HSB
Hochschule Bremen
City University of Applied Sciences

Bachelor-Thesis

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

Untersuchung der Distributed Ledger Technologien auf Zukunftstauglichkeit

Fakultät 4 - Elektrotechnik und Informatik

Janusz Spatz and Jens Altrock

Lorem Ipsum

Abstract

äöüß Lorem ipsum dolor sit amit, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua.

```
const express = require('express');
const address_route_getAddress =
    require('./api/routes/getAddress.js');
const address_route_user = require('./api/routes/user.js');
const address_route_fueling = require('./api/routes/fueling.js');
const bodyParser = require('body-parser');
const api = express();

api.use(bodyParser.urlencoded({extended: true}));
api.use(bodyParser.json());

api.use('/getAddress', address_route_getAddress);
api.use('/fueling', address_route_fueling);
api.use('/user', address_route_user);

module.exports = api;
```

At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum
dolor sit amet, consetetur sadipscing siehe Abbildung 1 auf seite ii elitr, sed

Abbildung 1: Das Logo der Hochschule Bremen

diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua.

At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Distributed Ledger Technologie	Skalierbarkeit	Programmierbarkeit	Funktion	Sicherheit
Bitcoin	nicht sehr gut	viele apis	durchdacht	sicher, jedoch nicht sehr anonym
IOTA	je mehr desto besser	gut, aber ich muss testen was passsiereren wuerde wenn hier viel text stehen tut	funktioniert anders als blockchain	sicher aber fehleranfaellig

Tabelle 1: Vergleich der Distributed Ledger Technologien

Danksagung

Lorem ipsum dolor [bal08] sit amet, consetetur sadipscing [DBT10] elitr, sed diam nonumy [Nak08] eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Inhaltsverzeichnis

1 Einleitung **1**

1.1 Geschichte 1
1.2 1979 Der Hash Tree (Merkle Tree) 2
1.3 1997 Proof-of-Work (Hashcash) 2
1.4 2008 Bitcoin Whitepaper - Satoshi Nakamoto 2
1.5 Kooperationen mit Blockchain/Tangle 3
1.6 Ziel der Thesis 4

2 Grundlagen **5**

2.1 Raspberry Pi 5
2.2 GPIO 6
2.3 Steckbrett (Breadboard) 6
2.4 Internet of Things (IoT) 7
2.5 Industrie 4.0 7
2.6 Tangle 8
2.7 Funktionsweise 8
2.8 Distributed Ledger 9

I Theoretischer Teil **11**

3 Einführung **13**

4 Funktion **15**

4.1 Bitcoin 15

4.1.1 Allgemeine Funktion 15
4.1.2 Transaktionen 16
4.1.3 Privater & öffentlicher Schlüssel 17
4.1.4 Signatur 17
4.1.5 Mining 17
4.2 IOTA 19
4.2.1 IOTA Seed 19
4.2.2 Tangle 20
4.2.3 Directed Acyclic Graph 21
4.2.4 Transaktion 21
4.2.5 Zusammenfassung 24
4.2.6 Proof-of-Work 24
4.2.7 Bundle 25
4.2.8 Signatur 26

5 Sicherheit **29**
5.1 Bitcoin 29
5.1.1 Wallet Schutz 29
5.1.2 Anonymität 29
5.1.3 Validität einer Transaktion 30
5.1.4 Angriffsszenarios 30
5.1.5 Zukunftssicherheit 30
5.2 IOTA 30
5.3 Gegenüberstellung 30
5.4 Ergebnis 30

6 Programmierbarkeit **31**

7 Nutzen **33**
7.1 IOTA 33
7.1.1 MAM 33
7.2 Bitcoin 35

8 Beurteilung **37**

II Praktischer Teil **39**

9 Tanken in der Zukunft **41**
9.1 Projektbeschreibung 41
9.2 Projektbegründung 41
9.3 IOTA als Währung 42

10 Fahrzeug - Client **43**
10.1 Ziel und Anforderungen 43
10.2 Aufbau 43
10.2.1 Vorbedingungen 44
10.2.2 Einrichtung 45
10.2.3 Ergebnis der Vorbereitung 48
10.3 Umsetzung 48
10.3.1 GPIO-Pins 48
10.3.2 IOTA Einrichtung 48
10.3.3 IOTA Transaktionen 48
10.4 Ergebnis 49

11 Tankstelle - Server **51**
11.1 Projektziel 51
11.2 Projektaufbau 52
11.2.1 Vorbedingungen 52
11.2.2 Projektabgrenzung 54
11.3 Umsetzung 55
11.3.1 Tank-Vorgang 55
11.4 Ergebnis 57

III Schlussteil **59**

12 Evaluierung **61**

13 Fazit **63**

Abbildungsverzeichnis

1 Das Logo der Hochschule Bremen ii

Tabellenverzeichnis

1 Vergleich der Distributed Ledger Technologien ii

Kapitel 1

Einleitung

Seit 2017 besteht ein starker Wachstum des Interesses an Blockchain, wobei Bitcoin als prominenteste Implementation angesehen wird. Am Sonntag, den 17. Dezember 2017 durchbricht Bitcoin erstmals fast die $20.000 Marke ($19,783.21) [Hig17]. Doch nicht nur Bitcoin hat von dem starken Interesse profitiert, sondern auch viele weitere Kryptowährungen sowie andere Blockchain-Implementierungen. Damit war die Technologie in das Bewusstsein der breiten Masse gerutscht und zugleich häufen sich Fragen im Bezug auf Zukunftssicherheit und Tauglichkeit der jeweiligen Technologien. Jene Fragen sollen in dieser Bachelorarbeit erörtert werden.

1.1 Geschichte

Die Geschichte der Blockchain ist eng mit der Geschichte von Bitcoin verknüpft. Bitcoin nutzt verschiedene Techniken um seine verschiedenen Funktionen zu realisieren. Das besondere an diesen Funktionen sind folgende Herausstellungsmerkmale: Dezentralität, Transparenz und Sicherheit. Um dies zu gewährleisten wurde auf bereits bekannte Techniken zurückgegriffen die in der Vergangenheit realisiert wurden.

1.2 1979 Der Hash Tree (Merkle Tree)

Im Jahr 1979 patentierte Ralph Merkle das Prinzip des Hash Trees. Dieses ist später auch bekannt als "Merkle Tree".[5] Der Nutzen des Hash Trees wird bei der effizienten Verifizierung von Daten deutlich. Anfänglich werden Dateien, in der Regel Paare aus zwei Dateien, miteinander gehasht. Der daraus resultierende Kind-Hash wird mit dem Kind-Hash anderer Dateien zu einem gemeinsam Hash verrechnet. So entsteht nach und nach ein Baum aus diversen Hashes die schließlich einen Top-Hash ergeben. Mithilfe dieses Top-Hashes ist es möglich die Unversehrtheit jeder anfänglichen Datei zu gewährleisten.

1.3 1997 Proof-of-Work (Hashcash)

Das Proof-of-Work Konzept wurde anfänglich gegen denial-of-service Attacken oder für die Abwehr von Spam beim Email Verkehr gedacht. Der Erfinder Adam Back nannte dieses System auch "Hashcash".[6] Damit eine Person, die dieses Konzept nutzt, beispielsweise eine Email versenden kann, muss sie vorher einen kleinen Aufwand betreiben. Dies könnte eine einfache mathematische Formel sein. Bei erfolgreicher Berechnung ist die Email freigegeben. Da eine Vielzahl dieser kleinen Aufwände sich zu einem großem Rechenaufwand aufstaut und da die Komplexität der Aufwände mit jeder erfolgreichen Abarbeitung steigt, wird in der Schlussfolgerung das schnelle Versenden vieler aufeinander folgender Emails verhindert.

1.4 2008 Bitcoin Whitepaper - Satoshi Nakamoto

Im Oktober 2008 veröffentlichte eine Person oder Gruppe unter dem Namen Satoshi Nakamoto ein Whitepaper auf dem das heute funktionierende Bitcoin Protokoll beschrieben wird.[7] Dies geschah ca. einen Monat nachdem die Lehman Brothers insolvent gingen und eine der größten Finanzkrisen der

Geschichte stattfand.[8] Bitcoin versprach eine digitale Währung zu sein, welche mit Hilfe kryptographischer Techniken ein "Peer To Peer" Bezahlsystem ermöglicht. Dabei verwendete es das in früheren Jahren entwickelte Proof-Of-Work Konzept welches auch ermöglicht Einheiten herzustellen.[7] Gleich zu Beginn wurde die maximale Anzahl an je verfügbaren Einheiten auf 21 Mio eingeschränkt. Nur wenige Monate später wurde ein open Source Programm veröffentlicht, welches das Bitcoin Protokoll implementierte und für jedermann frei verfügbar machte.[9]

1.5 Kooperationen mit Blockchain/Tangle

Immer mehr kommerzielle Firmen entdecken das Potenzial von Kryptowährungen und weiteren Blockchain-technologien. Die Kooperation zwischen dem Automobilhersteller VW und diversen Kryptowährungen ist ein aktuelles Beispiel in dem eine Firma potential in dieser Technologie sieht. In einer am 8. August, 2018 veröffentlichten Kurznachricht auf der Social-Media-Plattform Twitter schreibt die Volkswagen Group:

> „Bringing #blockchain systems to the road: We're working full steam ahead on making super-safe #cryptosystems available to our customers. For filling the tank, unlocking your car – and all kinds of other possibilities: #bitcoin #ethereum #iota"
>
> - Volkswagen Group auf Twitter[1]

In einem Blockeintrag der Volkswagen Group geht hervor, dass sie anhand von Blockchains, die Zuverlässigkeit und Sicherheit von Autos erhöhen wollen. Zum Beispiel indem Kilometerzähler vor Manipulation geschützt sind, und Hackerangriffe auf selbstfahrende Autos vorzubeugen.[2]

Auch weitere Firmen wie Bosch geben in einer Pressemitteilung ende 2017 bekannt, dass auch sie in IOTA investieren und eng mit ihnen an verschiedenen Fronten zusammenarbeiten werden. [3] Sowie auch Microsoft hat sich die Kryptowährungen zu Nutzen gemacht, die Nutzer können anhand von

[1] https://twitter.com/VWGroup/status/1027205629436407810

Bitcoin käufe im Microsoft Store tätigen [4].

Es ist davon auszugehen, dass viele weitere namhafte Firmen dem Beispiel folgen werden und auch Blockchain/Tangle Technologien in ihren jeweiligen Anwendungsbereichen einbinden werden. So kristallisiert sich die Möglichkeit heraus, dass Blockchains oder Tangles im zukünftigen Alltag eine erhebliche Rolle spielen werden, was zum Beispiel Transaktionen und Sicherheit betrifft.

Aber die Reichweite von Kryptowährungen ist nicht nur auf Unternehmen limitiert, sondern erstreckt sich international, sodass diverse Nationen sich umfassend mit der neuen Art von Währung beschäftigen.

1.6 Ziel der Thesis

Grundlegend soll die Frage geklärt werden, wie sich ein, auf Blockchain basierendes, Konzept für die Zukunft durchsetzen kann und mit welchen realistischen Aussichten zu rechnen ist. Hierbei ist es notwendig, vorhandene Implementierungen nach Einsatzmöglichkeit und Art der Technologie zu kategorisieren und zu beschreiben.

Um einen genauen Ausblick für die Zukunft geben zu können, muss die Entwicklung, Sicherheit und Usability von den zuvor kategorisierten Technologien ausführlich und kritisch untersucht werden. Unterstützt wird das Ergebnis der Thesis von einem selbst erstellten Programm, welches mit einer Kryptowährung arbeitet. Dieses Programm soll eine mögliche zukünftige Nutzung von Blockchain aufzeigen. Sie soll eine Dienstleistung annehmen können, und anschließend autonom den Dienst mittels einer Kryptowährung bezahlen können. Dies ist ein Anwendungsfall, welcher im Rahmen der zuvor erwähnten Kooperation zwischen VW und IOTA auch ausgearbeitet wird.

Kapitel 2

Grundlagen

2.1 Raspberry Pi

Bei einem Raspberry Pi handelt es sich um einen Einplatinencomputer im Kreditkarten-Format. Das Gerät besitzt unter anderem USB-, Ethernet- und AUX-Eingänge, sowie einen Steckplatz für eine Micro-SD-Karte, die als Speicherort des Betriebssystems und der Daten genutzt wird. Bildinformationen werden an den HDMI-Ausgang gesendet. Als Stromquelle dient der Micro-B-Eingang der mit einer Eingangsspannung von 5 Volt betrieben werden kann. [11] Ursprünglich diente die Entwicklung des Raspberry Pi's dafür, heranwachsende Leute in Großbritannien dem Programmieren nahezubringen. [12] Inzwischen ist der Raspberry Pi in verschiedenen Modellen mehr als 17 Millionen mal (Stand: ende 2017) verkauft worden. [13] Damit geht der Raspberry Pi weit über den anfänglichen Zielen hinaus und wird auch von erfahrenen Programmierern für vielfältige Projekte genutzt. Der Einplatinencomputer wird mit einem auf Linux basierendem Betriebssystem genutzt. Aufgrund der Architektur des CPUs sind Betriebssysteme wie Windows nicht auf dem Raspberry Pi verfügbar. Das offizielle Betriebssystem ist das auf Debian (Linux) basierende Raspbian. [17] Es beinhaltet alle Programme die nötig sind um mit dem Programmieren des Raspberry Pis zu beginnen. Durch die Faktoren von einer umfangreichen und integrierten Programmieroberfläche, der

simplen Installation und die geringen Kosten machen den Raspberry Pi zu einer idealen Umgebung um prototypische Anwendungen umzusetzen.

2.2 GPIO

Zahlreiche GPIO-Pins befinden sich auf dem Raspberry Pi um eine direkte Kommunikation mit externen Schaltkreisen zu erlauben. GPIO ist eine Abkürzung für General Purpose Input Output, übersetzt bedeutet dies Ein- und Ausgabe für allgemeine Einsatzmöglichkeiten. Dafür stehen auf dem Raspberry Pi bis zu 40 Pins zur Verfügung. Die einzelnen Pins können mittels Software und Hardware an- oder ausgeschaltet werden. Darüber hinaus kann ein Wert auch eingelesen oder gesetzt werden. Doch nicht jeder Pin steht für GPIO-Zwecke zur Verfügung. Einige Pins sind für andere Funktionen vorbehalten (z.B. Datenübertragung, Strompotenzial von +5V, +3.3V und 0V). Aus diesem Grund empfiehlt es sich, das Datenblatt über die Verteilung der Pins anzugucken. Die GPIO-Pins ermöglichen es, angeschlossene Schaltkreise zu steuern oder einzulesen. Sie sind nach belieben programmierbar. Daher finden Raspberry Pis oft Verwendung in Projekten wie Heimautomatisierung, Drohnen und Sicherheitssystemen. [14][15][16]

2.3 Steckbrett (Breadboard)

Ein Steckbrett wird genutzt um lötfrei Komponenten mit den Pins des Raspberry Pis zu verbinden. Mittels Kabeln werden die Pins mit einer Reihe des Bretts verbunden, wodurch an der Reihe eine Spannung angelegt wird. Nun kann an dieser Reihe eine Komponente angesteckt werden. Damit der Schaltkreis geschlossen wird, muss das andere Ende mit einem Pin verbunden werden, der geerdet ist (ein GRND-Pin). Da es damit möglich ist, schnell neue Komponenten anzuschließen und den Schaltkreis nach belieben zu verändern ist es gut geeignet um prototypisch Schaltkreise zu erstellen.

2.4 Internet of Things (IoT)

Das Internet of Things (deutsch: Internet der Dinge) beschreibt die Verknüpfung von Geräten und Sensoren die vorher eigenständig und isoliert waren mit dem Internet. Damit ist es ihnen möglich Daten untereinander auszutauschen indem sie ihre eigenen Daten übertragen und externe Daten anfordern. So lassen sich die Geräte über das Internet gleichzeitig von Menschen und auch anderen Maschinen fernsteuern. Mit diesem Begriff wird die Idee eingeleitet, dass die Nutzer des Internets nicht ausschließlich Menschen sind, sondern zunehmend auch „Dinge“ (Things) über diesem Kommunizieren. Im privaten Gebrauch werden als Beispiele alltägliche Geräte aufgeführt, die durch die Vernetzung im Internet mit anderen Geräten das Leben des Nutzers komfortabler machen oder auf sonstige Weise positiv beeinflussen. Ein Beispiel wäre die Hausautomation mit Sicherheitskameras, die, wenn etwas verdächtiges ermittelt wird, sofort den Hauseigentümer über das Internet warnen und das Videomaterial live auf das Smartphone übertragen wird. [18] Das Internet der Dinge lässt sich auch im industriellen Bereich anwenden. So lassen sich Herstellungsprozesse durch die Vernetzung von Sensoren, Anlagen und Maschinen so automatisieren, dass der Prozess effizienter wird und menschliche Hilfe immer obsoleter wird. [18]

2.5 Industrie 4.0

In diesem Zusammenhang fällt oft der Begriff Industrie 4.0. [19] Damit ist eine vierte industrielle Revolution gemeint. Kurz beschrieben geht die erste industrielle Revolution mit der Mechanisierung einher, die zweite mit der Massenfertigung von Produkten mittels Fließbändern und Fabriken, und die dritte mit der Automatisierung von Arbeitsschritten mittels Maschinen. [20] Insofern handelt es sich bei Industrie 4.0 um die Vernetzung der automatisierten Arbeitsschritte, sodass sie autonom miteinander kommunizieren können und sich dem nächsten Arbeitsschritt anpassen können.

2.6 Tangle

Bei Tangle handelt es sich um ein Distributed Ledger Software Protokoll, welches sich grundlegend von Blockchain unterscheidet. Die Tangle-Technologie wird von den Entwicklern als nächste evolutionäre Weiterentwicklung der Blockchain beschrieben. [21] Die Technologie nimmt sich den Schwächen der Blockchain an und integriert Lösungen für diese. Konzeptionell bedeutet dies, dass die heterogene Unterscheidung von Minern und Usern in Blockchains homogenisiert werden soll, indem Transaktionen von Nutzern verifiziert werden sollen, die selber eine Transaktion veranlassen möchten. So löst sich das Problem der Skalierung und zugleich fallen die Transaktionsgebühren weg. Letzteres ist möglich, da es zwingend notwendig ist, andere Transaktionen zu verifizieren, falls eine eigene Transaktion aufgegeben werden soll. So „zahlt“ der Nutzer die Gebühren mit Rechenpower. Tangle wurde von der IOTA-Foundation entwickelt und kommt in der gleichnamigen Kryptowährung IOTA zum Einsatz.

2.7 Funktionsweise

Anders als bei Blockchain ist es nicht möglich, neue Tokens zu „minen“ (bedeutet: herzustellen). Es besteht seit der Entwicklung ein fester Betrag an verfügbaren Tokens von genau 2.779.530.283 MIOTA (1 MIOTA = 1.000.000 IOTA). [22] Anstatt einer globalen Blockchain, existiert bei IOTA nur ein sogenannter Tangle. Dieser richtet sich nach einem Mathematischen Konzept, dem Directed Acylic Graph. Mit diesem Konzept speichert IOTA die Transaktionen. Eine Transaktion muss von einem Node aufgegeben werden. Dabei handelt es sich um ein Gerät, welches, um eine Transaktion auszuführen zwei Transaktionen verifiziert anhand des Directed-Acylic-Graph-Algorithmus‘. Sobald eine Transaktion nicht verifiziert werden kann kann die ursprüngliche Transaktion nicht ausgehen. [21]

2.8 Distributed Ledger

Lorem ipsum

Teil I

Theoretischer Teil

Kapitel 3

Einführung

Kapitel 4

Funktion

4.1 Bitcoin

Bitcoin ist ein Distributed Ledger Protokoll das erstmalig Geldtransfer über eine dezentrale, anonyme Übereinstimmung der Richtigkeit verfügt. Diesen Konsens erreicht es über die implementierte Blockchain, die wie ein global einsichtiges Kassenbuch funktioniert. Die Blockchain kann auf verschiedene Art und Weise genutzt und entwickelt werden. Sie ermöglicht es über das Internet Werte auszutauschen ohne von einem Mittelsmann abhängig zu sein.[23] Durch kryptographische Techniken sind Daten die in die Blockchain gelangen nur durch sehr großen Aufwand manipulierbar, was sie für das Speichern von sensiblen Daten attraktiv macht. [10] Im Folgenden wird die grundlegende Funktion der Blockchain sowie dessen Einsatz bei Bitcoin sachlich untersucht.

4.1.1 Allgemeine Funktion

Für die gegenwärtige Funktion von Bitcoin und dessen Blockchain sind zwei elementare Techniken notwendig. Zum Einen ist das die Public-Key-Kryptographie zum anderen die kryptographischen Hashfunktionen.

Bei der Public-Key-Kryptographie resp. digitalen Signatur erstellt der Sender

einer Nachricht ein Schlüsselpaar bestehend aus einem privaten sowie einem öffentlichen Schlüssel. Die beiden Schlüssel haben zwei ergänzende Funktionen. Der Autor der Nachricht verwendet den privaten Schlüssel um diesen mit seinen Informationen zu verbinden und dadurch zu signieren. Die signierten Daten werden gemeinsam mit dem öffentlichen Schlüssel an den Empfänger weitergegeben. Dank des öffentlichen Schlüssel ist es dem Empfänger möglich die Daten zu authentifizieren. Des Weiteren ist durch die Verknüpfung der Daten mit dem privaten Schlüssel die inhaltliche Integrität vor Manipulation geschützt.

Durch kryptographischen Hashfunktionen entstehen aus Zeichenketten mit variabler Länge Zeichenketten fester Länge. Diese Zeichenketten sind "deterministisch". Die gleichen Eingangsdaten werden nach der Hash-Berechnung immer den gleichen Hash verursachen. Veränderte Eingangsdaten führen zu einem stark abweichenden Hash.

Außerdem gibt es drei weitere nennenswerte Eigenschaften von Hashfunktionen. Zum Einen ist es nahezu unmöglich durch den Hash die anfänglichen Daten wiederherzustellen. Des Weiteren ist es nahezu unmöglich mit abweichenden Eingangsdaten denselben Hash zu generieren. Zuletzt ist es nahezu unmöglich zwei verschiedene Eingangsdaten zu finden aus denen sich derselbe Hashwert ergibt.

4.1.2 Transaktionen

Um eine Transaktion im Bitcoin Netzwerk zu veranschaulichen wird ein fiktives Beispiel herangezogen. Alice möchte an Bob zwei Bitcoins (BTC) versenden. Es existieren jedoch keine Konten oder Kontostände, sondern ausschließlich die öffentliche Liste aller jemals getätigten Transaktionen, also die Bitcoin - Blockchain selbst, sowie die Hashwerte der einzelnen öffentlichen Schlüssel denen die existierenden Transaktionen zugeordnet werden. Aufgrund der vergangenen Transaktionen kann der Wertbestand an BTCs des zugehörigen privaten Schlüssels ermittelt werden. Mithilfe einer digitalen Software (Wallet) kann dieser Wertbestand des privaten Schlüssels ein-

gesehen und neue Transaktionen eingereicht werden. Um eine Transaktion zu veranlassen werden mithilfe des privaten Schlüssels die zurückliegenden eingegangenen Transaktionen signiert und zusammen mit der ausgehenden Transaktion an das Bitcoin Netzwerk versandt. Dies entspricht in dem anfänglichen Beispiel einer Auflistung aller Transaktionen in denen Alice involviert war, sowie Alice' Intention zwei BTCs an den öffentlichen Schlüssel von Bob zu senden.

Diese Nachricht gelangt an einen Netzknoten der über verschiedene Ressourcen verfügt. Neben einer aktuellen und vollständigen Kopie der Blockchain, einen Cachespeicher (UTXO), der Transaktionswerte der Blockchain enthält die noch nicht für neue Transaktionen verwendet wurden, existiert noch eine Datenbank mit unbestätigten Transaktionen. In diesem Netzknoten wird die Legitimität von Alice' Transaktion überprüft, indem zum einen ihre Liquidität aufgrund ihrer bisherigen Transaktionen, sowie gleichzeitig ihre Signatur anhand des öffentlichen Schlüssels, bestätigt wird. Mithilfe der UTXO wird außerdem geprüft, ob die genutzten BTCs nicht bereits anderweitig reserviert wurden. Wenn dies fehlerfrei geschieht, wird die unbestätigte Transaktion in die dafür eingerichtete Datenbank aufgenommen. Der Netzknoten meldet diese eingereichte Transaktion an möglichst viele weitere Netzknoten weiter, die wiederum die Daten nach dem erklärten Schema überprüfen und anschließend in die Datenbank der unbestätigten Transaktionen aufnehmen.

4.1.3 Privater & öffentlicher Schlüssel

4.1.4 Signatur

Das Signieren in Bitcoin funktioniert mit dem sogenannten „Elliptical Curve Digital Signature Algorithm“. Eine elliptische Kurve bietet in der Mathematik einige besondere Eigenschaften.

4.1.5 Mining

Das Bitcoin-Netzwerk verfügt über zwei Arten von Netzwerkknoten. Einer ist ausschließlich für das zuvor genannte Verifizieren und Speichern der einge-

henden Transaktionen sowie dessen Weiterleitung an weitere Netzknoten zuständig. Der zweite ist darüber hinaus befähigt neue Blöcke in der Blockchain zu speichern. Dieser sogenannte "Mining-Netzknoten" speichert zunächst unbestätigte Transaktionen in einem Block zusammen. Dieser Block besitzt zusätzlich einen Blockheader, der für die kommende Abspeicherung in der Blockchain essentiell ist.

Es ergeben sich ab diesem Zeitpunkt einige Problematiken. Aufgrund von Verzögerungen im Netzwerk oder möglichen Ausfällen sind einige Netzknoten aktueller. Dies führt dazu, dass verschiedene Transaktionen, welche sich voneinander unterscheiden, in den einzelnen Netzknoten existieren. Außerdem können bösartige Absender gefälschte Transaktionen weiterleiten, welche dieselben Wert-Ressourcen aufweisen. Bitcoin verwendet für diese Problematiken die Proof-Of-Work (PoW) Technologie. In der Regel wird diese Technik für das Verhindern von missbräuchlicher Benutzung von Diensten eingesetzt. Damit ein Dienst genutzt werden kann muss bei dieser Technik ein gewisser Aufwand erbracht werden. Im Falle der Bitcoin-Blockchain muss gemäß dieses Schemas eine rechenintensive Aufgabe gelöst werden. Der Block-Header ergibt einen gewissen Hashwert. Dieser muss solange manipuliert werden bis das Ergebnis unterhalb eines gewissen Zielwertes liegt. Diese partielle Hashinversion beruht auf dem Hashcash-Prinzip von Adam Back (s. Geschichte). Der Hash entspringt dem Block-Header und wird aus einer Referenz zum vorherigen Block, einem Zeitstempel, dem Zielwert des Hash-Rätsels, dem Top Hash eines Merkle-Trees, welcher alle Transaktionen durch aufeinander aufbauende Hashes zusammenfasst, sowie einer variablen Zeichenfolge (Nonce) errechnet. Die Nonce wird so oft geändert bis der Hashwert unterhalb des Zielwertes liegt. In anderen Worten muss dieser mit einer gewissen Anzahl von Nullen beginnen. Der erste Mining-Netzknoten der dieses Rätsel löst versendet seinen Block an das Netzwerk. Dort wird der Block ebenfalls auf Validität geprüft und bei Erfolg in die eigene Blockchain integriert.

In 1,69% der Fälle, finden zwei Netzknoten zu nahezu der gleichen Zeit eine Lösung des Hash-Rätsels und versenden ihre jeweiligen Blöcke. In diesem Fall erhalten unabhängige Netzknoten verschieden Versionen der Blockchain.

Infolgedessen teilen sich die Netzknoten auf. Dieses Phänomen ist als Gabelung bekannt. Die jeweiligen Netzknoten arbeiten weiter auf Grundlage ihrer bekannten Blockchain bis zu dem Moment bei dem sie über eine längere Blockchain Version informiert werden. Tritt dieser Fall ein, wird die bekannte Blockchain durch die längere ausgetauscht. Sollte die längere Version einige Transaktionen der zuvor bekannten Blockchain nicht enthalten, werden diese wieder in die Datenbank der nicht bestätigten Transaktionen aufgenommen. Momentan werden ca. alle zehn Minuten ein neuer Block in die Blockchain aufgenommen. Damit sich diese Zeitspanne weiter in einem angemessenen Rahmen befindet, wird stetig die Komplexität des Hash-Rätsels angepasst. Für das Finden der Lösung eines Hash-Rätsels und das erfolgreiche Integrieren eines neuen Blockes in der Blockchain, erhält der Mining-Netzknoten eine gewisse Menge an BTCs, wodurch neue BTCs erzeugt werden.[24]

4.2 IOTA

IOTA ist ein Distributed Ledger Protokoll, welches eine grenzenlose Skalierbarkeit des eigenen Netzwerkes zulässt. Dies wird durch die Eigenschaft ermöglicht, das der Benutzer und der Miner nicht länger voneinander getrennt sind. In IOTA wird das Prinzip der durch Bitcoin bekannten Blockchain umgedacht. Transaktionen werden nicht länger in einem Block gespeichert, sondern über ein Netz aus Transaktionen verteilt. In diesem Netz bestätigen sich die jeweiligen Transaktionen gegenseitig ihre Richtigkeit. Dieses Netz wird Tangle genannt und kann sich wie ein „Directed Acyclic Graph“ vorgestellt werden. Die folgende Untersuchung liefert einen sachlichen Einblick in die Funktion IOTAs und der implementierten Tangle.

4.2.1 IOTA Seed

Der IOTA Seed ist essentiell wichtig für die Funktion von IOTA. Er fungiert als „Kontonummer“ und verweist auf die Menge an IOTAs die dem Konto zugeordnet werden (das IOTA-Wallet). Wenn der Besitzer des Seeds eine Transaktion durch die IOTA-Tangle veranlassen möchte, wird mit Hilfe von

kryptographischen Hashfunktionen eine Adresse aus dem Seed erstellt [s. Sicherheit]. Diese Adresse kann nun genutzt werden um IOTAs zu empfangen oder, sollte sie bereits einen Wert besitzen, zu versenden. Adressen können nur durch den Seed generiert werden und es ist (technisch) unmöglich von der Adresse den Seed wiederherzustellen. [31] Ein Seed besteht aus 81 Trytes die dank des von IOTA zur Verfügung gestellten „Tryte-Alphabet“ durch die 26 Großbuchstaben A-Z des Alphabets und der Zahl 9 dargestellt werden.

4.2.2 Tangle

Bei Tangle handelt es sich um ein Distributed Ledger Software Protokoll, welches sich grundlegend von Blockchain unterscheidet. Die Tangle-Technologie wird von den Entwicklern als nächste evolutionäre Weiterentwicklung der Blockchain beschrieben. [21] Die Technologie nimmt sich den Schwächen der Blockchain an und integriert Lösungen für diese. Konzeptionell bedeutet dies, dass die heterogene Unterscheidung von Minern und Usern in Blockchains homogenisiert werden soll, indem Transaktionen von Nutzern verifiziert werden sollen, die selber eine Transaktion veranlassen möchten. So löst sich das Problem der Skalierung und zugleich fallen die Transaktionsgebühren weg. Letzteres ist möglich, da es zwingend notwendig ist, andere Transaktionen zu verifizieren, falls eine eigene Transaktion aufgegeben werden soll. So „zahlt“ der Nutzer die Gebühren mit Rechenleistung.

Tangle wurde von der IOTA-Foundation entwickelt und kommt in der gleichnamigen Kryptowährung IOTA zum Einsatz.

Anders als bei Blockchain ist es nicht möglich, neue Tokens zu „minen“ (bedeutet: herzustellen). Es besteht seit der Entwicklung ein fester Betrag an verfügbaren Tokens von genau 2.779.530.283 MIOTA (1 MIOTA = 1.000.000 IOTA). [22] Die Tangle richtet sich nach einem Mathematischen Konzept, dem Directed Acyclic Graph. Mit diesem Konzept speichert IOTA die Transaktionen. Eine Transaktion muss von einem Node aufgegeben werden. Sobald eine Transaktion nicht verifiziert werden kann, kann die ursprüngliche Transaktion nicht ausgehen. [21]

4.2.3 Directed Acyclic Graph

Die IOTA-Tangle ist aufgebaut in Form eines „Directed Acyclic Graph“ (DAG) was in diesem Kontext bedeutet, dass sie sich in eine Richtung bewegt und niemals kreisförmig ist.[32] Damit eine neue nicht verifizierte Transaktion in die Tangle aufgenommen werden kann, ist neben dem Proof-of-Work eine Verifikation von zwei ebenfalls nicht verifizierten Transaktionen notwendig. Diese beiden Transaktionen werden in dem Transaktions-Bundle abgespeichert und sind dementsprechend referenziert. Da nur nicht verifizierte Transaktionen in diesem Prinzip verwendet werden, entspricht der entstehende Graph eines DAG.

4.2.4 Transaktion

Eine Transaktion in IOTA besteht aus mehreren Hashes und Werten, die jeweils untereinander eine verschiedene Aufgabe erfüllen. In den folgenden Abschnitten werden diese detailliert aufgeführt.

signatureMessageFragment Sollte die Transaktion eine ausgehende Zahlung sein, wird dieses Feld benötigt um die Signatur des privaten Schlüssels zu enthalten. Die Länge der Signatur ist davon abhängig, mit welcher Sicherheitsstufe diese Transaktion gehasht wird. Sollte die Sicherheitsstufe 2 oder 3 sein, wird eine weiter wertlose Transaktion benötigt. Diese speichert in dem noch freien Signature Message Fragment den Rest der Signatur der vorangehenden ausgehenden Zahlung.

Sollte jedoch die Signatur des privaten Schlüssels nicht erforderlich sein, verbleibt dieses Feld leer und kann für das Übertragen einer Nachricht genutzt werden. Diesem Feld werden 2187 Trytes reserviert.

hash In diesem Feld wird der „transaction Hash“ nach dem Finden der „nonce“ und dem Proof-of-Work abgespeichert.

address Die Adresse der Transaktion wird erneut für verschiedene Aufgaben verwendet, die sich aus der Art der Transaktionen definieren. Sollte

eine Zahlung durchgeführt werden, wird in diesem Feld die Adresse des Empfängers angegeben. Handelt es sich jedoch um einen Geldeingang so ist hier eine Adresse aufgeführt die aus einem private Key des Besitzers Seed erstellt wurde. Der Adresse sind 81 Trytes reserviert.

value Der Wert einer Transaktion definiert dessen Art. Sollte dieser Wert positiv sein, handelt es sich um eine zahlende Transaktion (Output). In dem Feld „Address“ wird sich in diesem Fall die Adresse des Empfängers befinden und das Feld „Signature Message Fragment“ ist entweder leer oder beinhaltet eine benutzerspezifische Nachricht.

Sollte der Wert negativ sein handelt es sich bei der Transaktion um eine empfangene Transaktion (Input). Das Feld „Address“ enthält entsprechend eine Adresse die dank des Seeds und einem daraus entstandenen private Key generiert wurde. Eine eingehende Transaktion enthält einen negativen Wert, um den positiven Wert einer ausgehende Zahlung im Bundle zu rechtfertigen. Zwangsläufig wurde diese Adresse zuvor von einer fremden IOTA-Transaktion als Empfänger Adresse genutzt.

Um den Besitz des entsprechenden privaten Schlüssels zu beweisen, wird in dem „Signature Message Fragment“ die Signatur durch den privaten Schlüssel abgespeichert. Sollte die Sicherheitsstufe größer als 1 sein, werden weitere Transaktionen benötigt, um die gesamte Signatur zu speichern.

Trägt dieses Feld keinen Wert und entspricht 0, handelt es sich bei der Transaktion entweder um das Versenden einer einfachen Nachricht an eine Empfängeradresse oder wird genutzt um die Signatur einer vorangehenden Transaktion zu speichern.

Für dieses Feld wurden 27 Trytes reserviert.

obsoleteTag Der obsolete Tag enthält einen von Benutzer definierten Tag. Dieser könnte in zukünftigen IOTA - Iterationen entfernt werden. Die reservierte Länge dieses Feldes beträgt 27 Trytes.

timestamp Der Zeitpunkt ist nicht verpflichtend und kann ausgelassen werden. Ihm werden 9 Trytes reserviert.

currentIndex Dieses Feld zeigt die momentane Position im Bundle. Ihm werden ebenfalls 9 Trytes reserviert.

lastIndex Dieses Feld führt den Index der letzten Transaktion des umfassenden Bundles auf und liefert dementsprechend die Länge dessen. Reserviert werden erneut 9 Trytes.

bundle Dieses Feld enthält den Hash des umfassenden Bundles. Er wird genutzt um alle Transaktionen dieses Bundles zu gruppieren. Jede Transaktion in einem Bundle enthält in diesem Feld denselben entsprechenden Bundle Hash.

branch- & trunkTransaction Diese beiden Felder sind jeweils 81 Trytes lang und enthalten zwei zufällig gewählte, nicht verifizierte Transaktionen aus der Tangle. Nicht verifizierte Transaktionen im Tangle werden "Tips" genannt. Jede Transaktion ist verpflichtet, zwei zufällig gewählte Tips in der Tangle zu verifizieren, um selbst als Tip aufgenommen zu werden.

tag Dieser Tag wird vom Benutzer gewählt und kann frei vergeben werden. Er kann die Suche einer Transaktion im Tangle unterstützen. Seine Länge in der Transaktion beträgt 27 Trytes.

attachmentTimestamp Dieses neun Trytes große Feld beinhaltet den Zeitpunkt direkt nachdem der Proof-of-Work durchgeführt wurde.

attachmentTimestampLowerBound & attachmentTimestampUpperBound Diese beiden Felder zeigen ein Intervall auf in der die Aufnahme in die Tangle geschah. Sie sind jeweils 9 Trytes groß.

nonce Die nonce ist ein besonders wichtiger Teil einer Transaktion, denn sie wird benötigt um den Proof-of-Work durchzuführen. Die Länge dieses Feldes beträgt 27 Trytes.

4.2.5 Zusammenfassung

Rechnet man die Längen aller Felder zusammen, so erhält man die gesamte Länge von 2673 Trytes. Dies ist die von der IOTA-Foundation vorgegebene Länge die eine Transaktion haben soll. Die Art einer Transaktion bestimmt sich durch den Wert. Sollte eine Transaktion einen negativen Wert haben, so handelt es sich dabei um eine „Input“ Transaktion. Bei einem positiven Wert, wird Balance an eine „Output“ Adresse geschickt. Viele Felder tragen eine wichtige Aufgabe für die Aufnahme in der Tangle. So wird das Feld „signatureMessageFragment„ entweder für benutzerspezifische Nachrichten genutzt oder, falls benötigt, für die Signatur.[33]

4.2.6 Proof-of-Work

Um Transaktionen in die Tangle zu platzieren, sind keine Gebühren notwendig. Das ist aufgrund zwei essentieller Eigenschaften möglich. Zum Einen muss jede Transaktion zwei weitere nicht verifizierte Transaktionen bestätigen, des Weiteren wird mit Rechenleistung in Form eines Proof-of-Work die Transaktion beglaubigt. Im Detail läuft dieses Verfahren wie folgt ab: Nachdem alle Felder der Transaktion, bis auf die „nonce“ und den Hash für „bundle“, einen Wert erhalten haben, wird mit Hilfe eines Algorithmus die nonce gesucht. Diese ist 27 Trytes lang. Sobald die nonce gefunden wurde, wird sie für die letzten 27 Trytes der insgesamt 2673 Trytes der Transaktion verwendet. Diese Trytes werden durch den Curl Hash Algorithmus zu einem 81 Trytes langen Hash umgewandelt. Der Hash wird im Anschluss in Trits umgerechnet. Am Ende der resultierenden 243 Trits soll eine Folge aus Nullen stehen. Die länge der Folge wird von der IOTA-Foundation vorgegeben. So müssen die letzten Trits der Trytes, eines Transaktions Hashes, momentan mindestens 14 mal Null in Folge für die Aufnahme in der Tangle und Neun mal Null in Folge für die Aufnahme in dem Testnetzwerk betragen. Diese

Vorgabe wird „Minimum Weight Magnitude (MWM)“ genannt. Der Vorgang wird sooft wiederholt bis das vorgegebene MWM erreicht ist. Grundsätzlich kann die Aussage getroffen werden, dass mit einer steigenden MWM auch die Zeit, die für das Finden der passenden nonce benötigt wird, exponentiell ansteigt.

4.2.7 Bundle

IOTA ist nach einem Account - Schema aufgebaut, was genauer bedeutet, dass Adressen benötigt werden auf die Balance (der Fachausdruck des Geldwertes) eingegangen ist, um Balance weiter zu senden. Ein Bundle in IOTA umfasst in der Regel vier Transaktionen. Die erste Transaktion enthält die Adresse des Empfängers. Diese Transaktion wird mit einer positiven Balance versehen. In dem Bundle ist dies eine sogenannte „Output“ - Transaktion. Daraufhin folgen in der Regel, abhängig von der gewählten Sicherheitsstufe, zwei Transaktionen. Jede dieser Transaktionen basiert auf der gleichen Adresse, auf der zuvor Balance eingegangen ist. In anderen Worten muss diese Adresse zuvor Teil einer „Output“ - Transaktion gewesen sein. Es werden mehrere Transaktionen benötigt, da die Signatur des privaten Schlüssels mit versandt wird. Diese Signatur vergrößert sich, abhängig von der gewählten Sicherheitsstufe. Eine Transaktion, die sich einer Adresse bedient, die zuvor Balance empfing, nennt sich „Input“ - Transaktion. Sollte die Balance der Input Transaktion nicht ausreichen um den Betrag der Output Transaktion zu decken, müssen weitere „Input“ - Transaktionen angefügt werden, die jeweils durch ihren privaten Schlüssel signiert werden. Sollte die Balance der „Input“ - Adressen den Wert der 2Output“ - Transaktion überschreiten, wird der Restwert an eine 2Output“ - Adresse des Besitzers übertragen. Solange der summierte Wert der „Input“-Adressen nicht überschritten wird, können beliebig viele „Output“-Adressen einem Bundle an gehangen werden.

Da Bundles atomar sind, werden entweder alle oder keine Transaktion verifiziert.

4.2.8 Signatur

Bei IOTA werden ausgehende Zahlungen (Output) mit zuvor eingegangenen Zahlungen (Input) durchgeführt. Um den Besitz des notwendigen Inputs nachzuweisen, verwendet IOTA die „Winternitz One Time Signature“. Diese Art der Signatur veröffentlicht einen Teil des privaten Schlüssels und wird aus diesem Grund für den einmaligen Gebrauch von Signaturen verwendet. Um die Funktionsweise einer „One Time Signature“ (kurz OTS) zu verdeutlichen, wird die ähnliche Lamport OTS in einem fiktiven Beispiel herangezogen.

Ein privater Schlüssel besteht aus zwei gleichlangen Zahlenfolgen „k1“ & „k2“. Die Länge des jeweiligen Schlüssel Teils stimmt mit der Länge der zu signierenden Nachricht überein. In diesem Beispiel beträgt die Schlüssellänge 512 (2x 256). Die Nachricht kann eine beliebige Zeichenkette sein, die bspw. durch den SHA-256 Hash Algorithmus zu einem Hash „H“ mit der Länge von insgesamt 256 Bits umgewandelt wurde. Der Wert eines Bits kann ausschließlich Null oder Eins betragen. In einer Schleife wird jeder Wert des Hashes durchlaufen. Beträgt der Wert H(n) des Hashes an der Stelle n => H(n) = 0, so wird der Wert des ersten Schlüssel Teils an selbiger Stelle für die Signatur Sig(n) = k1(n) verwendet. Sollte jedoch der Wert H(n) des Hashes an der Stelle n => H(n) = 1 betragen, wird der Wert des zweiten Schlüssel Teils in der Signatur Sig(n) = k2(n) verwendet. Nach dem erfolgreichen Durchlaufen des Hashes „H“ wurde eine Signatur „Sig“ mit einer Länge von 256 angefertigt. Die Signatur enthält 50% der Werte des privaten Schlüssels.

Damit der Empfänger die Signatur überprüfen kann, benötigt er den öffentlichen Schlüssel. Dieser ist bspw. ein, durch den SHA-256 Hash Algorithmus angefertigter, Hash „pub“ des privaten Schlüssels. Der Empfänger erhält die Nachricht, die Signatur “Sig” und den öffentlichen Schlüssel „pub“. Er wiederholt den Vorgang, den der Versender der Nachricht für die Signatur angewandt hat, mit dem öffentlichen, mitgelieferten Schlüsselpaar „pub1“ & „pub2“ anstatt des privaten Schlüsselpaars „k1“ & „k2“. Als Resultat erhält er eine gehashte Variante „HSig“. Sollte jeder Wert der „HSig(n)“ identisch mit jedem Hash-Wert der Signatur „SHA-256(Sig(n))“ sein, ist die Integrität

der Nachricht gewährleistet.

Kapitel 5

Sicherheit

Für die Beantwortung der zugrundeliegenden Thesis müssen die Sicherheitsmechanismen untersucht werden, durch die die Distributed Ledger Technologien vor Angriffen geschützt sind. Die auserwählten Technologien werden darüber hinaus auch darauf geprüft, wie gut die Zahlungsmittel in einer Wallet vor Diebstahl geschützt sind. Dies beinhaltet auch zu prüfen, mit welchen Verschlüsselungsalgorythmen gearbeitet wird und wie hoch die Warscheinlichkeit ist, diesen Algorythmus lösen zu können. Weiterhin wird geprüft, welche Ausmaße fehlerhafte Nutzung der Kryptowährung auf die Sicherheit hat.

5.1 Bitcoin

5.1.1 Wallet Schutz

- wallet anlegen, wie viele verschiedene wallets könnte es geben – wie hoch warscheinlichkeit, dass 2 dieselben keys bekommen

5.1.2 Anonymität

- bezahlen mit bitcoin, was wird über die identität revealed

5.1.3 Validität einer Transaktion

- wie wird eine einzelne transaktion verifiziert, wie wird bestimmt dass sie gültig ist
- wann ist eine transaktion verifiziert

5.1.4 Angriffsszenarios

- wie schützt sich das Bitcoin netzwerk vor Angriffen
- welche angriffsszenarios sind denkbar
- was kann dagegen getan werden

5.1.5 Zukunftssicherheit

- ist die kryptowährung quantenprozessor resistent
- falls bugs gefunden werden, kann man sie fixen?

5.2 IOTA

- IOTA White paper, possible attack scenarios - quantenresistenz durch one time hash - wie wird eine einzelne transaktion verifiziert - wann ist eine transaktion verifiziert

5.3 Gegenüberstellung

[Tabelle mit kriterien und vergleichen]

5.4 Ergebnis

Alles in allem, sind diese kryptos sicher? Auch im Bezug auf zukunftssicherheit

Kapitel 6

Programmierbarkeit

Kapitel 7

Nutzen

7.1 IOTA

IOTA wurde in erster Linie für das „Internet of Things“ entworfen und soll Transaktionen und Kommunikationen sowohl zwischen Menschen als auch zwischen Gerätschaften ermöglichen. Die Vision der IOTA-Foundation beschreibt die Tangle und das dazugehörige Distributed Ledger Protokoll, als eine einheitliche Übereinstimmung der Richtigkeit von Daten. Dieses Prinzip wird über kryptographische Hash Verfahren erreicht. Des weiteren möchte IOTA eine Lösung für den wachsenden globalen Datentransfer liefern, indem das Server - Client Modell durch eine dezentrale Lösung ersetzt wird. Im Folgenden wird die Kommunikation in der Tangle durch die Verwendung von „Masked Authenticated Message“ (kurz MAM) im Detail beleuchtet.

7.1.1 MAM

MAM steht für „Masked Authenticated Message“ und ist die Lösung für das Kommunizieren innerhalb der Blockchain. Es ist möglich durch eine wertlose Transaktion eine Nachricht zu verschicken. Sollen jedoch mehrere Nachrichten versendet werden, bspw. soll die Temperatur eines Motors alle 15 Minuten aktualisiert werden, treten einige Probleme auf. Der Versender kann die Nachricht zwar immer an die gleiche Adresse versenden, aber jeder andere

IOTA Benutzer auch. Bei einem möglichen böswilligen Angriff könnten die validen Nachrichten nicht von den böswilligen unterschieden werden.

MAM basiert auf Kanälen die ein interessierter Hörer „abonnieren“ kann. Es werden Transaktionen generiert die jeweils den Index des Branches, die Geschwister - Knoten des Merkle Trees, die Adresse der nächsten Generation und die Nachricht als Hash enthalten. MAM's verweisen immer auf die nächste Generation, aber nie auf die vergangene Generation oder den Genesis.

Verschlüsselung Abhängig davon in welchem Modus die Daten publiziert sind, benötigt der Hörer entweder ausschließlich den „root“ oder zusätzlich einen “sideKey” um die Daten zu entschlüsseln. In dem Modus „public“ kann der Inhalt durch den „root“ entschlüsselt werden. Sollte der Modus „restricted“ sein, ist es zwar möglich die MAM mit Hilfe des „root“ in der Tangle zu finden, um die Daten jedoch zu entschlüsseln wird ein „sideKey“ benötigt. Derjenige, der die Daten publiziert, kann dementsprechend entscheiden, wer die Daten einsehen kann.

Signatur Ein böswilliger Hörer, der von einem Kanal den „root“ und den „sideKey“ erhält, darf nicht die Möglichkeit haben, dank dieser beiden Daten, den Kanal für sich zu beanspruchen um zukünftig Beiträge in diesem zu verfassen. Der Kanal benötigt eine Signatur. Diese wird nach dem „Merkle tree based signature scheme“ erreicht. Eine Generation ist aufgebaut wie ein Merkle Tree. Die Blätter sind private Schlüssel. Diese Schlüssel werden verwendet um IOTA-Adressen zu generieren. Adressen werden jeweils paarweise miteinander gehasht. Das paarweise Hashen der Ergebnisse wird solange fortgeführt, bis der root Hash erreicht ist. Es können beliebig viele Generationen erstellt werden. Der Hörer nutzt die Adresse der MAM und hasht diesen mit dem ersten, in der MAM angegebenen Geschwister Hash. Das Ergebnis wird mit dem nächsten Geschwister Hash zusammengeführt. Dieser Vorgang wiederholt sich, bis das letzte Paar zu dem sogenannten „temp_root“ zusammengeführt wurde. Sollte der „temp_root“ identisch mit dem root der MAM sein, ist die Nachricht authentisch.

7.2 Bitcoin

Kapitel 8

Beurteilung

Teil II

Praktischer Teil

Kapitel 9

Tanken in der Zukunft

9.1 Projektbeschreibung

loremipsum loremipsum lorem ipsumloremipsum loremipsum lorem ipsumloremipsum loremipsum lorem ipsumloremipsum loremipsum lorem ipsumloremipsum loremipsum lorem ipsumloremipsum loremipsum lorem ipsumloremipsum loremipsum lorem ipsumloremipsum loremipsum lorem ipsumloremipsum loremipsum lorem ipsumloremipsum loremipsum lorem ipsumloremipsum loremipsum lorem ipsumloremipsum loremipsum lorem ipsumloremipsum loremipsum lorem ipsumloremipsum loremipsum

9.2 Projektbegründung

Um eine bessere Aussage über die Zukunftstauglichkeit der Blockchain zu gewinnen, wird gemeinsam mit der IOTA Technologie ein Anwendungsfall prototypisch implementiert. Bei diesem Anwendungsfall handelt es sich um eine Tankstelle an der ein Auto möglichst autonom den Tankvorgang startet und im Anschluss den anfallenden Betrag mit der Kryptowährung IOTA begleicht. Die vorliegende Dokumentation fokussiert sich auf die Implementation der Tankstelle sowie die Kommunikation mit dem Fahrzeug.

9.3 IOTA als Währung

Warum haben wir uns für IOTA entschieden, welche alternativen gibt es, wieso waren die nicht so gut wie IOTA...ä

Kapitel 10

Fahrzeug - Client

10.1 Ziel und Anforderungen

Das Programm soll für einen realitätsnahen Anwendungsfall auf einem relativ leistungsschwachen System arbeiten können. Daher liegt es nahe für die Programmierung des Programmierteils, der für das tanken und bezahlen mit IOTAs zuständig ist, einen Raspberry Pi anzusteuern.

Der Raspberry Pi soll in dieser prototypischen Umsetzung die Steuereinheit eines zukünftigen Autos ersetzen. Es soll mittels zwei Knöpfen bedienbar sein, die die Steuerelemente in der Armatur des Autos darstellen sollen.

Hier noch beschreiben was die anforderungen an das smartauto sind, und graphische nutzerinteraktion anzeigen. Hier soll erklärt werden, was von der tankanwendung gefordert ist, und wie sich das auf die tatsächlich umsetzung auswirkt.

10.2 Aufbau

Unter diesem Aspekt wird der Aufbau und die Einrichtung der einzelnen Komponenten beschrieben die für die Umsetzung nötig sind.

10.2.1 Vorbedingungen

Unter diesem Aspekt werden die Voraussetzungen für die Umsetzung der clientseitigen Tankanwendung beschrieben. Diese Vorbedingungen wurden genutzt um den Raspberry Pi so einzurichten und zu konfigurieren, damit das Gerät die Anforderungen an das Programm erfüllen kann.

Um die prototypische Anwendung umsetzen zu können, werden folgende Komponenten genutzt:

Raspberry Pi 3 Model B

Auf dem Raspberry Pi wird die Anwendung ausgeführt, Daten des Benutzers Verarbeitet und Anfragen auf Webservices vollzogen. Auch die Transaktionen von IOTAs sollen mithilfe des CPUs erstellt werden.

Steckbrett

Das Steckbrett dient der Verbindung von Bedienelementen mit den GPIO-Pins des Raspberry Pis. Die Knöpfe für die Bedienung der Tank-Anwendung werden auf diesem Befestigt und mit den GPIO-Pins elektronisch verbunden.

Micro SD Karte

Diese Speicherkarte wird für den Raspberry Pi gebraucht. Auf dieser Befindet sich das Betriebssystem mit dem der Raspberry Pi erst genutzt werden kann. Auch das Programm und sonstige Benutzerdaten werden auf dieser Speicherkarte gespeichert.

Micro USB Netzteil 5,1 Volt 3,1 Ampere

Der Raspberry Pi muss mittels eines Micro USB Eingangs mit Strom versorgt werden. Das Netzteil sollte eine Ausgangsspannung von mindestens 5,1 Volt aufweisen und es werden mindestens 2,5 Ampere empfohlen.[25]

5x Weiblich-Männlich Kabel, 2x Männlich-Männlich Kabel

Verbindungskabel für das Steckbrett um elektronische Komponenten miteinander zu Verbinden.

2x Taster, 1x Rote LED

Elektronische Komponenten um dem Benutzer der Anwendung Bedienelemente zur Verfügung zu stellen. Die LED dient für ein visuelles Feedback.

Widerstände 1x 330 Ohm, 2x 10.000 Ohm, 2x 1.000 Ohm

Die Widerstände sind für den kontrollierten Stromfluss unentbehrlich und sie verhindern, dass die Komponenten durch zu hohen Stromfluss beschädigt werden.

10.2.2 Einrichtung

Um den Raspberry Pi so einzurichten, dass es möglich ist, Anwendungen zu programmieren und zu testen, sollte dieser mittels eines HDMI-Kabels an einen Monitor angeschlossen werden. Darüber hinaus sollten Peripheriegeräte wie Tastatur und Maus über die USB-Ausgänge angeschlossen werden um sich durch die Bedienoberfläche navigieren zu können.

Damit das Gerät jedoch vollständig bedienbar wird, muss es mit einem Betriebssystem ausgestattet sein. Aufgrund von offiziellem Support seitens der Raspberry Pi Foundation [26] wurde das Raspbian Betriebssystem für die Umsetzung ausgewählt. Bei Raspbian handelt es sich um ein, auf Linux basierendes Betriebssystem, welches von der Raspberry Pi Foundation veröffentlicht wurde.

Das Raspbian Betriebssystem muss auf die Micro SD Karte geladen werden, damit es für den Raspberry Pi zur Verfügung steht. In folgenden Schritten wurde die Speicherkarte mit Raspbian beschrieben:

1. Download von Raspbian (https://www.raspberrypi.org/downloads/)
2. Gegebenenfalls .zip/.rar entpacken

3. Speicherkarte in ein Kartenlesegerät stecken
4. Mithilfe eines Brennprogramms die Imagedatei auf die Speicherkarte brennen (flashen)

Die Speicherkarte ist nun formatiert und mit Raspbian ausgestattet, sie kann nun aus dem Kartenlesegerät genommen werden und in den Raspberry Pi gesteckt werden.

Nachdem das Gerät zum ersten Mal gestartet wird, muss das Betriebssystem einmalig eingerichtet werden. Wenn dies geschehen ist, sollte man jegliche Software-Pakete in Raspbian updaten um mögliche Fehler vorzubeugen. Dies kann mit folgenden Befehlen in dem Terminal des Betriebssystems vollzogen werden:

```
> sudo apt-get update
> sudo apt-get dist-upgrade
```

Steckbrett und GPIO-Pins

Die Taster und LEDs müssen elektrisch mit den GPIO-Pins verbunden werden, um von dem Raspberry Pi angesprochen werden zu können. Die Taster stellen Knöpfe auf der Armatur des Autos dar. Die LED dient dafür, ein visuelles Feedback zu geben, zum Beispiel im Bezug auf die Tank-Anwendung ob nun getankt werden kann.

Dafür ist es nützlich, mit dem Pin-Layout des Raspberry Pis vertraut zu sein. Mit dem Terminal-Befehl

```
> pinout
```

ist es möglich einen Überblick über das Layout des Raspberry Pis zu erhalten. Auch die Benennung der einzelnen Pins ist aufgeführt.

[Bild der „pinout“-Ausgabe einfügen]

Die Komponenten müssen in das Steckbrett gesteckt werden, sodass sie erreichbar bleiben, um sie zu verkabeln und zugleich nutzen zu können. Für eine optimale Übersicht wurden die Komponenten mittig plaziert.

[Bild von der Plazierung der Komponenten Button 1 und 2 und LED]

Für die Umsetzung müssen diese Komponenten für den Nutzer interagierbar sein, dazu werden sie mit zufälligen, freien GPIO-Pins verbunden.

[Tabelle hier einfügen, wie buttons 1 und 2 und LED mit gpios verbunden sind]

Dies erfolgt indem die Weibliche Seite des Kabels in den jeweiligen Pin geführt wird, die andere Seite des Weiblich-Männlich Kabels wird nun in die Reihe des Steckbretts gesteckt an der sich auch die Komponente befindet, die mit dem GPIO-Pin interagieren soll.

Um auf dem Steckbrett einen Stromkreis erzeugen zu können, sollten an den Querreihen des Steckbretts Spannungen angelegt werden. Dafür gibt es auf dem Raspberry Pi Pins, die entweder eine Spannung von 5V, 3,3V oder 0V (GND) haben. Für diese Umsetzung nutzen wir die 3.3V Pins und GND-Pins um einen Stromkreis zu erzeugen, da 3,3V ausreichen um ein Signal auf den Pins zu erfassen. Diese werden jeweils auf die korrespondierende Querreihe mittels Weiblich-Männlich Kabeln verbunden.

Zum Schutz vor Überspannung an der LED muss ein Widerstand in den Stromkreis eingebaut werden, der verhindert, dass die LED zu viel Energie erhält und sie dadurch zu heiß wird und durchbrennt.

Für die Berechnung des passenden Widerstandes empfiehlt es sich, mit der Formel

[Formel für berechnung des widerstandes]

zu arbeiten. Wobei Im = Eingangspannung also 3.3V und Ia = Spannungsverringerung der LED also 0.2A ist. Dies ergibt

[Formel mit eingesetzten parametern]

Hierbei handelt es sich um einen Wert der eine geringe Abweichung erlaubt. Aus diesem Grund wird der nächsthöhere verfügbare Wert angenommen. In diesem Fall ist dies 330 Ohm. Dieser Widerstand muss nun in den Stromkreis mit der LED eingebunden werden.

Damit die Taster eine blabla...

PyOta

Lorem ipsum

10.2.3 Ergebnis der Vorbereitung

Nachdem alle Vorbedingungen erfüllt sind, ist der Raspberry Pi einsatzbereit um ein Programm zu erstellen und auszuführen, welches auf Benutzereingaben an Knöpfen horcht, visuelles Feedback gibt und mit der PyOta-Bibliothek arbeitet.

10.3 Umsetzung

10.3.1 GPIO-Pins

Wie sind die GPIO-Pins eingerichtet wurden, damit man sie ansprechen kann.

10.3.2 IOTA Einrichtung

public fullnode auswählen, fullnode da raspbi zu schwach um POF zu machen, dauert zulange

10.3.3 IOTA Transaktionen

Bezahlen mit IOTA

10.4 Ergebnis

Was tut das Programm nun.

Einmal probedurchlaufen mit dem Programm und erklären wie das mit dem vorher erklärten zusammenhängt

Kapitel 11

Tankstelle - Server

11.1 Projektziel

Die Kommunikation zwischen der Tankstelle und einem Fahrzeug soll durch eine Restful API geschehen. Das Fahrzeug soll imstande sein, durch das Abfragen der implementierten API, einer vorübergehenden ID zugeordnet zu werden, welche für den gesamten Tankvorgang notwendig ist. Auch der Tank - Prozess soll durch gezielte Abfragen an die API gestartet und beendet werden. In einem abschließenden Schritt soll die Tankstellen API die anfallenden Kosten, sowie die notwendige IOTA - Adresse, übermitteln.

Dem Besitzer der Tankstelle muss es möglich sein durch autorisierte Anfragen an die API die IOTA - Adressen zu wechseln oder diesen Prozess durch das hinterlegen eines IOTA - Seed zu automatisieren. Die Adressen, die für den Bezahlvorgang verwendet werden sollen, werden für den autorisierten Besitzer der Tankstelle frei verfügbar sein.

Es ist wichtig anzumerken, dass diese Art der Kommunikation eine von vielen Möglichkeiten ist, dieses Projekt zu realisieren. Im Endeffekt sind folgende Grundvoraussetzung zu erfüllen:

- Dem Kunden darf es nicht möglich sein den IOTA Seed einzusehen.

- Hinterlegte IOTA - Adressen dürfen für unbeschränkt viele Zahlungseingänge, aber ausschließlich für einen Zahlungsausgang verwendet werden.
- Die Aktualisierung der IOTA - Adresse kann automatisiert werden, dies birgt jedoch das Risiko, einen validen IOTA Seed dem Netzwerk zu hinterlegen.
- Autorisierten Personen muss es gestattet sein, neue IOTA - Adressen der Datenbank hinzuzufügen.
- Das System muss Zugang zu dem IOTA - Tangle besitzen und imstande sein dieser Transaktionen hinzufügen zu können.

11.2 Projektaufbau

11.2.1 Vorbedingungen

NodeJS

NodeJS ist eine asynchrone, auf "Events" basierende, JavaScript Laufzeitumgebung, die es ermöglicht mehrere Anfragen zu derselben Zeit abzuarbeiten. Dadurch entstehen niedrige Latenzen zwischen dem Client und dem Server. Desweiteren verfügt NodeJS über eine "EventLoop" die der in Web - Browsern bekannten "EventLoops" sehr ähnelt. Anders als aus Frameworks bekannt, startet die "EventLoop" mit Beginn des ausführenden Scripts und endet sobald keine "Callbacks" mehr vorhanden sind. Dank dieses Verhaltens ist NodeJS für das Entwickeln einer Restful API prädestiniert. [27]

Die Installation von NodeJS gestaltet sich für das Betriebssystem Windows nicht besonders herausfordernd. Auf der Homepage von NodeJS wird eine ausführbare Datei angeboten, welche alle nötigen Programme beinhaltet.

NPM

Durch die Installation von NodeJS wird der hauseigene sogenannte "node package manager" (kurz npm) zusätzlich installiert. Dieses Werkzeug befä-

higt Nutzer vorhandene und publizierte Module für die Entwicklung in NodeJS herunterzuladen oder eigene Module der Öffentlichkeit zur Verfügung zu stellen. Um npm nutzen zu können wird eine funktionierende "Bash" vorausgesetzt.

PowerShell

Um mit NodeJS effektiv arbeiten zu können benötigen alle Programmierer eine geeignete Bash (Shell). Viele Unix und Linux Distributionen besitzen eine geeignete vorinstallierte Bash die für die meisten Befehle ausreicht. Um mit Windows gescheit zu arbeiten, empfiehlt sich die Installation von der PowerShell [28].

Express.js

Eine Restful API basiert auf Anfragen (requests) die an einen Server übermittelt werden. Grundsätzlich folgt auf einen request stets eine Antwort (response). Ein handelsüblicher Webserver verkörpert dieses Prinzip. Sobald in einem Webbrowser die Anfrage zu einer IP - Adresse verschickt wird, antwortet dieser mit den hinterlegten Ressourcen (Meistens vorhandene .html Dateien) [s. Grundlagen "Restful API"]. Um angefragte Routen wie bspw. "/getAddress" zu verarbeiten, welche keine HTML - Daten als Antwort verlangen, werden sogenannte "Middlewares" geschrieben ["s. Grundlagen "Restful API"]. Diese nehmen den angefragten Request entgegen, verarbeiten ihn und senden entweder einen abschließenden Response oder geben den Request weiter an die nächste Middleware - Instanz. Dieser Vorgang wird so lange durchgeführt bis eine beendende Antwort an den Client zurückgesendet wird.

Express.js ist ein Modul für NodeJS, dass dem Programmierer ermöglicht vollwertige Middlewares zu erstellen. Die Installation gestaltet sich sehr leicht. Voraussetzung sind eine funktionierende Bash, sowie der Paketmanager "npm". In der Shell wird folgendes Kommando ausgeführt:

```
npm install express
```

Falls das Paket in einem Projekt abgespeichert werden soll, empfiehlt sich das folgende Kommando:

```
npm install --save express
```

MySQL

Eine MySQL - Datenbank wird für das permanente Speichern der IOTA - Adresse genutzt. Um die Datenbank aufzusetzen empfiehlt sich das Hilfsprogramm XAMPP [29]. Durch dieses Programm kann eine MySQL Datenbank gestartet und über eine lokale Webpage konfiguriert werden. Nach dem Download der ausführbaren Installationsdatei wird diese mit einem Doppelklick gestartet. Im Anschluss genügt es den Installationsanweisungen zu folgen. Nicht alle Komponenten sind notwendig, es genügen der Apache Webserver und die MySQL Unterstützung. XAMPP bietet eine GUI (graphical user interface) mit der neben einer Datenbank auch ein Apache Server gestartet werden kann. Dieser Apache Server ist notwendig um die lokale Konfigurations - Webseite für die MySQL Datenbank zu hosten.

Port Weiterleitung

Damit die API auf zukünftige Anfragen außerhalb des lokalen Systems reagieren kann, muss das Netzwerk entsprechend eingerichtet werden. Der Firewall des zentralen Routers (standard Gateway) werden entsprechende Regeln hinzugefügt, die Anfragen auf den Port 1717 (Dieser Port ist frei wählbar und wurde für dieses Projekt festgelegt) zulassen. Anfragen, die den genannten Port adressieren, werden zu dem Endgerät weitergeleitet, auf welchem die API instanziiert ist.

11.2.2 Projektabgrenzung

Eine geeignete Kommunikation innerhalb eines Netzwerkes kann auf verschiedene Weisen geschehen. Das Verwenden einer Restful API bietet den Vorteil

Informationen und Ressourcen auf vielfältige Art und Weise auszutauschen. Die Kommunikation kann dementsprechend mithilfe eines JSON - Objektes oder über andere Protokolle wie XML oder URL-Encoding gelöst werden. Anders als andere Netzwerk Kommunikationstechniken wie bspw. SOAP sind Serverressourcen direkt adressierbar, ohne das im vorhinein der Inhalt der Anfrage ausgewertet werden muss [30].

11.3 Umsetzung

Mit dem Start der API überprüft das System ob eine JSON-Datei hinterlegt ist, in der die notwendigen Datenbank-Anmeldeinformationen gespeichert sind. Falls diese Datei nicht vorhanden ist, wird eine neue erstellt, indem die Informationen abgefragt werden. Sobald der Zugang zur Datenbank gesichert ist, startet der HTTP-Server und wartet anschließend auf dem eingerichteten Port auf kommende Anfragen. An diesem Punkt sind verschiedene Routen eingerichtet, die sowohl Daten verfügbar machen, mit Daten interagieren oder Daten abspeichern. Diese verschiedenen Routen und ihre Funktionen werden im Folgenden erläutert.

11.3.1 Tank-Vorgang

Der Tankvorgang würde in einer realen Implementierung erst starten können sobald das Fahrzeug vorgefahren ist, die Zapfsäule ausgewählt und die Zapfpistole des notwendigen Kraftstoff in den Tankstutzen eingeführt wurde. Während dessen verbindet das Fahrzeug sich mit dem lokalen Netzwerk mit gegebenen Mitteln. Dieser gesamte Vorgang kann entweder von dem Fahrer oder durch einen Automatismus erfolgen und ist Voraussetzung für den kommenden Tank-Prozesses. Der vorliegende Prototyp schreibt dafür keine Regeln vor.

Die API hört auf den Port 1717 und erwartet dort folgende Anfragen:

- IP-Adresse des Servers/fueling/getStationInfo
- IP-Adresse des Servers/fueling/initializeFueling?ID der Zapfsäule&Kraftstoffart

- IP-Adresse des Servers/fueling/startFueling?zugeordnete ID
- IP-Adresse des Servers/fueling/pauseFueling?zugeordnete ID
- IP-Adresse des Servers/fueling/endFueling?zugeordnete ID
- IP-Adresse des Servers/fueling/getFueling?zugeordnete ID
- getStationInfo

Die Tankstelle liefert die aktuellen Tank-Preise der verschiedenen Kraftstofftypen an das Fahrzeug in Form eines JSON-Objektes.

- initializeFueling

Die API empfängt von dem Fahrzeug die gewählte Kraftstoffart und die ID der Zapfsäule, an der der Tank-Vorgang gestartet werden soll. Diese Daten werden gemeinsam mit einer neu erstellten Kundennummer in der Datenbank hinterlegt. Diese spezielle Zapfsäule ist ab diesem Zeitpunkt für den Tank-Prozess vorbereitet und wird anschließend nur durch die Kundennummer ansprechbar sein. Der Kunde erhält als Antwort die erstellte Kundennummer und kann den Tankvorgang beginnen.

- startFueling

Diese Anfrage startet den Tankvorgang des Fahrzeuges und benötigt die in der Initialisierung vergebene Kundennummer zur Freigabe. Das Tanken wird durch eine Schleife simuliert. Es wird davon ausgegangen, dass innerhalb einer Sekunde ein Liter bzw. Kilowattstunde in das Fahrzeug getankt wird. Die Kosten werden entsprechend aufgerechnet.

- pauseFueling

Auch diese Anfrage muss die in der Initialisierung vergebene Kundennummer übergeben um das Fahrzeug für den aktuellen Tank-Prozess zu autorisieren. Diese Anfrage pausiert den Tankvorgang der durch “startFueling” begonnen wurde. Technisch wird das erstellte Intervall gelöscht und der momentanen Beträge gespeichert.

- endFueling

Diese Anfrage muss die entsprechende Kundennummer enthalten und beendet den Tank-Prozess. Das Kundenfahrzeug erhält die Anzahl der getankten Einheiten, die zugehörigen Kosten und die aktuelle IOTA-Adresse als JSON-Objekt. Die Zapfsäule wird für eine erneute Initialisierung freigegeben. Die Kosten sowie der Betrag der getankten Einheiten werden in der Datenbank zu der entsprechenden Kundennummer gespeichert.

- getFueling

Der Kunde kann über diese Anfrage die Menge des getankten Kraftstoffes und die Kosten erfahren. Dafür muss die entsprechende Kundennummer übergeben werden. Die Antwort geschieht in Form eines JSON-Objektes.

11.4 Ergebnis

Lorem Ipsum

Teil III

Schlussteil

Kapitel 12

Evaluierung

dbsjakdbasdlas

Kapitel 13

Fazit

Literaturverzeichnis

[bal08] BALA, BLA: *Dies ist ein Article Test.* Welt der Wunder, 323(10):192–200, 2008.

[DBT10] DOE, JOHN, BERTOLD BRECHT und TESTI TESTER: *Example Title.* Examplisher, 2010.

[Hig17] HIGGINS, STAN: *From $900 to $20,000: Bitcoin's Historic 2017 Price Run Revisited.* https://www.coindesk.com/900-20000-bitcoins-historic-2017-price-run-revisited/, 2017. Abgerufen am 7. Oktober 2018.

[Nak08] NAKAMOTO, SATOSHI: *Bitcoin Whitepaper.* www.whitepaper.de/bitcoin/sn, 2008. Abgerufen am 7. Oktober 2018.