Die Blockchain-Technologie - eine Einführung

blank

Januar 2021

Inhaltsverzeichnis

1 Einleitung

Abstract. Eine Blockchain ist eine Art verteilte Datenbank, welche eine laufend größer werdende Informationsliste in Blöcken enthält. Um diese Blöcke vor fälschlicher Veränderung abzusichern, wird ein Block mit den anderen Blöcken verbunden, indem jeder Block den kryptographischen Hash des vorherigen Blocks enthält. Daraus entsteht eine Kette von Blöcken, die Blockchain. Dieses Paper wird sich mit den grundlegenden Funktionsweisen und Eigenschaften einer Blockchain beschäftigen und diese am Beispiel der Kryptowährung Bitcoin beleuchten.

Seit der Veröffentlichung des White Paper durch das Pseudonym Satoshi Nakamoto im Jahr 2008 und der Schöpfung der ersten Bitcoin, haben Kryptowährungen und die zugrundeliegende Technologie der Blockchain einen riesigen „Hype“ erfahren.
In das Auge der Öffentlichkeit ist Bitcoin besonders im Jahre 2017, durch die extreme Kursentwicklung, geraten. Viele Unternehmen und Privatpersonen haben darauf folgend angefangen ihre eigenen Blockchain Projekte zu verwirklichen, wobei unzählige unterschiedliche Implementierungen der Blockchain-Technologie entwickelt wurden und in momentan Entwicklung sind. Allein die Kryptowährungsseite Coinmarketcap listet mittlerweile 8271 verschiedene Kryptowährungen. [1]

2008 wird als Geburtsjahr der Blockchain-Technologie angesehen. Satoshi Nakamoto hat den Grundstein der Blockchain-Technologie im November 2008 mit seiner Veröffentlichung „Bitcoin: A Peer-to-Peer Electronic Cash System“ gelegt. Im Januar 2009 veröffentlichte er auch schon die erste Open-Source Version des Bitcoin Codes. Open-Source bedeutet, dass es der Code für jedermann einsehbar ist.
Die Identität von Satoshi Nakamoto bleibt ein Mysterium, da dies nur ein Pseudonym ist. Es wird vermutet, dass sich hinter diesem Namen eine Gruppe von Entwicklern verstecken.
Im zeitlichen Rahmen der Finanzkrise 2008 war Bitcoin dafür gedacht den Finanzsektor zu revolutionieren und eine alternative Bezahlungsmöglichkeit zu erschaffen. Diese Kryptowährung basiert auf einem dezentralen und kryptografisch sicheren Bezahlsystem.

> What is needed is an electronic payment system based on cryptographic proof instead of trust, allowing any two willing parties to transact directly with each other without the need for a trusted third party. - Satoshi Nakamoto [2]

Als trusted third party werden hier die Banken der aktuellen Finanzwelt angesprochen, auf welche, durch das neue Zahlungssystem Bitcoin, verzichtet werden kann.
Die Idee eines sicheren dezentralen Systems existierte schon vor der Veröffentlichung Bitcoins, jedoch konnte keiner der vorgestellten Entwürfe bestehen. Hier scheiterte es vor allem an dem Double Spending Problem und der ungenügenden Absicherung gegen eine Sybil Attacke - was genau diese Begriffe bedeuten werde ich im späteren Verlauf des Papers erklären.
Die Blockchain-Technologie jedoch, auf der Bitcoin basiert, bietet ein robustes und sicheres dezentrales System, ohne Beschränkungen der Anzahl der Nutzer und ohne notwendige Identitätsbestätigung. Ebenso biete diese Technologie Schutz gegen das Double Spending Problem und Sybil Attacken.
Hier muss man differenzieren, dass Blockchain eine Technologie ist und Bitcoin ein konkretes System, das diese Technologie für digitale Bezahlungsprozesse nutzt.

Da die Bitcoin Implementation Open-Source ist, kann sie jeder einsehen und für eigene Zwecke abändern und verwenden. Dementsprechend ist die Blockchain nicht auf Kryptowährungen limitiert sondern kann als programmierbare Vertrauensinfrastruktur angesehen werden.
In den folgenden Jahren sind weitere Begriffe hinzugekommen, wie die Distributed Ledger Technologie, was eine Art dezentrales Register ist. Diese Entwicklung wird als Blockchain 2.0 bezeichnet. Ein weiterer und hochaktueller Begriff sind die Smart Contracts - programmierbare autonome Verträge auf Basis der Blockchain. Dies wird als Blockchain 3.0 bezeichnet. [3]

Durch den Erfolg Bitcoins wird es immer deutlicher, dass die Blockchain-Technologie den Finanzsektor revolutionieren wird bzw. es schon tut. Jedoch stoppt diese Revolution nicht dort, sondern breitet sich in andere Anwendungsfelder aus. Darunter gehören z.B. die Medienbranche, der Logistiksektor (Supply Chain), Internet of Things, Identitätsmanagement und vieles mehr. Was genau steckt hinter dieser Blockchain und wie funktioniert sie? Diese Fragen werden in diesem Paper beleuchtet.
Zuerst werden die Probleme eines dezentralen Netzwerkes (also ein Netzwerk ohne Vertrauenspartei) erläutert, um das Verständnis und die Notwendigkeit einer korrekten Blockchain Implementierung zu verdeutlichen. Danach werden die kryptografischen Grundlagen geklärt, die diese Probleme lösen sollen. In der darauf folgenden Sektion wird die grundlegende Funktionsweise der Blockchain am Beispiel Bitcoin dargestellt und dann auf die unverzichtbaren Konsensmechanismen für Blockchains eingegangen. Das letzte Kapitel beschäftigt sich genauer mit Bitcoin und dem dahinter steckenden System von Adressen und Wallets.

2 Blockchain

In diesem Kapitel werde ich die zentralen Eigenschaften und Funktionsweisen einer Blockchain darstellen.

2.1 Probleme eines dezentralen Netzwerks

Um die Probleme besser verstehen zu können, werde ich diese hier an einem Beispiel einer finanziellen Institution, wie einer Bank, erläutern.
Einen Vertrauenspartner bzw. eine zentrale Instanz, wie eine Bank, zu haben ist unglaublich nützlich bei der Durchführung wichtiger Bankgeschäfte. Die Bank hilft zum Beispiel, wenn man aus Versehen eine Transaktion an den falschen Partner getätigt hat oder einen Tippfehler bei der Geldsumme gemacht hat. Hier kann die Bank das Geld nachverfolgen und die Transaktion ungeschehen machen. Bei einem Betrugsvorfall kann die Bank den Vorfall überprüfen und den Konflikt lösen. Es ist also sehr komfortabel eine Partner zu haben, dem man vertraut und welcher zur Not einschreiten kann. Auf der anderen Seite heißt das auch, dass die Bank viele Daten über den Kontoinhaber hat und in manchen Fällen das Konto einfrieren oder bestimmte Transaktionen verbieten kann. Hier muss man der Bank vertrauen, dass sie mit den persönlichen Daten der Kontoinhaber verantwortungsvoll umgeht und sie nicht an Dritte weiterverkauft oder anderweitig verwertet. Dieses Konzept des Vertrauens wird auch als Client-Server Modell bezeichnet. Der Client nutzt die Dienste des Servers, unter der Bedingung, dass der Server die Kontrolle über den Client behält. Dieses Modell wird auch bei vielen anderen Online-Plattformen verwendet, wie z.B. bei Sozialen Netzwerken und kostenlosem Cloud Speicher. In diesen Beispielen vertraut man dem Vertrauenspartner besonders seine persönliche Daten an, mit welchen dieser in dem Großteil der Fälle sein Geld verdient. [3]

Bei einem dezentralen Konzept gibt es keinen Vertrauenspartner. Das heißt auch, dass man hier auf die Vorteile, wie die Überwachung einer Bank, verzichten muss. So ein Konzept lässt sich mittels eines Peer-to-Peer (P2P) Netzwerks realisieren. Das heißt, man interagiert nur mit anderen Benutzern (Peers) des Netzwerks, ohne eine dritte Instanz. Jeder Benutzer ist hier ein Teil des Netzwerks und kann mit jedem anderen Benutzer interagieren. Um Teil des Netzwerks zu werden muss man sich nicht identifizieren oder anderweitige Bedingungen erfüllen, außer die Anwendung des Netzwerks zu installieren. Dementsprechend kennen sich die Nutzer nicht. So ein P2P-Netz wird „permissionless“ genannt. Hier verbirgt schon ein großes Problem: Wie kann man einem anderen Benutzer vertrauen?
Das dezentrale Netzwerk funktioniert so, dass alle Informationen allen Benutzern im Netzwerk zur Verfügung stehen. Bei dem Beispiel einer Bank wären das alle Transaktionsinformation, also welcher Benutzer, wem, wie viel Geld zugesendet hat. Eine Transaktion wäre also dann gültig, wenn die Mehrheit der Benutzer des Netzwerks diese erhalten und bestätigen, dass diese ausgeführt wurde. Also bildet das Netzwerk ein demokratisches Konstrukt, da hier ein simples Mehrheitsgesetz angewendet wird. Man muss aber davon ausgehen, dass es auch hier Benutzer gibt, die betrügen wollen.

Ein Beispiel:
Alice will Bob einen Computer abkaufen. Dafür überweist sie ihm das Geld über

das dezentrale Netzwerk und alle Nutzer können sehen, dass sie das getan hat. Alice kann nun aber behaupten, dass sie das Geld nicht überwiesen hat und damit eine andere Transaktion durchführen. Dies nennt sich das **Double Spending Problem**. Im Normalfall würde so eine Transaktion sofort auffallen und die Nutzer des Netzwerkes können Alice als Betrügerin identifizieren. Was wäre aber, wenn Alice sich mehrere falsche Identitäten im Netzwerk erstellt? Dies ist ohne Probleme möglich, da ja keine besonderen Bedingungen zum Beitritt des Netzwerkes gelten (permissionless). Diese Identitäten werden alle behaupten, dass Alice im Recht liegt und könnten die Mehrheit im Netzwerk bilden, womit die ehrlichen Nutzer überstimmt wären. So ein Vorgehen wird auch als **Sybil Angriff** bezeichnet Es ist also sehr wichtig, dass die ehrlichen Nutzer die Mehrheit des Netzwerks bilden.

Aufgrund dessen muss ein Konsensmechanismus entwickelt werden, welcher aussagt, welche Transaktion gültig ist und welche eine Fälschung. So ein Konsensmechanismus basiert auf folgend kryptografischen Technologien.

2.2 Grundlegende kryptografische Technologien

2.2.1 Hash-Funktionen

Die Verwendung von Hash-Funktionen ist in der Informatik eine alltägliche Angelegenheit und bildet einen der Grundpfeiler auf der die Blockchain Technologie aufbaut. Eine Hash-Funktion ist eine mathematische Funktion die einen beliebig langen Input, z.B. Daten oder Text, auf einen Output (auch Hash-Wert genannt) fester Länge abbildet, welcher jedoch zufällig gewürfelt aussieht. Eine kleine Änderung beim Input (z.B. einen Buchstaben ändern), ergibt eine Große Änderung im Hash-Wert, so dass der neue Hash-Wert nicht auf den alten rückschließen lässt (Avalanche-Effekt). Bei kryptografischen Hash-Funktionen, welche auch bei Bitcoin zum Einsatz kommen, muss es praktisch nicht möglich sein, aus dem Hash-Wert den Input zu ermitteln. Hash-Funktionen sind deterministisch, was bedeutet, dass der selbe Input immer den selben Hash-Wert ergibt. Eine weitere Eigenschaft ist ihre Effizienz - Hash-Funktionen brauchen wenig Rechenpower und können leicht in Software implementiert werden.
Solche Hash-Funktionen finden ihren Gebrauch zum Beispiel bei der Integritätsprüfung von Dokumenten. Bei der Übermittlung eines Dokuments, z.B. per Email, kann man sicher gehen, dass man das richtige, unveränderte Dokument erhalten hat, indem man den Hash-Wertes des Dokuments vor der Übertragung kennt und mit dem Hash-Wert des Dokuments nach der Übertragung vergleicht.

Mathematisch beschrieben: Ist H die Hash-Funktion, h der Hash-Wert und d der Input, dann gilt:

$$h = H(d)$$

Mengentheoretisch beschrieben: Ist I unsere mögliche Inputmenge und O unsere mögliche Hash-Werte Menge, dann gilt:

$$h : I \to O$$

Wobei $|I| > |O|$ und $|O|$ fix ist. Da der Output eine feste Länge hat und die Inputs beliebig groß sind, ist es möglich, dass Kollisionen auftreten, also zwei Inputs den selben Hash-Wert ergeben. Dies wäre problematisch, da so z.B. die Integritätsprüfung bei dem vorher genannten Beispiel der Dokumentübertragung nicht mehr helfen würde. Deshalb muss eine kryptografisch sichere Hash-Funktion stark kollisionsresistent sein. Die bei Bitcoin verwendete Hash-Funktion SHA-256 erfüllt diese Eigenschaft. Sie bildet einen beliebigen Input auf eine Zahl mit der Länge

von 256 Bit ab. Es gibt also 2^{256} Möglichkeiten den Eingabewert abzubilden. Um diese Zahl in Kontrast zu setzen: Es gibt ca. 10^{22} – 10^{24} Sterne im Universum, was eine viel geringere Zahl ist. Dementsprechend ist der Outputbereich ausreichend groß, um Kollisionen zu vermeiden. Eine Kollision zu suchen ist unpraktikabel, da man viel Rechenleistung aufwenden müsste. [4]

Beispiel: Hier wurde die SHA-256 Hash-Funktion in Windows Powershell auf einem Dokument angewendet. Der Hash-Wert ist in Hexadezimal angegeben.

```
PS C:\Users\blank\Documents> Get-FileHash .\TestDokument.txt -Algorithm SHA256

Algorithm       Hash
---------       ----
SHA256          E3B0C44298FC1C149AFBF4C8996FB92427AE41E4649B934CA495991B7852B855
```

Abbildung 2.1: SHA-256 in Windows Powershell

2.2.2 Hash-Bäume

Eine weiterführende Anwendung der Hash-Funktionen sind Hash-Bäume oder auch Merkle-Bäume (nach Ralph Merkle benannt). Dies ist eine Baum-Datenstruktur, welche als Blätter Hash-Werte von Daten enthält. Sie kann zum Beispiel dafür verwendet werden, die Integrität von mehreren Dateien nachzuweisen. Es werden die Hash-Werte von allen Dateien gebildet, welche die untersten Blätter bilden. Die Knoten darüber bilden den Hash-Wert der Konkatenation der Hash-Werte der Blätter. Meist wird dies als Binärbaum dargestellt, wo immer von zwei Blättern die Hash-Werte konkateniert werden und davon der Hash-Wert gebildet wird. Dies wird weitergeführt, bis ein oberster Wurzel Hash berechnet wird. Diese Wurzel wird unter anderem als Merkle-Root bezeichnet und ist als oberster Hash von allen Dokumenten des Baums abhängig. Dementsprechend kann dieser Hash dafür verwendet werden, die Integrität aller Daten des Baumes zu sichern. Wenn eine Datei geändert wird, dann ändert sich auch die Merkle-Root. Die Versionsverwaltung Git implementiert solche Hash-Bäume, sowie natürlich die Blockchain Technologie von Bitcoin. [5]

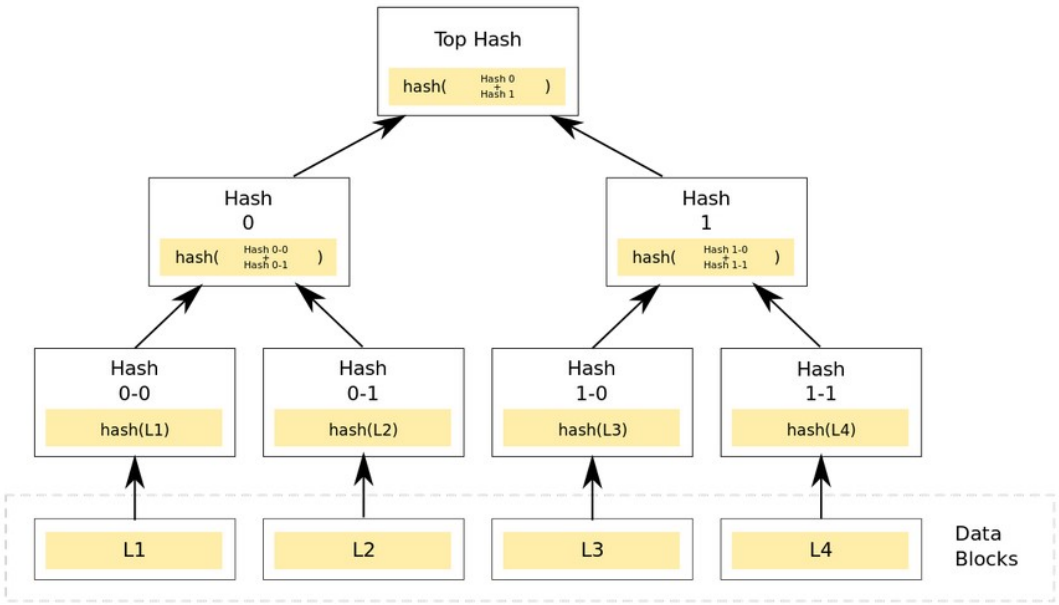

Abbildung 2.2: Quelle: Wikipedia - Hash-Baum

2.2.3 Digitale Signaturen

Digitale Signaturen sind ein Verfahren, basierend auf asymmetrischer Kryptografie, um die Authentizität und Integrität von Nachrichten (bzw. von allen möglichen Daten) darzustellen. In einem asymmetrischen System besitzt jeder Akteur zwei Schlüssel. Einmal den privaten Schlüssel, den er nur für sich behält und einmal den öffentlichen Schlüssel, der für die anderen Akteure einsehbar ist z.B. im Internet oder auf einem öffentlichen Dateisystem eines Zertifizierungsdiensteanbieters. So ein Anbieter korreliert die öffentlichen Schlüssel zu den Identitäten der Personen zu und dient als Vertrauensinstanz.

Nun zu der Funktionsweise von digitalen Signaturen an einem Beispiel eines zu signierenden Dokument: Wenn die Unterzeichnerin, Alice, dem Prüfer, Bob, ein signiertes Dokument schicken will, erstellt sie sich zunächst ein Schlüsselpaar mithilfe eines Schlüsselgenerierungsalgorithmus, welcher ihr den privaten und den korrespondierenden öffentlichen Schlüssel erstellt. Danach berechnet sie den Hash-Wert des zu sendenden Dokuments mit einer Hash-Funktion. Um das Dokument eindeutig und unwiderruflich zu signieren, verschlüsselt Alice den Hash-Wert mit ihrem privaten Schlüssel. Dieses „verschlüsseln“ wird durch die Signaturfunktion erledigt. Diese Signaturfunktion nimmt als Input den Hash-Wert des Dokuments und den privaten Schlüssel von Alice und berechnet die Signatur. Diese Signatur kann nur durch den öffentlichen Schlüssel von Alice entschlüsselt werden. Außerdem muss die Signaturfunktion es einem Angreifer praktisch unmöglich machen, den privaten Schlüssel von Alice allein aus der resultierenden Signatur zu berechnen. Somit ist die digitale Unterschrift eindeutig zu Alice zuzuordnen.
Sei SigFunc die Signaturfunktion, d das zu signierende Dokument, p der private Schlüssel von Alice und S die Signatur:

$$SigFunc(H(d), p) = S$$

Nun schickt sie das Dokument (im Klartext) mit der Signatur an Bob. Um sicherzustellen, dass das Dokument auch wirklich von Alice kommt, entschlüsselt Bob die Signatur von Alice mit ihrem öffentlichen Schlüssel. Nun hat Bob den Hash-Wert, den Alice berechnet hat und berechnet nochmal den Hash-Wert des erhaltenen Dokuments und vergleicht diese. Die Signatur ist gültig, wenn die Hash-Werte übereinstimmen. [6]

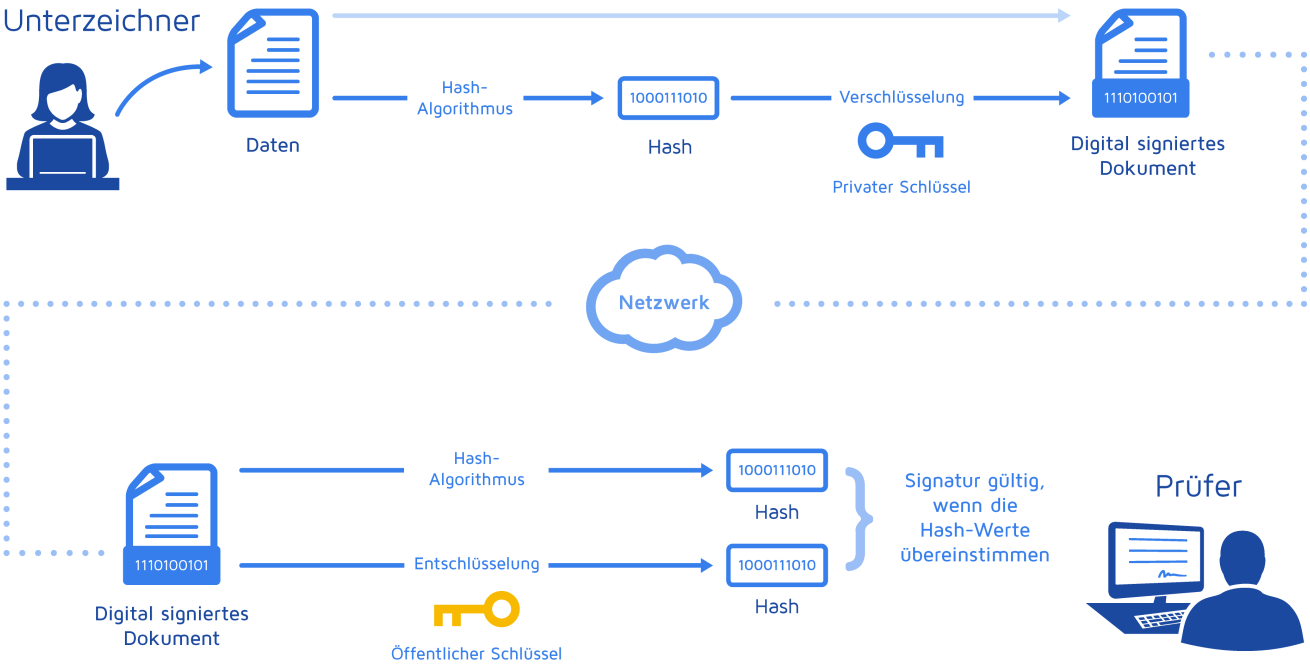

Abbildung 2.3: Quelle: docusign.de

2.3 Grundlegende Funktionsweise und Eigenschaften

Da nun die kryptografischen Grundlagen geklärt sind, können wir uns mit der genauen Funktionsweise und den Eigenschaften der Blockchain-Technologie beschäftigen. Da das Konzept einer Blockchain keine genaue praktische Definition hat, werde ich die einzelnen Aspekte an dem Beispiel der Kryptowährung Bitcoin erläutern. Man sollte jedoch im Hinterkopf behalten, dass diese grundlegenden Konstrukte auch für alle anderen möglichen Bereiche angewendet werden können.

Wie schon in der Einleitung erwähnt, ist die Blockchain ein Mittel zur **dezentralen Datenspeicherung**. Die Blockchain wird bei jedem Beteiligten des P2P-Netzwerkes gespeichert und wird deshalb auch oft als dezentrale Datenbank bezeichnet. Dort kann man Daten speichern, die für alle Beteiligten einsehbar sind und somit die Eigenschaft der **Transparenz** erfüllen.
Die Kryptowährung Bitcoin speichert in der Blockchain die Überweisungen (Transaktionen) der Nutzer. Diese Liste der Transaktionen wird auch oft als „Ledger“ (zu deutsch: Geschäftsbuch) bezeichnet. Da jeder diese Liste einsehen kann, weiß auch jeder wie viel Bitcoin auf den einzelnen Konten („Wallets“ - Bezeichnung für Kryptowährungskonten) vorhanden sind. Dementsprechend gibt es, wie oft falsch angenommen, auch keine physischen Bitcoins, die sich im Wallet befinden. Der Wert des Wallets richtet sich nach den eingegangenen und ausgegangenen Transaktionen, welche in der Blockchain festgehalten werden.

Vereinfacht funktioniert das Senden einer Transaktion so: Sie wird von einem Sender im Netzwerk erzeugt und auch **digital signiert** (siehe Kapitel 2.2.3), damit die Transaktionen eindeutig einem Sender zugeordnet und nicht gefälscht werden können. Somit ist **Nichtabstreitbarkeit** gewährleistet und das erste Problem eines dezentralen Netzwerkes gelöst.

> **Angriff auf die digitale Signatur in Bitcoin:** Wenn ein böswilliger Nutzer trotzdem so eine Transaktion im Namen (Wallets) eines anderen Nutzers ausführen will, muss er den privaten Schlüssel des Wallets kennen. Dann kann er eine neue Transaktion digital signieren und an das Blockchain-Netzwerk schicken. Den privaten Schlüssel eines Wallets kann man nur herausfinden, indem man ihn „stiehlt“ oder versucht, die Signatur einer anderen Transaktion des Kontos zu „knacken“. Letzteres ist nur möglich, indem man durch stumpfes Ausprobieren verschiedener Werte (Brute Force) versucht, die Signatur der Transaktion nachzubilden und so den privaten Schlüssel zu erhalten.

Nun sendet der Sender die signierte Transaktion an das Netzwerk bzw. an alle Knoten, die für die Konsensfindung zuständig sind. [7] Dies können alle Teilnehmer des Blockchain-Netzwerkes sein oder ausgewählte Teilnehmer, die die Konsensfindung übernehmen. In dem Bitcoin-Netzwerk sind das die sogenannten „Miner“. Diese Teilnehmer empfangen alle Transaktionen, die in der Blockchain getätigt werden und sammeln eine bestimmte Anzahl, um daraus einen Block zu bilden. Dieser Block wird nach der Konsensfindung zu der Blockchain hinzugefügt und von allen Teilnehmern im Blockchain-Netzwerk als gültig anerkannt. Es gilt nur das, was in der Blockchain festgehalten wurde und von allen Teilnehmern anerkannt wurde. Wie genau die Konsensfindung funktioniert wird im nächsten Kapitel erläutert. Durch die immer neuen Transaktionen wird die Blockchain fortlaufend größer und ist unendlich skalierbar.

Das Herz der Blockchain sind die darin enthaltenen Blöcke, welche auch als Datenstruktur angesehen werden können. Die Blöcke enthalten die Daten der jeweiligen Implementierung der Blockchain, wie die Transaktionsinformationen im Falle von Bitcoin. Die Transaktionen sind sehr kompakt gespeichert und enthalten die notwendigsten Information. Die Bitcoin-Transaktionen

enthalten mindestens eine Eingabe, bestehend aus den Transaktionen bzw. den Transaktionshashes, aus denen man die zu überweisenden Bitcoin erhalten hat, und eine Ausgabe, bestehend aus der Empfängeradresse und dem entsprechenden Betrag, der zu überweisen ist.
Sobald ein Block „befüllt“ ist, wird aus den Daten der Transaktionen, immer paarweise, ein Merkle-Baum (siehe Kapitel 2.2.2) gehasht. Der Root-Hash wird im Header des Blocks abgespeichert. Der Header enthält mehrere Metainformationen des Blocks. In dem Header eines Bitcoin Blocks stehen die aktuelle Bitcoin-Version, der Hash-Wert des vorigen Block-Headers, ein Target (die Schwierigkeit des kryptografischen Rätsels der Konsensfindung), ein Zeitstempel und eine Nonce (Number used only once - die Lösung des kryptografischen Rätsels, welche als Bestätigung dafür gilt, dass der Block gültig ist). Vor allem der Zeitstempel, der Hash-Wert des vorigen Blocks und die Nonce finden sich in anderen Blockchain Implementationen wieder. Da sich in jedem Block der Hash des vorigen Blocks befindet, sind alle Blöcke, durch die Eigenschaften der Hash-Funktion miteinander verbunden. Wenn man einen schon bestehenden Block in der Blockchain ändern wollen würde, würden sich auch alle anderen Blöcke nach diesem, verändern, da die Hash-Werte sich verändern. Diese Verbindung durch die Hash-Werte im Block-Header wird auch als **Verkettungsprinzip** bezeichnet. Diese Metainformationen sind wichtig um **Manipulationssicherheit** zu gewähren.

> **Manipulation im Falle Bitcoin:** Wenn man eine Transaktion, die schon geschehen ist ändern würde, würde sich der Root-Hash des Blocks, welcher die Transaktion enthält, auch ändern und somit alle anderen Blöcke auch. Dies würde, nach mehreren Überprüfungen des Konsensmechanismus in Bitcoin, ausgemacht und als ungültig erklärt werden.

Die folgende Abbildung zeigt den Aufbau der Bitcoin-Blockchain. Man sieht einzelne Blöcke, welche die Header-Information und Transaktionen enthalten. Die Verkettung wird durch die Pfeile des vorigen Block-Hashes auf den vorigen Block-Header dargestellt.

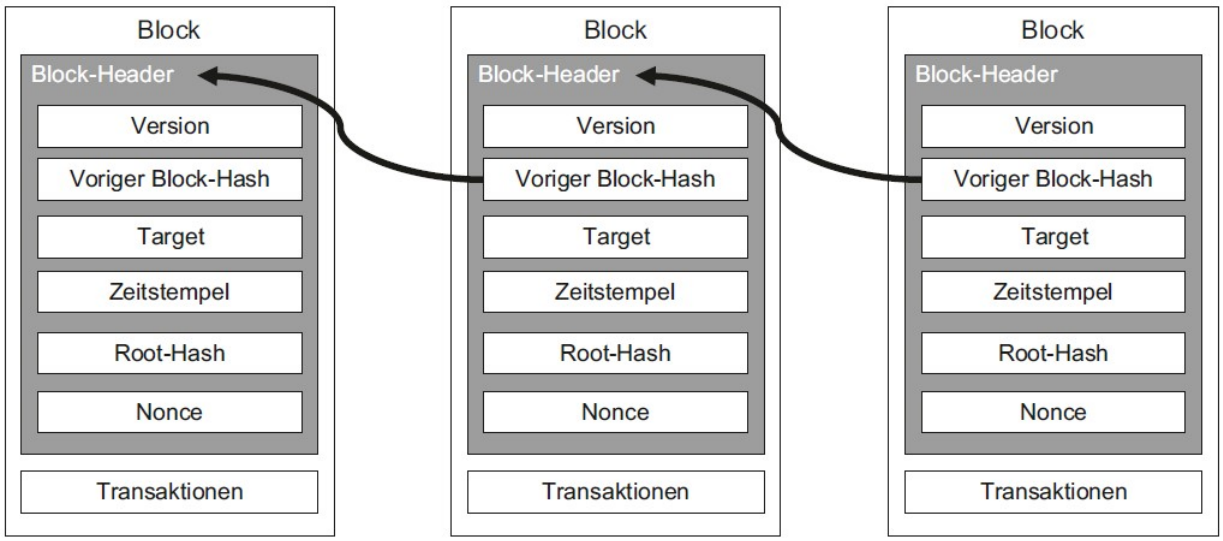

Abbildung 2.4: Verkettung von Bitcoin-Blöcken, Quelle: Hans-Georg Fill & Andreas Meier: Blockchain, Springer Verlag, 2020. S.12

Wie die Blockchain die **Transparenz**, die **Nichtabstreitbarkeit** und die **Manipulationssicherheit** gewährt, wurde nun erläutert. Damit werden die Probleme, welche in Kapitel 2.1 beschrieben wurden, weitestgehend gelöst.
Eines der wichtigsten Probleme eines dezentralen Netzwerks wurde jedoch noch nicht behandelt - die **Konsensfindung**:

> Die Teilnehmer des Blockchain-Netzwerkes bauen nun ihre eigenen Blöcke und Blockchains auf. Wie kann man sicher gehen, dass alle Teilnehmer die selbe Blockchain bauen, welche die korrekten Transaktionen (Blöcke) in der richtigen Reihenfolge enthält? Alle Teilnehmer müssen sich auf eine Blockchain - einen Konsens - einigen, welcher dann auch allgemeingültig ist. So einen Konsens zu finden wird, durch eventuelle falsche Informationen böswilliger Teilnehmer, stark erschwert.

Hierfür muss jede Blockchain einen Konsensmechanismus besitzen, der beweist, dass die Transaktionen in den Blöcken gültig und in der richtigen Reihenfolge sind, so dass alle Teilnehmer des Netzwerks darauf **Vertrauen** können. Wie dieses Problem gelöst wird, wird im nächsten Kapitel dargestellt.

2.4 Konsensmechanismen

Das Problem der Konsensfindung wird auch als „**Problem der byzantinischen Generäle**“ bezeichnet. Dies ist ein Problem in der Informatik, bei dem ein Gleichnis zwischen einem Kommunikationsproblem bei der Belagerung einer Stadt und dem Problem der Konsensfindung in einem dezentralen Netzwerk, gezogen wird:

> Bei der Belagerung einer Stadt haben die byzantinischen Generäle ein Kommunikationsproblem. Sie wollen aus verschiedenen Richtungen angreifen und haben deshalb mehrere Lager aufgeschlagen. Die Generäle haben sich auf die Lager aufgeteilt und kommunizieren über Boten miteinander. Allerdings gibt es einige Generäle, die gegen andere intrigieren und somit ihre Konkurrenten mit geschickter Streuung von Falschinformationen fehlleiten wollen. Um einen erfolgreiche Eroberung durchzuführen, müssen die Generäle gleichzeitig den Befehl zum Angriff geben, um ihre Stärke zu bündeln. Dies kann an den Falschinformationen eines böswilligen Generals scheitern. Wie können sich die loyalen Generäle untereinander trauen? Die Lösung ist ein Algorithmus der den loyalen Generälen hilft, sich auf einen Konsens zu einigen, obwohl mögliche Verräter versuchen diesen zu beeinflussen.[3]

Je mehr böswillige Nutzer ein Netzwerk enthalten kann, desto robuster muss auch so ein Algorithmus sein. In der Vergangenheit waren Konsenslösungen für dezentrale Netzwerke abhängig von einigen Faktoren, wie z.B. von der Anzahl und der Identitätsprüfung der Teilnehmer so eines Netzwerks (so ein Netzwerk wid auch als „permissioned“ bezeichnet). Im heutigen Internet („permissionless“), was ein dezentrales Netzwerk ist, wäre so eine Identitätsprüfung sehr ineffizient. Der Konsensmechanismus im Bitcoin Netzwerk (auch **Nakamoto Konsensmechanismus**) funktioniert auch in Netzwerken ohne solche Bedingungen. Nutzer können nach freiem Willen dem Netzwerk beitreten und dieses auch wieder verlassen. Dieser Konsensmechanismus basiert auf dem Proof-of-Work Konzept, auf welches ich nun eingehen werde.

2.4.1 Proof-of-Work (PoW)

Der Nakatomoto Konsensmechanismus geht davon aus, dass in einem dezentralen Netzwerk ohne Beitrittsbeschränkungen (ein Nutzer kann mehrere Identitäten erstellen), der Großteil der **Rechenleistung** von den ehrlichen Nutzern ausgeht und nicht, dass der Großteil der Nutzer ehrlich ist. Dieses Konzept nennt sich Proof-of-Work.
Das bedeutet, dass der Nutzer, der am schnellsten eine aufwendige Rechenaufgabe löst, das Recht erhält die Entscheidung zu treffen. In der Bitcoin-Implementation besteht die Aufgabe aus einem kryptografischen Hash-Rätsel.

Nakomoto Konsensmechanismus: Die Frage, die bei Bitcoin gestellt wird, ist: Wer darf seinen Block mit den darin enthaltenen Transaktionen an die allgemeingültige Blockchain hängen?:
Derjenige, der einen bestimmten Hash-Wert berechnet, also seine Rechenleistung aufwendet, darf seinen Block anhängen. Diese Hash-Wertberechnung ist ein kryptografisches Rätsel, welches gelöst werden soll. Wie schon im vorherigen Kapitel erwähnt, gibt es den Target-Wert und die Nonce in einem Block. Der Target-Wert wird vorgegeben und gibt an, wie viele führende Nullen der Hash-Wert einer Hashberechnung, aus dem Merkle-Hash der Transaktionen und einer zufällig ausprobierten Zahl (der Nonce) besitzen soll. Demnach müssen die Teilnehmer alle möglichen Nonces ausprobieren (Brute-Force-Methode), bis sie eine Nonce finden, die verhasht mit dem Merkle-Hash der Transaktionen, dem Ziel-Wert entspricht. Also ist der Nutzer, der mit leistungsstarker Hardware am schnellsten alle möglichen Zahlen mit dem Merkle-Hash der Transaktionen verhasht der und somit den Ziel-Wert erreicht, der den Block an die allgemeingültige Blockchain hängen darf. Da hier Rechenaufwand geleistet wird, nennt sich dieses Konzept Proof-of-Work.
Sobald ein Nutzer den Target-Wert erreicht, schickt er seinen Block bzw. seine Version der Blockchain, mit der Lösung des Rätsels (der Nonce), an alle anderen Nutzer (Broadcast). Diese können dann ohne großen Aufwand überprüfen, ob die Rechenleistung aufgewandt wurde, indem sie ebenso den Merkle-Hash mit der Nonce verhashen und vergleichen, ob das Target erreicht wurde.
Es besteht jedoch immer noch die Möglichkeit, dass mehrere Nutzer für verschiedene Blöcke die Ziel-Werte fast gleichzeitig erreichen und so die Frage entsteht, welcher Blockchain nun getraut werden kann (dieses Abzweigen der Blockchain wird auch als Branching bzw. Forking bezeichnet). Da nun parallel auf zwei Blockchains weiter gearbeitet wird, müssen die Nutzer des Netzwerkes darauf warten, welche Blockchain sich durchsetzt. Jeder angehängte Block ist eine weitere Validierung, dass auf der richtigen Blockchain gearbeitet wird. Das PoW Konzept geht davon aus, dass die allgemeingültige und richtige Blockchain die größte Rechenpower hat und dementsprechend auch die Längste ist. Also werden die Nutzer immer die längste Blockchain als die allgemeingültige Blockchain ansehen.
Um den Ziel-Wert des Rätsels zu erreichen, braucht man hocheffiziente Hardware, welche z.B. 120 Millionen Hash-werte in einer Sekunde berechnen kann[1]. Dementsprechend ist der Energieverbrauch dieser Hardware um einiges höher als im Normalfall. Da dieser Prozess unerlässlich für das Bitcoin-Netzwerk ist, muss es eine Motivation geben, damit weiterhin Transaktionen validiert werden: Als Belohnung für den Rechenaufwand bekommt der „Miner“ (Bezeichnung für die Block-Ersteller) eine Belohnung in Form von neu geschaffenen Bitcoins. Der Miner darf also einfach eine Transaktion zu der Liste der Transaktionen hinzufügen, welche ihm einen bestimmten Betrag an Bitcoin überweist. Diese Bitcoins haben keinen Absender und werden somit aus dem „Nichts“ erschaffen. Auf diesen Weg kommen neue Bitcoin in das Netzwerk. Dieser Prozess wird oft mit dem Abbau von Gold verglichen und hat daher seinen Namen verliehen bekommen - Mining. Um einen Bitcoin Block zu minen braucht ein Miner im Durchschnitt zehn Minuten. Der Target-Wert für den nächsten Block wird dahergehend bestimmt, wie lange die Miner gebraucht haben, um den letzten Block zu minen. Wenn weniger als zehn Minuten gebraucht wurden, wird das Rätsel erschwert und andernfalls erleichtert. So bleibt die durchschnittliche Zeit für eine Validierung der Blockchain bei zehn Minuten.
Mit dem Erfolg Bitcoins bildeten sich auch immer professionellere Miner, welche

[1] Z.B: mit der Grafikkarte NVIDIA GeForce RTX 3090, welche den Ethereum Algorithmus berechnet

den konventionellen „Hobby-Miner“ von der Rechenleistung übertreffen. Es gibt zum einen Mining-Pools, wo sich Miner ihre Rechenleistung teilen, um im Kollektiv Hash-Werte zu berechnen und zum anderen gibt es Mining-Farmen, was zum Teil riesige Rechenzentren sind, welche zum Bitcoin-Mining verwendet werden. [3]

Ein unüberschaubarer Nachteil ist der extrem hohe Energieverbrauch solcher Mining-Pools/Farmen, welche vor allem in Ländern mit niedrigen Stromkosten zu finden sind. Laut dem Cambridge Bitcoin Electricity Consumption Index verbraucht das gesamte Bitcoin-Netzwerk mittlerweile so viel Strom, wie das gesamte Land Norwegen. [8] Der folgende Konsensmechanismus basiert nicht auf Rechenleistung und ist deswegen eine „grüne“ Alternative.

2.4.2 Proof-of-Stake (PoS)

Der Proof-of-Stake Konsensmechanismus basiert auf der prozentualen Verteilung von „Coins“ einer Kryptowährung (Stakes), welche die Teilnehmer des Netzwerkes besitzen. Je mehr digitale Coins ein Teilnehmer einer Kryptowährung besitzt, desto mehr Blöcke darf er erstellen. Jedoch verwenden viele Implementierungen von Proof-of-Stake abgeänderte Methoden, aufgrund von einigen Problemen die mit diesem Konsensmechanismus einhergehen.
Mit dem ursprünglichen PoS-Konzept ergibt sich schnell ein ähnliches Problem, wie beim PoW. Wenn zwei Minter (Bezeichnung der Miner in PoS) unterschiedliche Blöcke an die Blockchain anhängen wollen, teilt sich die Blockchain sich in zwei „Zweige“ auf (Forking bzw. Branching) und die Minter bauen parallel auf ihren Zweigen weiter, ohne dabei Aufwand zu erbringen, da PoS-Konzepte kaum Rechenleistung benötigen. Aufgrund dessen besteht auch hier die Möglichkeit des Double-Spending von digitalen Coins. Da hier der zu erbringende Aufwand viel geringerer ist, als bei dem Proof-of-Work Konzept, bietet dieser Konsensmechanismus auch eine viel größere Angriffsfläche, die bei einer Implementierung zu umgehen ist.
Dieses Problem wird mit einer Modifizierung des PoS-Konzepts, auch bekannt als delegiertes PoS (DPoS), gelöst. Hier werden bestimmte Teilnehmer als Vertrauenspersonen (Delegierte) ausgewählt. Dies geschieht anhand von spezifischen Kriterien, wie z.B. durch Abstimmungen der anderen Nutzer (die Stimmkraft ist ebenso abhängig von dem Anteil der Coins eines Teilnehmers). Die Delegierten sind befähigt zum Minten und zum verifizieren von Blocks, welche von anderen Minter erschaffen worden. Damit ein neuer Block erstellt werden kann, müssen mehrere Delegierte diesen verifizieren. In manchen Implementierungen von DPoS müssen die Minter ihre Coins in einem besonderen Sicherheitsaccount anlegen, welcher gesperrt werden kann, falls böswilliges Verhalten des Minters festgestellt wird. Dies ist eine weitere Methode um Angriffen entgegenzuwirken.

Eine Implementierung von DPoS ist die Kryptowährung **Bitshares**. Das Konzept liegt hier bei der Auswahl von sogenannten Zeugen. Die Nutzer von Bitshares wählen eine ‚vorher ebenso gewählte, Anzahl von Zeugen, welche befähigt sind, Blöcke zu minten. Jeder Nutzer hat eine Anzahl von Stimmen proportional zu dem Betrag von BitShares in seinem Besitz.
Ähnlich zu den Zeugen wählen die Nutzer auch Delegierte, welche die Privilegien haben, bestimmte Parameter des Netzwerkes zu ändern. Dazu gehören Transaktionsgebühren, Blockgrößen und -intervalle und die Belohnung für die Zeugen. Anders als die Zeugen bekommen die Delegierten keine Belohnung. [9]

3 Bitcoin - Adressen und Wallets

Da nun die grundsätzliche Funktionsweise einer Blockchain und Bitcoins geklärt sind, möchte ich in diesem Kapitel spezifisch auf weitere kryptografische Aspekte der Bitcoin Implementierung eingehen. Interessant ist hier die Erstellung von Adressen und der Aufbau von sogenannten Wallets.
Um ein Bitcoin-Konto zu erstellen wird ein privater und ein öffentlicher kryptografischer Schlüssel benötigt. Wie schon festgestellt wurde, wird jede Bitcoin Transaktion mit einer digitalen Signatur (mithilfe des privaten Schlüssels) unterschrieben. Dementsprechend kann jeder der den privaten Schlüssel besitzt, Transaktion im Namen dieses Bitcoin Kontos durchführen. Um Bitcoin-Konten zu adressieren, wird aus dem öffentlichen Schlüssel eine Adresse abgeleitet. Wie das Generieren dieser beiden Schlüssel und einer Bitcoin Adresse funktioniert, wird im nächsten Kapitel behandelt.

3.1 Adressen

Um einen **privaten Schlüssel** zu erstellen muss man sich zunächst eine echte Zufallszahl zwischen 1 und 2^{256} generieren. Hier ist es wichtig eine sichere Entropiequelle zu finden, damit der private Schlüssel nicht nachproduziert werden kann. Die Bitcoin Software nutzt den Zufallsgenerator des jeweiligen Betriebssystems, um eine maximal 256 Bit große Zahl zu generieren. Diese Zahl wird dann mit der kryptografisch sicheren SHA256 Hash-Funktion verhasht und ergibt den privaten Schlüssel.

$$PrivateKey = SHA256(RandomNumber)$$

Der **öffentliche Schlüssel** wird aus dem privaten Key mithilfe von elliptischer Kurvenmultiplikation berechnet. Eine elliptische Kurve ist eine mathematische Funktion.

$$PublicKey = PrivateKey * G$$

G ist der Generator Punkt, ein konstanter Punkt auf der elliptischen Kurve, der für alle Bitcoin Nutzer gleicht ist. Bitcoin verwendet eine spezifische elliptische Kurve und mathematische Konstanten, welche durch den secp256k1-Standard definiert werden.
Die elliptische Kurven Kryptografie würde den Rahmen dieses Papers sprengen, weshalb ich hier nur kurz auf die wichtigsten Aspekte eingehen werde.
Das Ergebnis der Multiplikation ist ein weiterer Punkt auf der elliptischen Kurve und ergibt den öffentlichen Schlüssel. Da dieser Schlüssel öffentlich ist, darf es nicht möglich sein, aus diesem wieder den privaten Schlüssel zu berechnen. Diese Umkehrfunktion der elliptischen Kurvenmultiplikation, um den privaten Schlüssel zu bestimmen, wird auch als Finden des diskreten Logarithmus bezeichnet (Diskretes Logarithmus Problem). Dies ist eines der wichtigsten Mittel in der Kryptografie, um Funktion unumkehrbar zu machen. So eine Berechnung ist nicht praktikabel, da diese zu lange dauern würde.
Da das Ergebnis einer der elliptischen Kurvenmultiplikation ein Punkt ist, gibt es eine X-Koordinate und eine Y-Koordinate. Diese Koordinaten werden konkateniert und ergeben den öffentliche Schlüssel.

Aus dem öffentlichen Schlüssel wird nun die **Bitcoin-Adresse** abgeleitet. Hierzu werden wieder Hash-Funktionen verwendet. Aus dem öffentlichen Schlüssel wird zunächst der SHA-256 Hash gebildet und das Ergebnis wird mit dem RIPEMD160 Hash-Algorithmus verhasht. Dann wird noch die Bitcoin Versionsnummer angehängt.

$$PublicKeyHash = RIPEDMD160(SHA256(PublicKey)) + Versionnumber$$

Um diesen Hash (160 Bits + Versionssnummer) zu komprimieren und in ein für Menschen leserliches Format zu bringen wird die Base58Check-Funktion verwendet. Hierfür wird eine Prüfsumme aus dem obigen Ergebnis gebildet, indem dieses zweimal mit SHA256 verhasht wird und davon die ersten vier Bytes genommen werden.

$$Checksum = First4Bytesof(SHA256(SHA256(PublicKeyHash))))$$

Diese Prüfsumme wird an das obige Ergebnis angehängt und darauf wird die Base58-Funktion angewendet. Das Ergebnis ist die Bitcoin-Adresse A.

$$A = Base58(PublicKeyHash + Checksum)$$

Ein Beispiel für so eine Adresse wäre:
bc1qxy2kgdygjrsqtzq2n0yrf2493p83kkfbaa0wlh [10]

3.2 Wallets

Bitcoin Wallets enthalten mehrere Schlüssel und nicht, wie oft falsch angenommen, die physischen Bitcoin. Jeder Bitcoin-Nutzer hat also ein Wallet, was eine Sammlung von Schlüsselpaaren ist. Wieso gibt es nun mehrere Schlüsselpaare bzw. Adressen und nicht nur eine? Dies liegt daran, dass man bei einer einzigen Adresse schlecht unterscheiden kann, welche Zahlung zu welcher Person zugeordnet werden soll. Wenn eine Adresse für eine Transaktion bzw. eine Zahlung verwendet wird, ist sofort klar, dass diese Zahlung getätigt wurde. Z.B. kann so ein Verkäufer, bei dem mehrere Kunden ein Produkt kaufen, sicher sein, welcher Kunde welche Zahlung schon getätigt hat und dementsprechend die Produkte zu verschicken. [11]
Außerdem bietet die Nutzung von mehreren Schlüsselpaaren eine zusätzliche Sicherheit. Wenn alle Transaktionen an eine Bitcoin-Adresse gehen, hängt auch die Nutzung der darin enthaltenen Bitcoin allein von einem privaten Schlüssel ab. Falls dieser verloren geht oder gestohlen wird, sind auch alle Bitcoin weg.
Es gibt unterschiedliche Arten von Wallets, welche davon verwendet werden hängt von dem installierten Bitcoin-Client ab. Der Bitcoin-Client ist eine Software zum erstellen von Wallets und zum Senden und Empfangen von Transaktionen. Eines der weitverbreitesten Bitcoin-Clients ist die Software Electrum.

3.2.1 Nicht-deterministiche oder Random Wallets

In solchen Wallets werden zufällig und unabhängig generierte Schlüssel abgespeichert. Pro Transaktion wird so ein Schlüssel erstellt und muss auch abgespeichert werden, was bei Nutzern mit vielen Transaktionen schnell unübersichtlich wird. Deswegen wird ein Nicht-deterministisches Wallet auch als „Just a Bunch of Keys“ bzw. JBOK bezeichnet.
Da alle Schlüssel einzeln abgespeichert werden, muss oft ein Back-Up erstellt werden. Falls einer der privaten Schlüssel der Transaktionen verloren geht, werden auch die damit verbundenen Bitcoin unbrauchbar.
Aufgrund der Nachteile und besseren Alternativen wird davon abgeraten so ein Wallet zu verwenden.[10]

3.2.2 Deterministische oder Seeded Wallets

Bei einem deterministischen oder auch Seeded Wallet werden alle privaten Schlüssel von einem „Seed“ abgeleitet, indem eine Hash-Funktion verwendet wird. Der Seed ist eine zufällig gewählte Nummer, welcher mit anderen Daten, wie z.B. einer Index Nummer (je Transaktion) kombiniert wird, um private Schlüssel abzuleiten. Da hier eine Hash-Funktion verwendet wird und die privaten Schlüssel, ähnlich wie in der Blockchain, miteinander verkettet sind, genügt der Seed um alle abgeleiteten Schlüssel wieder zu erstellen. Dementsprechend kann der Seed auch in andere Wallet Implementierungen importiert werden und somit Zugang zu den mit dem Seed assoziierten Bitcoins gewähren.[10]

3.2.3 Hierarchisch-Deterministische Wallets

Diese Form von Wallets ist eine Weiterentwicklung der deterministischen Wallets, welche die Organisation und den Umgang mit den privaten Schlüsseln um einiges erleichtert. Ein Hierarchisch-Deterministisches Wallet (kurz HD-Wallet) bildet eine Hierarchie von privaten Schlüsseln. Nämlich wird hier aus dem Seed ein Master Schlüssel abgeleitet, aus welchem mehrere Kinder Schlüssel und daraus Enkel Schlüssel abgeleitet werden. Diesen Zusammenhang kann man auch in einem Baum darstellen:

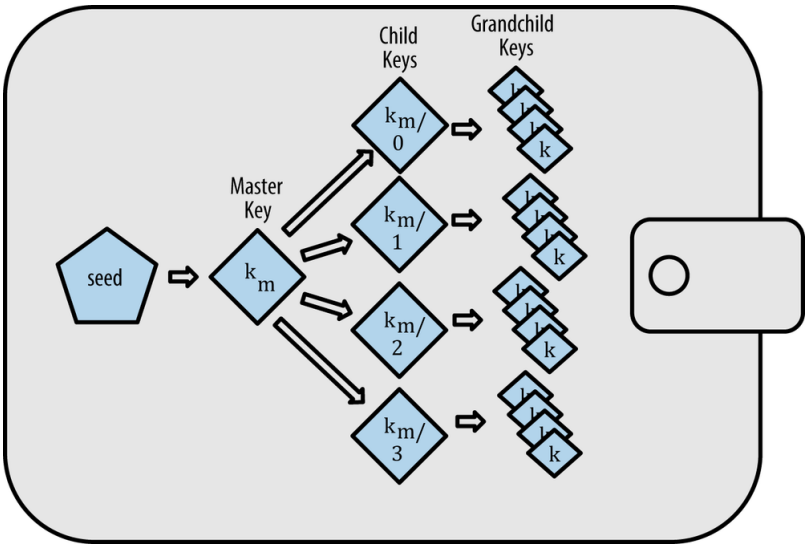

Abbildung 3.1: Quelle:https://www.oreilly.com/library/view/mastering-bitcoin/9781491902639/ch04.html

So ein HD-Wallet hat mehrere Vorteile gegenüber den anderen beiden Wallet-Varianten. Durch die Möglichkeit, Schlüssel zu strukturieren kann man ganze Gruppen von Schlüsseln bilden, welche einer bestimmten Transaktionssache zugeordnet werden kann. Z.B. kann ein Verkäufer einen Zweig des Schlüsselbaumes einem Produkt zuordnen, womit die Organisation der Transaktionen um einiges erleichtert wird.
Ein weiterer und noch wichtigerer Vorteil ist die Fähigkeit öffentliche Schlüssel und daraus Adressen zu erstellen, ohne dabei Zugang zu den korrespondierenden privaten Schlüssel zu haben. Dies ist möglich, da die Ableitungsfunktion um die Kinder-Schlüssel zu erstellen (Child Key Derivation Function) aus dem öffentlichen Eltern-Schlüssel die Kinder-Schlüssel erstellt. Aufgrund dieser Eigenschaft ist es möglich HD-Wallets auf unsicheren Systemen zu verwalten. Das Wallet wird hier in einen Empfangs-Modus gestellt, in dem es für jede Transaktion einen öffentlichen Schlüssel und eine Adresse erstellt, ohne die Fähigkeit besitzen Transaktionen selber zu tätigen. [10]

4 Schluss

In diesem Paper wurden die grundlegenden Themen der Blockchain Technologie behandelt und ein Einstieg in die Implementation von Bitcoin gegeben.
Die in diesem Paper behandelten Probleme von dezentralen Netzwerken wurden nach und nach am praktischen Beispiel Bitcoins erläutert und aufgelöst. Die kryptografischen Grundlagen sind hier unerlässlich um Vertrauen, Integrität und Nichtabstreitbarkeit zu gewährleisten. Dies sollte klar machen, das die hier behandelte Technologie durchaus praktisch implementierbar ist und definitiv die bisher etablierten Branchen (vorallem die Finanzbranche) stark erschüttern wird und es schon tut.

Literaturverzeichnis

[1] Coinbase, letzter Zugriff: 18. Januar 2021. https://coinmarketcap.com

[2] Satoshi Nakamoto: Bitcoin: A Peer-to-Peer Electronic Cash System, Whitepaper, 2008

[3] Tatiana Gayvoronskaya & Christoph Meinel: Blockchain: Hype or Innovation, Springer Verlag, 2021

[4] Hans-Georg Fill & Andreas Meier: Blockchain: Grundlagen, Anwendungsszenarien und Nutzungspotenziale, Springer Verlag, 2020

[5] Wikipedia: Hash-Baum, zuletzt bearbeitet: 19. Januar 2021. https://de.wikipedia.org/wiki/Hash-Baum

[6] Wikipedia: Digitale Signatur, zuletzt bearbeitet: 18. September 2020. https://de.wikipedia.org/wiki/Digitale_Signatur

[7] Fraunhofer FIT: Blockchain und Smart Contracts - Technologien, Forschungsfragen und Anwendungen, 2017

[8] Cambridge: Center for alternative Finance, letzter Zugriff: 10. Februar 2021. https://www.cbeci.org/cbeci/comparisons

[9] BitFury Group: Proof of Stake versus Proof of Work, Whitepaper, 2015

[10] Andreas M. Antonopaulus: Mastering Bitcoin, O'Reilly Media, Inc., 2014

[11] The Morpheus Tutorials: Bitcoin, die Blockchain und Kryptowährungen einfach erklärt und praktisch angewandt, YouTube. https://youtube.com/playlist?list=PLNmsVeXQZj7oiW2UVh1Go8JgllRbZRcPK