssels von Alice (6).

Zuletzt vergleicht Bob den soeben neu erzeugten Hash B (5) und den entschlüsselten Hash A (6) und vergleicht diese miteinander (7).

Wenn beide Hashwerte identisch sind, kann Bob sicher sein, dass ihn die Nachricht von Alice unverändert erreicht hat.

[12] Es kann sich hier um beliebige Daten handeln, aus Illustrationszwecken wird hier eine einfache Nachricht verwandt

Das RSA Konzept basiert auf der Fakturierung von großen Zahlen, was bedeutet, dass der Grad der Sicherheit mit der Größe der Zahlen skaliert [29]. Doch hier liegt auch ein fundamentaler Nachteil von RSA, die Größe der Schlüssel im Verhältnis zum Grad der Sicherheit [29].

Hier bietet es sich an das RSA Verfahren mit dem besonders in Blockchains verwendeten Elliptic Curve Digital Signature Algorithm (ECDSA) Verfahren zu vergleichen [22]. Dieses Verfahren nutzt elliptische Kurvenkryptographie (ECC), um digitale Signaturen zu generieren und zu verifizieren [30].

Hierbei unterscheidet sich die ECC Schlüssellänge bei äquivalenter Sicherheit gravierend von der RSA Schlüssellänge [29].

Sicherheitslevel	RSA Schlüssellänge	ECC Schlüssellänge
80	1024 Bit	>= 160 Bit
112	2048 Bit	>= 224 Bit
128	3072 Bit	>= 256 Bit
192	8192 Bit	>= 384 Bit
256	15360 Bit	>= 512 Bit

Tabelle 2 - Äquivalente Schlüssellängen für RSA und ECC Verfahren

(In Anlehnung an [29])

Durch die vergleichsweise kleinen Schlüssel empfiehlt sich der Einsatz von ECC Verfahren besonders in Systemen mit beschränkten Ressourcen - seien sie beschränkt in Rechenleistung, Speicherplatz, Bandbreite oder Energieverbrauch [29].

Es gilt zu beachten, dass digitale Signaturen nicht verhindern, dass Transaktionsinhalte verändert werden. Hierzu ist zusätzliche Verschlüsselungstechnik notwendig [31].

Mit Hilfe von Digitale Signaturen lassen sich Transaktionen und ihre beteiligten Parteien eindeutig identifizieren. Durch die Wahrung ihrer drei Prinzipien Authentizität, Integrität und Beweisbarkeit wird die Basis für vertrauliche Kommunikation in einer Blockchain geschaffen.

2.1.3 P2P Netzwerk

Netzwerke lassen sich laut Baran [32], egal wie sie inhaltlich konstruiert sind, immer in drei grundlegende Netzwerkarchitekturkategorien einteilen. [32]

1. Zentralisierte Netzwerke (A)
    - Ein zentraler Knoten bildet den Kommunikationsmittelpunkt für alle angeschlossenen Knoten
2. Dezentralisierte Netzwerke (B)
    - Knoten unterliegen keinem festen Schema, sondern sind untereinander (ggf. über mehrere Knoten verzweigt) verbunden.
    - Knoten verteilen anfallende Arbeit an Unterknoten
3. Verteile Netzwerke (C)
    - Knoten sind gleichwertig und durchgängig mit jedem anderen Knoten verbunden

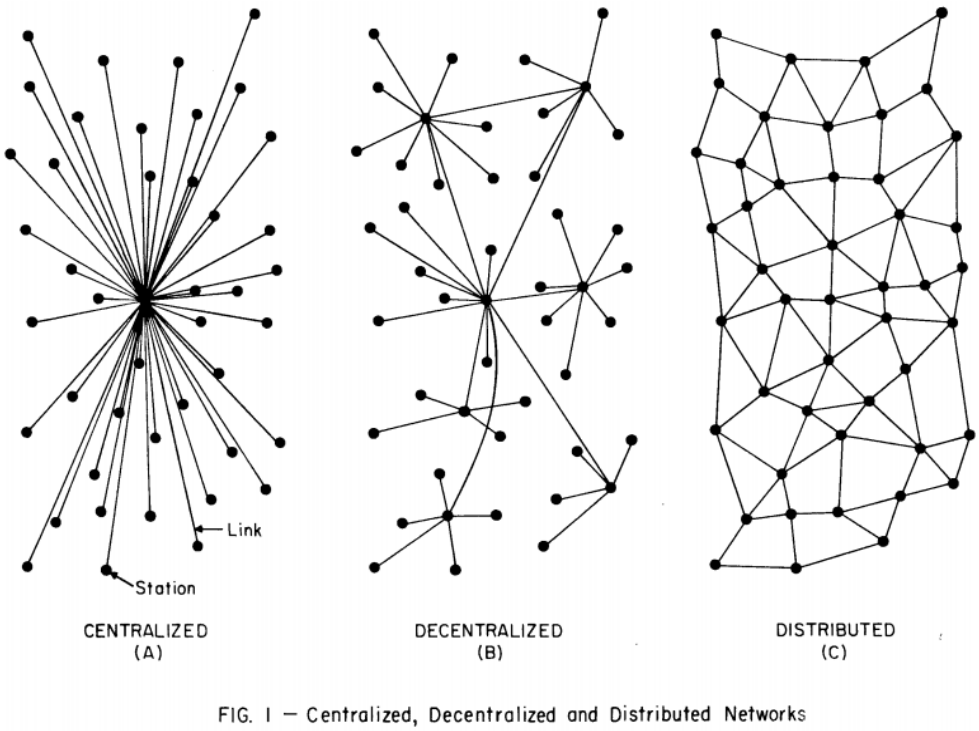

Abbildung 4 - Netzwerkarchitekturkategorien nach Baran [32]

Betrachten wir mit der klassischen Client-Server Architektur ein einfaches Beispiel für zentralisierte Netzwerke.

Die klassische Client-Server Architektur sieht klar verteilte Rollen vor. Netzwerkteilnehmer treten entweder als Client oder als Server auf, jedoch nie in beiden Rollen gleichzeitig [33]. Kommunikation zwischen zwei Clients in diesem Netzwerk ist nur über einen zentralen Server möglich, der die Datenströme weiterleitet. Eine direkte Kommunikation zwischen zwei Clients ist nicht möglich [34].

An einem Client-Server Netzwerk nehmen beliebig viele[13] Clients teil, deren Kommunikation genau über einen zentralen Server abläuft [34].

[13] Abhängig von der Kapazität des Netzwerks

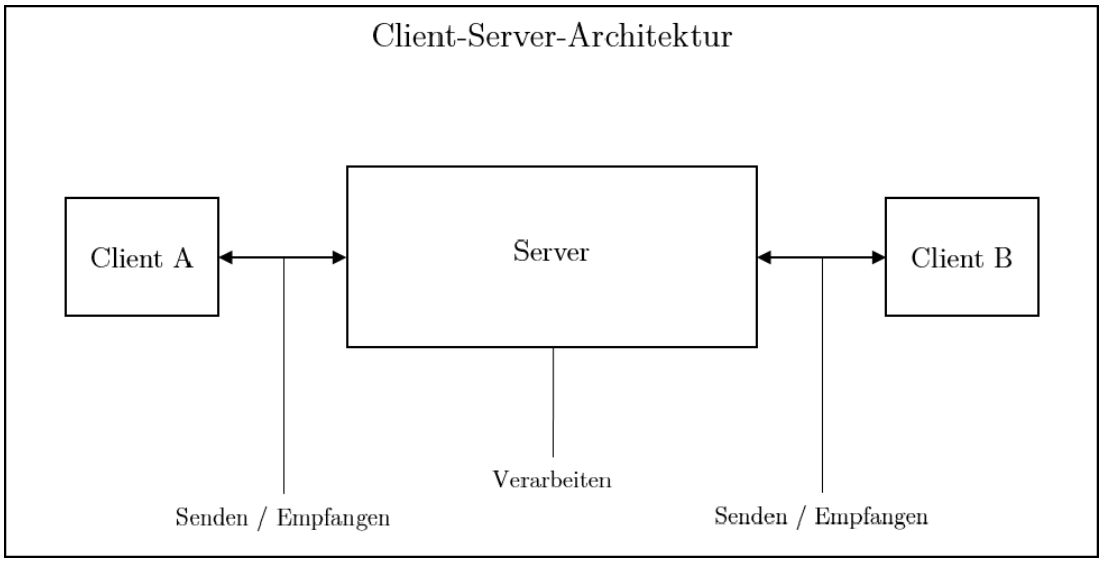

Abbildung 5 - Client-Server-Architektur[14]

Abbildung 5 zeigt die Kommunikation zwischen Client A und Client B über einen zentralen Server. Da der Server in dieser Architektur als zentrale Instanz auftritt, bezeichnet man diesen Architekturtypen auch als zentralisierte Architektur [32].

Im Vergleich dazu verrichten die Knoten in einer dezentralisierten Architektur die anfallende Arbeit in einer Art lokalen Umgebung, die nicht von einer zentralen Instanz gesteuert wird. Informationen werden ggf. über Verteilungsknoten an benachbarte Knoten weitergeleitet [35].

Hierdurch werden Redundanzen geschaffen, durch die das Risiko eines „*single point of failure*“ [36] vermieden wird. [32]

Im einer verteilten Architektur sind wiederum alle Knoten unabhängige Teilnehmer eines Gesamtsystems. Nachrichten werden immer an alle angeschlossenen Knoten verteilt. Die Kommunikation wird durch keine zentrale Kontrollinstanz gesteuert [32].

[14] (eigene Abbildung)

Die Peer-to-Peer (P2P) Architektur ist eine verteilte Netzwerkarchitektur, in der darüber hinaus alle Peers gleichberechtigte Teilnehmer des Netzwerks sind.

Die Teilnehmer des Netzwerks sind also in der Lage den Stand des Gesamtnetzwerks zu überprüfen [33].

Die Verifikation digitaler Signaturen erfolgt bei jeder Transaktion im Gesamtnetzwerk. Hierdurch wird ein sog. verteilter Konsens *(distributed consensus)* erreicht, der die Integritätsanforderungen einer Blockchain erfüllt [37].

Welche Algorithmen hierzu genutzt werden können, wird in 2.1.4 erklärt.

Wie in 2.1.1 und 2.1.2 erläutert, sind Integrität und Vertraulichkeit bereits durch Hashing und digitale Signaturen gewährleistet.

Nun gilt es nun die Anforderung an Verfügbarkeit zu erfüllen.

Greifen wir also die Netzwerkarchitekturen aus Abbildung 4 auf, um zu überprüfen welche Architektur besonders geeignet ist, um auf ihr eine Transaktionsverarbeitung aufzubauen.

Hierzu vergleichen wir die drei bekannten Architekturtypen anhand von den fünf Kategorien Wartbarkeit, Stabilität, Skalierbarkeit, Einfachheit der Entwicklung und Erweiterbarkeit / Diversität.

	Zentralisiert	Dezentralisiert	Verteilt
Wartbarkeit	Einfach, da nur ein *„single point of failure“* gewartet werden muss	Schwierig, da mehrere wartungsintensive Bereiche	Sehr schwierig, da jeder Knoten von auftretenden Problemen betroffen
Stabilität	Sehr schwierig zu garantieren, wenn *„single point of failure“* ausfällt ist immer das gesamte Netzwerk betroffen	Gut, jedoch problematisch wenn eine zu große Zahl an Verbindungsknoten ausfällt und somit das gesamte Netzwerk betroffen ist	Sehr hoch, einzelne Ausfälle richten keinen Schaden am Gesamtnetzwerk an.

Skalierbarkeit	Gering, Kapazitäten der Zentralinstanz begrenzt	Mittel, höhere Kapazität, da verteilt, aber dennoch begrenzt	Sehr hoch, problemloses Anschließen neuer Knoten
Einfachheit der Entwicklung	Einfach, viele Frameworks die mit leichter Anpassung genutzt werden können. Clients hängen jedoch von Zentralinstanz ab	Problematisch, Koordination der Nachrichten im Netzwerk und zwischen Verteilungsknoten schwierig zu regeln	Problematisch, Koordination der Nachrichten im Netzwerk und zwischen Verteilungsknoten schwierig zu regeln
Erweiterbarkeit / Diversität	Einzelnes Framework, also kaum Diversität, nur langsame Evolution	Quasi endlos wenn die Basisinfrastruktur vorhanden ist, jedoch limitierter als P2P	Quasi endlos wenn die Basisinfrastruktur vorhanden ist

Tabelle 3 - Vergleich von Netzwerkarchitekturen[15]

Wir stellen fest, dass eine zentralisierte Architektur alleine durch die geringe Stabilität und Skalierbarkeit keine gute Wahl ist, um auf ihr den Betrieb einer Blockchain aufzubauen.

Sowohl dezentralisierte als auch verteilte Architekturen weisen im Vergleich zur zentralisierten Architektur eine relativ geringe Wartbarkeit auf. Dies erfordert besondere Sorgfalt in der Entwicklung, da sich bereits kleine Fehler in der Entwicklung erheblich auf sämtliche in Tabelle 3 genannten Faktoren auswirkt.

Die dezentralisierte Architektur liefert uns bereits eine gute Skalierbarkeit, die jedoch begrenzt ist und sich nur durch erheblichen Kostenaufwand erweitern lässt [38].

Idealerweise möchten wir unsere Blockchain jedoch fast unbegrenzt und ohne großen Kostenaufwand skalieren [3]. Diese erlaubt uns nur eine verteilte Architektur, die durch ihre hohe Stabilität und Skalierbarkeit zusätzlich der hohen Verfügbarkeitsanforderung eines Blockchain Betriebs gerecht wird.

[15] In Anlehnung an [72]

2.1.4 Verteilter Konsens

Nachdem wir in 2.1.1 bis 2.1.3 die drei Kernfunktionen einer Blockchain erläutert haben, gilt es abschließend zu klären, auf welche Arten und Weisen der, in 2.1.3 angesprochene, verteilte Konsens erreicht werden kann.

Selbst in einem geschlossenen P2P Netzwerk[16] kann es zu Problemen der Transaktionsverarbeitung kommen. Betrachte man zum Beispiel unterschiedliche Transaktionsgeschwindigkeiten einzelner Peers oder unterschiedliche Datenstände, die durch technische Systemfehler auftreten [39].

Wie ist es also möglich, den wahren Stand der Daten zu bestimmen? Wie ist es möglich, in einem offenen P2P Netzwerk[17] Datenkorruption durch boshafte Peers zu verhindern?

Um diese Probleme zu lösen bedarf es einem Konsensalgorithmus (im Blockchain Kontext folgend auch Proof Algorithmen oder Konsensprotokolle genannt).

Konsensalgorithmen erlauben es, gesicherte Updates in einem verteilen System durchzuführen. Einen Konsens in verteilten Systemen zu erreichen ist extrem anspruchsvoll. Konsensalgorithmen müssen dem Ausfall von Knoten, Partitionierung des Netzwerks, Nachrichtenverzögerungen und Nachrichtenkorruption widerstehen. Außerdem wird durch sie ein Weg gefunden, um mit boshaften Knoten im Netzwerk umzugehen [37].

Für eine Blockchain garantiert die Findung eines Konsens, dass alle Knoten im Netzwerk sich auf genau einen Stand der Blockchain geeinigt haben [37].

[16] P2P Netzwerk mit Zugangsbeschränkung (Es ist anzunehmen, dass das Netzwerk so nur aus vertrauenswürdigen Peers besteht)

[17] P2P Netzwerk ohne Zugangsbeschränkung

Konsensalgorithmen sollten drei Kernprinzipien folgen [37]:

1. Sicherheit *(safety)*: Ein Konsensprotokoll gilt als sicher, wenn alle Knoten den selben Output liefern und dieser Output validiert werden kann.
2. Lebenderkennung *(liveness)*: Ein Konsensprotokoll bietet eine Lebenderkennung, wenn alle nicht fehlerhaften Knoten letzten Endes einen Output produzieren.
3. Fehlertoleranz *(fault tolerance)*: Ein Konsensprotokoll bietet Fehlertoleranz, sobald es in der Lage ist, sich ist vom Defekt von Knoten zu erholen.

Machen wir uns mit zwei ausgesuchten Konsensalgorithmen namens Proof of Work und Proof of Stake bekannt und analysieren wir abschließend ihre jeweiligen Vor- und Nachteile.

2.1.4.1 Proof of Work (PoW)

Der Proof of Work Algorithmus, wie er heute zum Beispiel von Bitcoin verwendet wird, entstammt einer Idee zur Bekämpfung von E-Mail Spam aus dem Jahr 1992 [40]. Adam Back griff diese Idee in seiner Ausarbeitung zu HashCash auf [9], welches schlussendlich die Basis für die Proof of Work Funktion von Bitcoin lieferte [11].

Proof of Work Algorithmen liefern, wie der Name bereits impliziert, eine Validierung für die Korrektheit der geleisteten Arbeit. Sie basieren auf der Lösung eines rechenintensiven Problems, eines speicherintensiven Problems oder eines Problems welches eine Nutzerinteraktion fordert (wie z.B. bei CAPTCHA Verfahren [41]) [42]. Bei gut entworfenen Proof of Work Algorithmen sollte dieses Problem für Nutzer ein geringes Hindernis, doch für Angreifer ein großes Hindernis darstellen.

Wenn nun ein Dienstanbieter dieses Problem seinen Nutzern stellt, wird nur den Nutzern, die bereit sind dieses Problem zu lösen, ein Zugang zum Dienst gewährt.

Proof of Work Algorithmen können die folgenden zwei Protokolle implementieren [42].

1. Challenge-Response Protokoll (Beispiel: CAPTCHA)
    a. Der Client sendet einen Request an den Server
    b. Der Server wählt das Problem aus und fordert den Client auf, dieses zu lösen
    c. Der Client löst Problem und sendet Ergebnis an Server
    d. Der Server verifiziert das Problem
2. Solution-Verification Protokoll (Beispiel: Bitcoin)
    a. Das Problem wird auf Clientseite erzeugt und gelöst. Dieses Problem sollte ein durch Algorithmen generiertes Problem sein, um zu gewährleisten, dass es sich bei jeder Anfrage unterscheidet.
    b. Der Client sendet die Lösung des Problems an den Server, der sie verifiziert

Um die Sicherheit einer Blockchain zu gewährleisten, können Verfahren nach den o.g. Protokollen bei der Verkettung der Blöcke, bestehend aus Transaktionen, angewandt werden.

Bitcoin in etwa nutzt das Verfahren der partiellen Hashumkehrung *(partial hash inversion)* [9] [11] als Proof of Work Funktion [42]. Das Verfahren zur Verfügung stellen von Rechenzeit zur Transaktionsbearbeitung ist als Mining bekannt [42]. Nutzer, die ihre Rechenzeit zur Verfügung stellen, werden bei Kryptowährungen wie Bitcoin anteilsmäßig entlohnt.

Die Rechenleistung ist also ausschlaggebend für die Wahrscheinlichkeit einen Block erfolgreich zu minen.

2.1.4.2 Proof of Stake (PoS)

Im Vergleich zu den in 2.1.4.1 genannten Proof of Work Algorithmen bieten Proof of Stake Algorithmen einen alternativen Weg zur Erreichung von Konsens im Netzwerk.

Bei Proof of Stake Algorithmen werden zufällig Nutzer anhand ihres Anteils am Gesamtnetzwerks zur Blockerstellung ausgewählt [37].

Diese Nutzer sind dann in der Lage ihren eigenen Anteil am Gesamtnetzwerk dazu zu nutzen, um sich eine proportionale Chance auf die Erstellung von neuen Blöcken zu sichern. Diese Nutzer bezeichnet man als Validatoren [37].

Validatoren sind, wenn sie ausgewählt werden, in der Lage den zu generierenden Block zu signieren und ihn damit zu verifizieren. Sie werden bei Kryptowährungen wie Ethereum anteilsmäßig entlohnt [43].

Der Prozess der Blockgenerierung in PoS Blockchains wird nicht wie in PoW Blockchains als „Mining“, sondern als „Minting“ bezeichnet und könnte mit „prägen“ anstatt „schürfen“ übersetzt werden.

2.1.5 Zusammenfassung der technischen Funktionsweise

Fassen wir zusammen welcher Bestandteile es bedarf, um den Betrieb einer Blockchain durchzuführen.

- Es bedarf kryptographischer Hashfunktionen, um die Integrität einer Blockchain zu wahren.
- Es bedarf digitaler Signaturen, um Nutzer und ihre Transaktionen eindeutig zu Authentifizieren und somit eine Vertrauensbasis zu schaffen.
- Es bedarf einem P2P Netzwerk, um eine Transaktionsplattform zu schaffen, die maximale Verfügbarkeit gewährleistet.
- Es bedarf einem Konsens, um eine Einigung über den aktuellsten Stand der Daten zu erlangen.

Mit Hilfe all dieser Bestandteile ist es nun möglich eine Blockchain zu erzeugen. Betrachten wir als Beispiel die Blockchain Zusammensetzung der Kryptowährung Bitcoin.

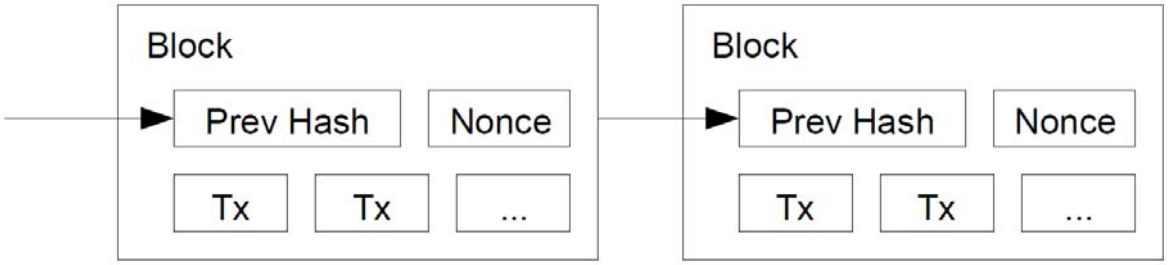

Abbildung 6 - Bitcoin Blockchain (vereinfacht) [11]

Wir erkennen Blöcke, bestehend aus den einzelnen Transaktionen, einer „Nonce“ und dem Hash des vorherigen Blocks.

Die Nonce *(number used once)* ist eine zufällig generierte 32-Bit Zahl, die als Startpunkt für Proof-of-Work Algorithmen dient. Bei Bitcoin berechnet dieser Algorithmus die Nonce so lange neu, bis sie eine Serie von führenden Nullen enthält. Die Veröffentlichung des Blocks ist damit ein Nachweis für die geleistete Arbeit [11].

Dadurch, dass jeder Block den Hash des jeweiligen Vorgängerblocks enthält, sind die angehängten Blöcke fälschungssicher verankert.

Ein Problem existiert jedoch am Anfang der Kette. Die Kette benötigt einen fest definierten Anfangsblock, auf dem sämtliche folgenden Blöcke aufbauen. Dieser Block ist als Genesis Block bekannt. Er wird im Blockchain Quellcode fest verankert und i.d.R. vor der Veröffentlichung der Blockchain generiert [3].

2.2 Smart Contracts

Die Idee der Smart Contracts stammt von Nick Szabo. Er definierte sie 1994 als computerisiertes Transaktionsprotokoll, welches Verträge und ihre Bestandteile automatisch ausführt [44] [45]. Im Vergleich zu traditionellen, rechtsbasierten Verträgen basieren sie auf mathematischen Computerprogrammen, anstatt auf Gesetzesgrundlagen [42].

Szabo sieht vier Prinzipien von traditionellen, rechtsbasieren Verträgen:

1. Beobachtbarkeit *(observability)*: Die Vertragspartner sind in der Lage, gegenseitig den Grad der Vertragserfüllung zu belegen.
2. Prüfbarkeit *(verifiability)*: Die Vertragspartner sind in der Lage, Vertragserfüllung oder Vertragsbrüche festzustellen.
3. Mitwirkungswissen *(privity)*: Die Vertragspartner besitzen die alleinige Kontrolle über die Festlegung der Vertragsbestandteile und deren Ausübung.
4. Vollstreckbarkeit *(enforceability)*: Der Vertrag ist rechtlich durchsetzbar.

In allen vier Punkten sieht Szabo Möglichkeiten durch Smart Contracts die Vertragssicherheit zu verbessern [46].

Besonders in Vergleich zu Punkt 4 sieht Szabo einen großen Vorteil von Smart Contracts, denn sie sollen in jedem Fall durchgesetzt werden, sobald die Voraussetzungen für die Vertragsschließung erreicht sind [47].

Außerdem sollen in einem Smart Contract die Vertragsbestandteile unveränderlich sein, sobald er angestoßen wurde.

Dies hat in erster Linie zum Vorteil, dass keinerlei Vertragsmanipulation möglich sind, was Smart Contracts unmittelbar sicherer macht, als traditionelle Verträge [47].

Durch seine automatisierte Ausführung werden Kosten und Zeit gespart [47].

Jedoch ist ein Computerprogramm nicht in der Lage den Input wie ein Mensch zu interpretieren. So kann die Vertragsabsicht, die den Parteien als völlig eindeutig erscheint, durch das Computerprogramm unterschiedlich interpretiert werden, sodass ein nicht gewünschtes Resultat entsteht [47].

Da keine Änderungen mehr möglich sind, können auch keine Fehler korrigiert werden. Dies gilt sowohl für den Inhalt des Vertrags, als auch für Bugs im Code des Smart Contracts.

Änderungen an der Vertragsumwelt können zu Änderungswünschen der Vertragsbestandteile führen, welche nun jedoch nicht umgesetzt werden können [48].

Die Vertragsdurchführung kann in keinem Fall abgebrochen werden, auch wenn sie (ggf. auch von beiden Parteien) nicht (mehr) gewünscht ist [47].

Wir stellen also fest, dass ein Smart Contract in der Praxis neben seinen Vorteilen auch mitunter schwerwiegende Nachteile mit sich führen kann.

Smart Contracts alleine scheinen vorerst nicht einsatzfähig, da keine Plattform existiert, um sie auszuführen. Offensichtlich bestand im Jahr 1994, zur Zeit der Begründung von Smart Contracts, auch keinerlei Verbindung zwischen Smart Contracts und Blockchain Technologie.

Szabo versucht jedoch in den Folgejahren seine Idee der Smart Contracts zu formalisieren, um sie praxisfähig zu machen, er findet jedoch weiterhin keine Plattform, die deren Einsatz universell möglich macht [44] [46].

Die erste universell einsetzbare Blockchain Applikationsplattform, die Szabos Idee von Smart Contracts implementiert, ist Ethereum [49] [50].

Die Ethereum Gründer Vitalik Buterin, Dr. Gavin Wood und Jeffrey Wilcke bezeichneten ihr Projekt, welches die Eigenschaften eines dezentralisierten Mining-

Netzwerks und einer Software Entwicklungsplattform vereint, als „Kryptowährung 2.0“ [51] [52].

Entwickler sollten in der Lage sein, auf der Ethereum Plattform ihre Apps zu entwickeln und sie in Form eines Smart Contracts im System zu veröffentlichen [51].

Durch das Einbetten von Smart Contracts in eine Blockchain Landschaft entstehen zusätzliche Vorteile gegenüber traditionellen Verträgen.

Betrachten wir die Vorteile von Smart Contracts gegenüber traditionellen Verträgen mit Hilfe der vier Prinzipien nach Szabo, die zu Beginn dieses Kapitels vorgestellt wurden.

Durch das dezentralisierte Netzwerklayout einer Blockchain ist es möglich, die von Szabo angesprochene automatische Ausführung eines Smart Contracts zu gewährleisten und den Grad der Beobachtbarkeit (1) zu steigern.

Außerdem wird so ein Backup des Vertrags geschaffen, was auf alle angeschlossenen Knoten verteilt wird [53].

Durch digitale Signaturverfahren kann auf einen Mittelsmann im Vertragsprozess verzichtet werden, da beide Parteien in der Lage sind sich gegenseitig zu authentifizieren. Durch Hashing ist es möglich Smart Contracts eindeutig zu beschreiben.

Hierdurch wird der Grad der Prüfbarkeit (2) gesteigert, sowie das Vertragsmitwirkungsrecht (3) ausschließlich auf die Vertragsparteien beschränkt.

Durch die Übergabe der Entscheidungsgewalt von Mensch zu Computer wird außerdem die Vollstreckbarkeit (4) garantiert.

Alle Punkte führen schlussendlich zu einer Erhöhung der Sicherheit, wie es Szabo ursprünglich beabsichtigte [44].

Jedoch können bei Nutzung eines Smart Contracts in einer Blockchain auch Nachteile entstehen.

Da der Smart Contract Code von Menschen geschrieben wird, kann es weiterhin bei Fehlern im Programmierprozess zu schwerwiegenden Auswirkungen bei der Vertragsabwicklung kommen [54].

Des Weiteren erfordert die Entwicklung von Smart Contracts erhebliche Zeit- und Kostenaufwendungen.

Außerdem ist es mitunter kompliziert Smart Contracts für die jeweiligen Applikationen korrekt in die bestehende Blockchain Infrastruktur einzubetten [54].

Es ist essentiell keine Fehler in der Entwicklung zu begehen, um die o.g. Auswirkungen zu vermeiden [54].

Ein weiteres Problem ist, dass (zum Stand dieser Arbeit) noch keine Rechtsgrundlage für Smart Contracts existiert, sodass sie auch nicht rechtlich abgesichert werden können [54].

Es ist also zu beachten, dass auch die Implementierung von Smart Contracts in einer Blockchain grundlegende Nachteile bestehen bleiben, die bei Einsatz von Smart Contracts in jedem Fall berücksichtigt werden müssen.

3 Anwendung von Blockchain Technologie

Nachdem wir uns mit den Grundlagen von Blockchain Technologie bekannt gemacht haben, gilt es nun eine Übersicht über Blockchain Geschäftsmodelle zu erlangen.

In Folge dessen gilt es Applikationsmöglichkeiten und Disruptionspotenziale aufzuzeigen.

3.1 Blockchain Geschäftsmodelle

In einer von Deloitte durchgeführten Studie [55] zum Thema Blockchain in Unternehmen für das Jahr 2017 gaben bereits über 40% der befragten Führungskräfte an, dass sie der Meinung sind, Blockchain Technologie werde ihre Industrie spalten.

55% geben an, dass sie fürchten ihre Wettbewerbsfähigkeit zu verlieren, wenn sie zukünftig keine Blockchain Technologie einsetzen.

Besonders interessant sind jedoch die drei Hauptvorteile, die Führungskräfte in dem Einsatz von Blockchain Technologie sehen.

So sehen 37% der Befragten verbesserte Sicherheit, 36% niedrigere Kosten bei erhöhter Geschwindigkeit und 24% neue Geschäftsmodelle bzw. Einnahmequellen als Hauptvorteile an.

Besonders die Bereiche Finanzdienstleistungen, Versicherungsdienstleistungen, Logistik, Gesundheit und öffentliche Dienstleistungen nehmen laut Deloitte eine Pionierrolle in der Blockchainimplementierung ein [56].

Wir stellen also fest, dass ein grundlegendes Interesse an Blockchain Technologie besteht und ihre Potenziale langsam erkannt werden.

3.2 Applikationsmöglichkeiten und Disruptionspotenziale

3.2.1 Finanzindustrie

Die Transaktionseffizienz von Banken liegt, selbst nach der Einführung des Online Banking, weit hinter den, aus dem Alltag bekannten, technischen Möglichkeiten [57].

In einer Zeit von grenzüberschreitender Echtzeit-Onlinekommunikation über High-Speed Internetanschlüsse benötigt eine einfache transnationale Banküberweisung erfahrungsgemäß immer noch mehrere Werktage.

Währenddessen haben sich bereits Online Zahlungssysteme (PayPal) etabliert, die grenzübergreifende Transaktionen in Echtzeit mit vergleichsweise geringen Gebühren möglich machen.

Obwohl Online Zahlungssysteme die Bankenlandschaft bereits zugänglicher gemacht haben, verfügen dennoch Milliarden von Menschen, vor allem in Entwicklungsländern, über kein eigenes Bankkonto [58].

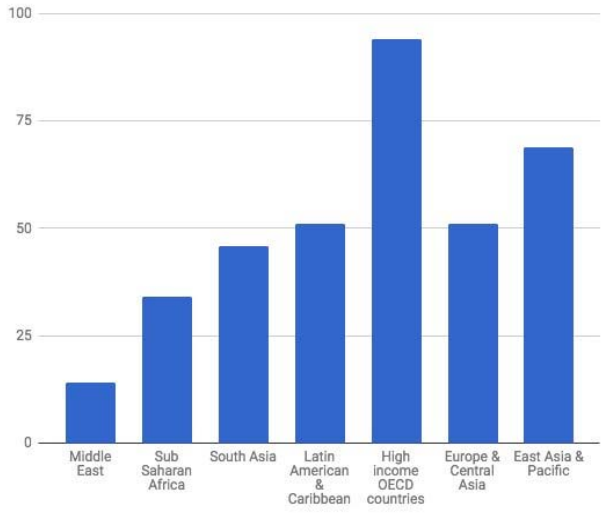

Abbildung 7 - Anteil der Personen mit Besitz eines Bankkontos nach geographischen Regionen [58]

Diese Menschen verfügen jedoch zumeist über ein Smartphone, welches sie für Zahlungsdienstleistungen nutzen [58].

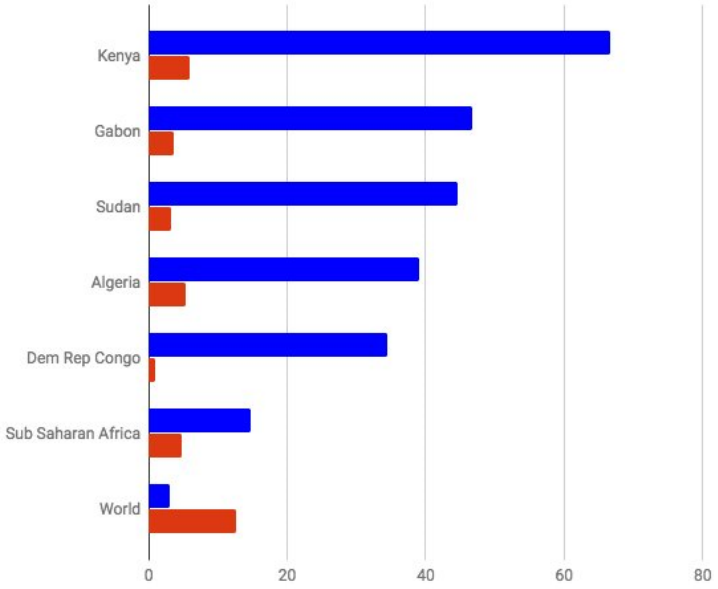

Abbildung 8 - Anteil der Bevölkerung über dem Alter von 15 Jahren, die mobile Zahlungsdienstleistungen nutzen (blau), zu Bankfilialen pro 100.000 Menschen (rot) [58]

Wir stellen fest, dass kaum eine traditionelle finanzielle Infrastruktur besteht, jedoch der Zugang zu internetfähigen mobilen Endgeräten gegeben ist.

Dies beschreibt einen fundamentalen Entwicklungsnachteil, der durch den Ausbau traditioneller Infrastruktur kaum aufzuholen ist.

Mit Hilfe von Blockchain Technologie sind jedoch globale Zahlungen in wenigen Sekunden und fast ohne Gebühren möglich, denn die Transaktionen werden in den aus 2.1.3 bekannten P2P Netzwerken durchgeführt.

Hierdurch ergibt sich ein erheblicher Kosten- und Geschwindigkeitsvorteil, da auf Server, Clearingstellen und Compliance Prüfungen verzichtet werden kann [57].

Ein weiterer Vorteil ist, dass bei grenzübergreifenden Zahlungen die evtl. anfallenden Währungskursumrechnungen entfallen [57].

Diese Mängel wurden längst von Banken erkannt, die nun versuchen ihr bestehendes Geschäftsmodell zu erweitern oder umzubauen, um zukünftige Wettbewerbsfähigkeit zu gewährleisten.

Die enorme Verbesserung der Transaktionsgeschwindigkeit und die geringen Transaktionskosten eröffnen Blockchain Technologie enorme Disruptionspotenziale in der Finanzindustrie.

Doch auch die erhöhte Sicherheit, durch die in 2.1.1 und 2.1.2 beschriebene Hashfunktionen und digitalen Signaturverfahren, gewährleistet eine Verbesserung zu der momentanen Situation der Finanzindustrie.

In einer Studie von der Economist Intelligence Unit, welche von IBM beauftragt wurde, haben 91% der untersuchten Banken bereits in Blockchain Lösungen im Geschäftsbereich des Einlagengeschäfts investiert, um sich bis 2018 gegen nicht bankaffiliierte Start-Ups zu schützen [59].

Bereits 15% der Banken gaben der Studie zu Folge an, bereits kommerziell einsatzfähige Blockchainlösungen in den Jahren 2017 und 2018 bereitstellen zu können. 51% gaben an bis zum Jahr 2020 einsatzfähige Blockchainlösungen entwickeln zu wollen und nur 34% geben an, erst ab dem Jahr 2020 Lösungen vorweisen zu wollen [59].

Bemerkenswert ist es jedoch, dass Blockchain Technologie eigentlich die gesamte etablierte Finanzindustrie überflüssig machen kann, denn mit Hilfe von Kryptowährungen können Werte außerhalb der bestehenden Finanzinfrastruktur geschaffen werden.

3.2.1.1 Fallbeispiel: Stellar

Stellar ist eine verteilte, open-source Finanzdienstleistungsplattform, welche zum besonderen Ziel hat, Menschen in Entwicklungsländern den Zugang zu internationalen Finanztransaktionen zu ermöglichen.

Insbesondere legen die Entwickler von Stellar Wert auf geringe Transaktionsgebühren, kurze Transaktionsdauer und ausreichend hohe Netzwerkkapazitäten.

Stellar basiert auf der Kryptowährung Ripple (XRP) und verwendet sein eigenes Konsensprotokoll - das Stellar Consensus Protocol (SCP).

Doch was macht Stellar zu einer besseren Kryptowährung, als Konkurrenten wie Bitcoin, Ethereum oder Ripple? Dazu ist ein Vergleich von Key Performance Indicators wie Transaktionsdauer, Transaktionskosten und der maximalen Transaktionsanzahl pro Sekunden nötig.

	Bitcoin	Ethereum	Ripple	Stellar
Marktkapitalisierung	$283.4 Mrd.	$108.2 Mrd.	$125.4 Mrd.	$12.5 Mrd.
ø Transaktionsdauer*	8m 47s	15.8s	3.3s	5s
ø Transaktionskosten**	$33	$0.80	$0.02	$0.000078
Max. Transaktionen pro Sekunde	~7	~15	~1500	> 2000
Konsensalgorithmus	Proof-of-Work	Proof-of-Stake	XRP Ledger Protocol	Stellar Consensus Protocol

* Im 7-Tage Durchschnitt, bezogen auf einen Block, gerundete Werte

** Im 7-Tage Durchschnitt, gerundete Werte

Tabelle 4 - Vergleich von Kryptowährungen [60] [61] [62] [63] [64]

Die o.g. Auswertungen beziehen sich auf einen Zeitrahmen vom 28.12.2017 bis zum 04.01.2018. Es handelt sich hierbei um Durchschnittswerte.

Wir stellen fest, dass Stellar bei ähnlich schneller Transaktionsgeschwindigkeit, im Vergleich zu Ripple, die bereits geringen Transaktionskosten weiter um den Faktor 256 unterbietet. Bitcoin und Ethereum scheinen bezüglich der Transaktionsdauer und vor allem bzgl. der maximal möglichen Transaktionen pro Sekunde völlig abgeschlagen. Hohe Geschwindigkeit, gepaart mit geringen Kosten und hoher Skalierbarkeit machen erst einen alltäglichen Einsatz, z.B. im grenzübergreifenden Zahlungsverkehr möglich.

3.2.2 eGovernment

Betrachten wir nun ein Beispiel zum Thema *eGovernment* und *eHealthcare*, einer digitalen Version der öffentlichen Verwaltung und des Gesundheitswesens am Beispiel Estland.

Estland begann bereits im Jahr 2008 mit Tests der Blockchain Technologie in einer Form von *„hash-linked time-stamping“*, also durch Hashing verküpfte Zeitstempel. Dies geschah bemerkenswerter Weise noch bevor die Blockchain Technologie, durch Entwicklungen wie Bitcoin, der breiten Masse bekannt gemacht wurde [65].

Seit 2012 verwendet Estland Blockchain Technologie, um Daten ihrer Bürger sicher zu speichern, die dadurch zu *eResidents* werden [65].

Hierzu wurde eine eigene Blockchain Implementierung mit dem Namen *KSI Blockchain* entwickelt, die Dienste wie *eHealth* und *ePresecription* im Gesundheitswesen, *eLaw*, *eCourt* und *ePolice* im öffentlich-rechtlichen Sektor sowie *eBanking* und *eBusiness* Dienste im finanziellen Sektor anbietet [65].

Hierdurch entstehen Möglichkeiten, wie Online Vertragsschließung, Online Wahlen, Digitale Medikamentenrezepte und die online Beantragung von staatlichen Förderleistungen [66].

Durch die Digitalisierung der Verwaltung wird es erst möglich, neue digitale Produkte in die Wertschöpfungskette zu integrieren.

Mittlerweile hat Estland nach dem Vorbild des inländischen Systems, ein Projekt zur Erschaffung einer globalen, transnationalen, digitalen Identität, der sog. *eResidency,* ins Leben gerufen [67].

Dieses Programm hat seit seiner Einführung Mitte 2017 bereits rund 27.000 Nutzer. Weitere ca. 28.000 Personen haben sich auf eine *eResidency* beworben[18].

Ca. 41% dieser Personen gaben die Betreibungsabsicht eines standortunabhängigen, globalen Geschäftsbetriebs an. Ca. 27% verfolgen das Ziel eine Geschäftstätigkeit in Estland ausüben zu wollen [68].

Überraschend ist die Altersverteilung der Nutzer. Obwohl es sich um einen Dienst handelt, der durch den Einsatz neuartiger Technologie wohl eher für die Generation der Digital Natives geeignet sein dürfte, sind fast 30% der Nutzer zwischen 31 und 40 Jahren alt. Nur ca. 20% der Nutzer sind unter 30 Jahre alt.

Durch die Anwendung von Blockchain Technologie in der öffentlichen Verwaltung bzw. in öffentlichen Institutionen, ergeben sich Vorteile sowohl für die Landesverwaltung, als auch für die Bürger des Landes.

So ist es durch Blockchain Technologie möglich, Zugriffe auf persönliche Daten zu beschränken sowie eine Protokollierung der erfolgten Zugriffe durchzuführen.

Blockchain Technologie erlaubt es, in einer Art Zeitstrahl, die Änderungen an Unternehmensdaten, Immobiliendaten oder in Dokumenten nachzuvollziehen.

[18] Stand 01.01.2018

Durch den Einsatz verteilter Netzwerke wird eine hohe Performanz und hohe Ausfallsicherheit garantiert, was besonders im öffentlichen Sektor ein großer Vorteil ist.

Der Service ist rund um die Uhr, das gesamte Jahr nutzbar, was die Nutzung öffentlicher Dienste für Bürger vereinfacht und zu Personal- und Kosteneinsparungen führt.

4 Fazit

Nachdem wir uns mit den Komponenten einer Blockchain, ihrer Funktionsweise sowie ihren Vor- und Nachteilen befasst haben sowie Applikationsmöglichkeiten und Disruptionspotenziale aufzeigt haben, gilt es nun ein abschließendes Fazit zu ziehen.

Blockchain Technologie ist weit mehr als nur eine neue Art Daten zu speichern. Sie stellt eine vollkommen neue Art und Weise dar, wie Werte gespeichert werden können.

Sie hat das Potenzial, neue Fundamente für Industrie, Wirtschaft und öffentliche Verwaltung zu legen und damit völlig neue Gesellschaftssysteme zu schaffen.

Jedoch befinden wir uns erst am Anfang dieser Technologie. Die Potenziale werden erst langsam erkannt. Noch besteht eine gewisse Unsicherheit über die Auswirkungen der Blockchain Technologie. So befinden sich auch Forschung und Entwicklung möglicher Anwendungsszenarien erst am Anfang.

Parallel aufkommende Technologien wie Deep Learning, Artificial Intelligence, Internet of Things werden ohne Frage Applikationsmöglichkeiten für Blockchain Technologie bilden.

Der Prozess weitreichender Blockchainimplementierung muss daher behutsam und sukzessiv erfolgen, um zeitgleich eine Weiterentwicklung der Technologie zu ermöglichen.
Es wird vermutlich noch Jahrzehnte dauern, bis Blockchain Technologie verbreitet in unser alltägliches Leben Einzug gehalten hat.

5 Einordnung in den Veranstaltungskontext

Im Rahmen der Veranstaltung „Mobile Anwendungen und Systeme“ wird u.a. die Bedeutung der Digitalisierung und ihre Auswirkung auf Geschäftsmodelle in unterschiedlichen wirtschaftlichen Bereichen behandelt. Die Blockchain Technologie bietet eine ideale Plattform für innovative, digitale Geschäftsmodelle.

Anhang

Literaturverzeichnis

[1] D. Furlonger, V. Ray und R. Kandaswamy, „Blockchain," *Hype Cycle for Emerging Technologies 2017,* 21 Juli 2017.

[2] V. Schlatt, A. U. N. Schweizer und G. Fridgen, „Blockchain: Grundlagen, Anwendungen und Potenziale," *Fraunhofer FIT,* pp. 1-54, Dezember 2016.

[3] D. Drescher, Blockchain Basics: A Non-Technical Introduction in 25 Steps, Apress, 2017.

[4] R. Bodo, „Usage of the word "blockchain"," 20 September 2017. [Online]. Verfügbar unter: https://medium.com/@richbodo/common-use-of-the-word-blockchain-5b916cecef29. [Zugriff am 16 November 2017].

[5] W. F. Ehrsam, C. H. W. Meyer, J. L. Smith und W. L. Tuchman, „Message verification and transmission error detection by block chaining". USA Patent US 4074066 A, 1976.

[6] „Cifrado CBC," 21 Juni 2006. [Online]. Verfügbar unter: https://commons.wikimedia.org/wiki/File:Cbc_encryption.png. [Zugriff am 18 November 2017].

[7] M. Scharwies, „Plain text," 17 März 2017. [Online]. Verfügbar unter: https://wiki.selfhtml.org/wiki/Plain_text. [Zugriff am 20 November 2017].

[8] N. Szabo, „Bit Gold," 29 Dezember 2005. [Online]. Verfügbar unter: http://nakamotoinstitute.org/bit-gold/. [Zugriff am 18 November 2017].

[9] A. Back, „Hashcash - A Denial of Service Counter-Measure," 1 August 2002. [Online]. Verfügbar unter: http://www.hashcash.org/papers/hashcash.pdf. [Zugriff am 16 November 2017].

[10] Gabler Wirtschaftslexikon, „Stichwort: Peer-to-Peer (P2P),“ [Online]. Verfügbar unter: http://wirtschaftslexikon.gabler.de/Archiv/77351/peer-to-peer-p2p-v11.html. [Zugriff am 19 November 2017].

[11] S. Nakamoto, „Bitcoin: A Peer-to-Peer Electronic Cash System,“ 2008. [Online]. Verfügbar unter: https://bitcoin.org/bitcoin.pdf. [Zugriff am 16 Dezember 2017].

[12] H. Finney, „Bitcointalk,“ 19 März 2013. [Online]. Verfügbar unter: https://bitcointalk.org/index.php?topic=155054.0. [Zugriff am 19 November 2017].

[13] M. Huzaifa Ali, „Medium,“ 4 Januar 2014. [Online]. Verfügbar unter: https://medium.com/@mhuzaifaali/second-bitcoin-miner-after-satoshi-shares-his-experience-1c9b0c107ed. [Zugriff am 19 November 2017].

[14] H. Finney, „The Cryptography Mailing List,“ 09 November 2008. [Online]. Verfügbar unter: http://satoshi.nakamotoinstitute.org/emails/cryptography/6/. [Zugriff am 19 November 2017].

[15] E. Pisciini, D. Dalton und L. Kehoe, „Blockchain & Cyber Security. Let's Discuss,“ *Deloitte Performance Magazine,* pp. 42-49, 27 September 2017.

[16] A. Menezes, P. van Oorschot und S. Vanstone, Handbook of applied cryptography, CRC Press, 1997.

[17] R. Merkle, *Secrecy, Authentication, and Public Key Systems,* UMI Research, 1979.

[18] I. B. Damgard, „Collision free hash functions and public key signature schemes,“ *Advances in Cryptology, Proc. Eurocrypt'87, LNCS 304,* pp. 203-216, 1988.

[19] B. Preneel, *Analysis and Design of Cryptographic Hash Functions, Doctoral Dissertation,* Katholieke Universiteit Leuven, 2003.

[20] A. G. Konheim, Hashing in Computer Science: Fifty Years of Slicing and Dicing, John Wiley & Sons, 2010.

[21] National Institute of Standards and Technology, *FIPS PUB 180-4 Secure Hash Standards,* N. I. o. S. a. T. (NIST), Hrsg., 2015.

[22] American National Standards Institute, *Public Key Cryptography For The Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA),* Washington, DC: Accredited Standards Committee X9, 1998.

[23] P. Hartmann, Mathematik für Informatiker: Ein praxisbezogenes Lehrbuch, 6., illustriert Hrsg., 2014: Springer-Verlag.

[24] Dudenreaktion, „Unterschrift," [Online]. Verfügbar unter: https://www.duden.de/node/704751/revisions/1331184/view. [Zugriff am 20 Dezember 2017].

[25] J. Buchmann, Einführung in die Kryptographie, 4., erw. Aufl. Hrsg., Springer Berlin Heidelberg, 2008.

[26] T. Pornin, „Stackexchange," 2 Oktober 2014. [Online]. Verfügbar unter: https://security.stackexchange.com/questions/68822/trying-to-understand-rsa-and-its-terminology?answertab=votes#tab-top . [Zugriff am 21 Dezember 2017].

[27] W. Diffie und M. E. Hellman, „New Directions in Cryptography," *IEEE Transactions on Information Theory,* Bd. 22, Nr. 6, pp. 644-654, November 1976.

[28] R. L. Rivest, A. Shamir und L. Adleman, „A Method for Obtaining Digital Signatures and Public-key Cryptosystems," *Commun. ACM,* Bd. 21, Nr. 2, pp. 120--126, Februar 1978.

[29] V. Katiyar, K. Dutta und S. Gupta, „A Survey on Elliptic Curve Cryptography for Pervasive Computing Environment," *International Journal of Computer Applications,* Dezember 2010.

[30] N. Koblitz, „Mathematics of computation," *Mathematics of computation,* Nr. 177, pp. 203-209, 1987.

[31] S. Thompson, „The preservation of digital signatures on the blockchain," *See Also: the University of British Columbia iSchool Student Journal,* Bd. 3 (Spring 2017), 2017.

[32] P. Baran, *On Distributed Communications Networks,* The RAND Corporation, 1962.

[33] R. Schollmeier, „A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications," *First International Conference on Peer-to-Peer Computing,* pp. 101-102, 2001.

[34] D. Volkov, „A Concept of a Universal Server for the Client–Server Architecture," *Programming and Computer Software,* Bd. 26, Nr. 6, pp. 341-345, 2000.

[35] J. Benet, „Github," 9 November 2015. [Online]. Verfügbar unter: https://github.com/WebOfTrustInfo/rebooting-the-web-of-trust/issues/50#issuecomment-154995201. [Zugriff am 26 Dezember 2017].

[36] Business Dictionary, „Single Point of Failure," [Online]. Verfügbar unter: http://www.businessdictionary.com/definition/single-point-of-failure.html. [Zugriff am 26 November 2017].

[37] A. Baliga, *Understanding Blockchain Consensus Models,* Persistent Systems Ltd., 2017.

[38] S. de Gyurky und M. A. Tarbell, The Cognitive Dynamics of Computer Science: Cost-Effective Large Scale Software Development, Wiley-IEEE Computer Society Pr, 2006.

[39] A. Lewis, *A Gentle Introduction to Blockchain Technology,* BraveNewCoin, Hrsg., 2016.

[40] C. Dwork und M. Naor, „Pricing via Processing or Combatting Junk Mail," *Advances in Cryptology - CRYPTO' 92: 12th Annual International Cryptology Conference Santa Barbara, California, USA August 16--20, 1992 Proceedings,* pp. 139-147, 1993.

[41] CAPTCHA. [Online]. Verfügbar unter: http://www.captcha.net/. [Zugriff am 2017 Dezember 29].

[42] P. Franco, Understanding Bitcoin: Cryptography, Engineering, and Economics, Chichester, Großbritannien: John Wiley & Sons, Ltd, 2014.

[43] S. Ray, „Hackernoon," 6 Oktober 2017. [Online]. Verfügbar unter: https://hackernoon.com/what-is-proof-of-stake-8e0433018256 . [Zugriff am 26 Dezember 2017].

[44] N. Szabo, „The Idea of Smart Contracts," 1997. [Online]. Verfügbar unter: http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_idea.html. [Zugriff am 4 November 2017].

[45] N. Szabo, „Smart Contracts," 1994. [Online]. Verfügbar unter: http://web.archive.org/web/20160305205247/http://szabo.best.vwh.net/smart.contracts.html. [Zugriff am 16 Dezember 2018].

[46] N. Szabo, „Formalizing and Securing Relationships on Public Networks," *First Monday,* Bd. 2, Nr. 9, 1 September 1997.

[47] D. Gerard, Attack of the 50 Foot Blockchain: Bitcoin, Blockchain, Ethereum & Smart Contracts, Amazon Digital Services, 2017.

[48] S. Grybniak, „Hackernoon," 28 Dezember 2017. [Online]. Verfügbar unter: https://hackernoon.com/advantages-and-disadvantages-of-smart-contracts-in-financial-blockchain-systems-3a443145ae1c. [Zugriff am 29 Dezember 2017].

[49] V. Buterin, „Ethereum White Paper," 2013. [Online]. Verfügbar unter: https://github.com/ethereum/wiki/wiki/White-Paper. [Zugriff am 04 November 2017].

[50] G. Wood, 2014. [Online]. Verfügbar unter: http://gavwood.com/Paper.pdf. [Zugriff am 04 November 2017].

[51] N. Hajdarbegovic, „Coindesk,“ 23 Januar 2014. [Online]. Verfügbar unter: https://www.coindesk.com/ethererum-launches-cryptocurrency-2-0-network/. [Zugriff am 28 Dezember 2017].

[52] V. Buterin, „Bitcointalk,“ 23 Januar 2014. [Online]. Verfügbar unter: https://bitcointalk.org/index.php?topic=428589.0. [Zugriff am 28 Dezember 2017].

[53] ruyzakilost, „Steemit,“ Oktober 2017. [Online]. Verfügbar unter: https://steemit.com/ethereum/@ryuzakilost/ethereum-smart-contracts-101-hello-world. [Zugriff am 29 Dezember 2017].

[54] J. Banu, „A to Z Forex,“ 2 November 2017. [Online]. Verfügbar unter: https://atozforex.com/news/pros-and-cons-of-smart-contracts/. [Zugriff am 29 Dezember 2017].

[55] Deloitte, „Deloitte blockchain survey 2017,“ 1 Dezember 2016. [Online]. Verfügbar unter: https://www2.deloitte.com/us/en/pages/about-deloitte/articles/innovation-blockchain-survey.html. [Zugriff am 30 Dezember 2017].

[56] J. Seffinga, L. Lyons und A. Bachmann, *Deloitte: The Blockchain (R)evolution - The Swiss Perspective,* 2017.

[57] S. Voshmgir, *Blockchains, Smart Contracts und das Dezentrale Web,* Berlin: Technologiestiftung Berlin, 2016.

[58] C. Hodgson, „The world's 2 billion unbanked, in 6 charts,“ 30 August 2017. [Online]. Verfügbar unter: http://www.businessinsider.de/the-worlds-unbanked-population-in-6-charts-2017-8?op=1. [Zugriff am 28 Dezember 2017].

[59] IBM Institute for Business Value, „Leading the pack in blockchain banking,“ IBM, Somers, New York, USA, 2016.

[60] Coinmarketcap, „Coinmarketcap,“ 2018. [Online]. Verfügbar unter: https://coinmarketcap.com/. [Zugriff am 5 Januar 2018].

[61] BitInfoCharts, „Kryptowährung Statistik,“ 2018. [Online]. Verfügbar unter: https://bitinfocharts.com/de/. [Zugriff am 05 Januar 2018].

[62] Etherscan, „Etherscan - The Ethereum Block Explorer,“ 2018. [Online]. Verfügbar unter: https://etherscan.io/. [Zugriff am 5 Januar 2018].

[63] Ripple, „Ripple XRP Charts,“ 2018. [Online]. Verfügbar unter: https://xrpcharts.ripple.com/#/metrics. [Zugriff am 5 Januar 2018].

[64] B. Nowotarski, „Stellar.org Dashboard,“ 2018. [Online]. Verfügbar unter: https://dashboard.stellar.org/. [Zugriff am 5 Januar 2018].

[65] E-Estonia, „E-Estonia,“ 2018. [Online]. Verfügbar unter: https://e-estonia.com/. [Zugriff am 1 Januar 2018].

[66] D. de Boer, „BTC-Echo,“ 24 August 2017. [Online]. Verfügbar unter: https://www.btc-echo.de/estland-erwaegt-eigenen-ico-fuer-estcoins/. [Zugriff am 1 Januar 2018].

[67] Republic of Estonia, „E-Residency,“ Januar 2018. [Online]. Verfügbar unter: https://e-resident.gov.ee/become-an-e-resident/. [Zugriff am 1 Januar 2018].

[68] Cyfe, „E-Residents Overview,“ 3 Januar 2018. [Online]. Verfügbar unter: https://app.cyfe.com/dashboards/195223/5587fe4e52036102283711615553. [Zugriff am 5 Januar 2018].

[69] Government Office for Science, Great Britain, „Distributed Ledger Technology: Beyond Block Chain,“ Government Office for Science, 2016.

[70] A. Back, „hashcash.org,“ 28 März 1997. [Online]. Verfügbar unter: http://www.hashcash.org/papers/announce.txt. [Zugriff am 19 November 2017].

[71] S. Kudva, „LinkedIn,“ 31 März 2016. [Online]. Verfügbar unter: https://www.linkedin.com/pulse/difference-between-system-record-source-truth-santosh-kudva. [Zugriff am 2017 November 26].

[72] S. Goyal, „Medium,“ 1 Juli 2015. [Online]. Verfügbar unter: https://medium.com/@bbc4468/centralized-vs-decentralized-vs-distributed-41d92d463868. [Zugriff am 26 Dezember 2017].

Quellcode

```
static String parseBinary(String input){
        String k = "";
        for(int i = 0; i < input.length(); i += 2){
                k += hexToBinary(input.substring(i, i+2)) + " ";
        }
        return k;
}
```

Abbildung 9 - parseBinary() erzeugt aus einem Inputstring n Substrings m für jeden Hexadezimalwert des Inputstrings n und übergibt diese hexToBinary()

```
static String hexToBinary(String hexBits) {
        int intversion = Integer.parseInt(hexBits, 16);
        String binaryVers = Integer.toBinaryString(intversion);
        int padding = 8 - binaryVers.length();
        while (padding > 0) {
                binaryVers = "0" + binaryVers;
                padding--;
        }
        return binaryVers;
}
```

Abbildung 10 - hexToBinary() einen String n in Blöcke von 8-Bit Binärzahlen um

```
static int compareBits(String a, String b){
        int ct = 0;
        for(int i = 0; i < a.length(); i++){
                if(a.charAt(i) != b.charAt(i)){
                        ct++;
                }
        }
        return ct;
}
```

Abbildung 11 - compareBits() gibt die Anzahl der Bits zurück, um die sich die Eingabestrings a und b unterscheidenden

```
static KeyPair generateKeyPair() throws Exception{
  KeyPairGenerator generator = KeyPairGenerator.getInstance("RSA");
  generator.initialize(2048, new SecureRandom());
  KeyPair pair = generator.generateKeyPair();

  return pair;
}
```

Abbildung 12 - generateKeyPair() generiert ein Schlüsselpaar bestehend aus öffentlichem und privatem Schlüssel

```
static String encrypt(String plainText, PublicKey publicKey) throws Exception {
   Cipher encryptCipher = Cipher.getInstance("RSA");
   encryptCipher.init(Cipher.ENCRYPT_MODE, publicKey);

   byte[] cipherText = encryptCipher.doFinal(plainText.getBytes("UTF-8"));

   return Base64.getEncoder().encodeToString(cipherText);
}
```

Abbildung 13 - encrypt() verschlüsselt die Eingabenachricht mit Hilfe des öffentlichen Schlüssels

```
static String decrypt(String cipherText, PrivateKey privateKey) throws Exception {
   byte[] bytes = Base64.getDecoder().decode(cipherText);

   Cipher decriptCipher = Cipher.getInstance("RSA");
   decriptCipher.init(Cipher.DECRYPT_MODE, privateKey);

   return new String(decriptCipher.doFinal(bytes), "UTF-8");
}
```

Abbildung 14 - decrypt() entschlüsselt die Nachricht mit Hilfe des privaten Schlüssels

```
static String sign(String plainText, PrivateKey privateKey) throws Exception {
   Signature privateSignature = Signature.getInstance("SHA256withRSA");
   privateSignature.initSign(privateKey);
   privateSignature.update(plainText.getBytes("UTF-8"));

   byte[] signature = privateSignature.sign();

   return Base64.getEncoder().encodeToString(signature);
}
```

Abbildung 15 – sign() signiert die Nachricht mit Hilfe des privaten Schlüssels

```
static boolean verify(String plainText, String signature, PublicKey publicKey) throws
Exception {
   Signature publicSignature = Signature.getInstance("SHA256withRSA");
   publicSignature.initVerify(publicKey);
   publicSignature.update(plainText.getBytes("UTF-8"));

   byte[] signatureBytes = Base64.getDecoder().decode(signature);

   return publicSignature.verify(signatureBytes);
}
```

Abbildung 16 - verify() verifiziert die Nachricht mit Hilfe des öffentlichen Schlüssels und der Signatur aus Abbildung 15

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Seminararbeit selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe.

Die Stellen der Seminararbeit, die anderen Quellen im Wortlaut oder dem Sinn nach entnommen wurden, sind durch Angaben der Herkunft kenntlich gemacht.

Dies gilt auch für Zeichnungen, Skizzen, bildliche Darstellungen sowie Quellen aus dem Internet.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Bochum, den 08.01.2018

Marius Golombeck