HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

BACHELORARBEIT

Herr
Maximilian Niemzik

Nutzung von Blockchains unter Berücksichtigung der Privatsphäre nach der DSGVO

Mittweida, 2018

Fakultät Angewandte Computer- und Biowissenschaften

BACHELORARBEIT

Nutzung von Blockchains unter Berücksichtigung der Privatsphäre nach der DSGVO

Autor:
Herr

Maximilian Niemzik

Studiengang:
Medieninformatik und Interaktives Entertainment

Seminargruppe:
MI13w1-B

Erstprüfer:
Prof. Dr.-Ing. Andreas Ittner

Zweitprüfer:
Dipl. Inf.(FH) André Mundo

Einreichung:
Mittweida, 16.07.2018

Verteidigung/Bewertung:
Mittweida, 2018

Faculty Applied Computer Sciences & Biosciences

BACHELORTHESIS

Usage of Blockchain while respecting privacy under the GDPR

author:
Mr.

Maximilian Niemzik

course of studies:
Media Informatics and Interactive Entertainment

seminar group:
MI13w1-B

first examiner:
Prof. Dr.-Ing. Andreas Ittner

second examiner:
Dipl. Inf.(FH) André Mundo

submission:
Mittweida, 16.07.2018

defence/ evaluation:
Mittweida, 2018

Bibliografische Beschreibung:

Niemzik, Maximilian:
Nutzung von Blockchains unter Berücksichtigung der Privatsphäre nach der DSGVO. - 2018. - I, 36, I S.
Mittweida, Hochschule Mittweida, Fakultät Angewandte Computer- und Biowissenschaften, Bachelorarbeit, 2018

Referat:

Die vorliegende Arbeit befasst sich mit der Analyse der Europäischen Datenschutz-Grundverordnung. Das Hauptziel hierbei ist das Herausfiltern von Artikeln die im Zusammenhang mit Datenverarbeitung auf Blockchain eine besondere Relevanz haben. Außerdem wird, basierend auf den gefundenen Artikeln, eine Applikation geschrieben, die die Anforderungen einhält und trotzdem mit einer Blockchain kommuniziert.

Inhalt

1 Einleitung und Zielsetzung

Blockchains werden immer relevanter, sowohl die Anzahl der Transaktionen [15], als auch die Anzahl der Blockchains selbst steigt [10]. Durch enorme Kursgewinne und Verluste gegen den üblichen Fiat Währungen finden sie immer mehr Erwähnungen in Mainstreammedien [17].

Aktuell ist besonders die Ethereum Blockchain hervorzuheben. Sie erlaubte als erstes Turing-Vollständige Smart Contracts, mit denen fast jede Art von Programmen möglich ist. Viele große Projekte die ihre Anwendungen auf Ethereum basieren demonstrieren schon jetzt, das Potential von dezentralisierter Infrastruktur basierend auf Blockchains. Beispiele hierfür sind Augur für transparente Prognosemärkte, Aragon für autonome Organisationen, Golem für verteiltes Mieten und Vermieten von Rechenleistung oder MakerDAO für stabile Währungen.

Aufgrund der besonderen Eigenschaften von Blockchains, wie der freien, öffentlichen Zugänglichkeit und Manipulationresistenz, spielt die im Frühjahr 2018 in Kraft getretene Datenschutz-Grund-verordnung (DSGVO) eine besondere Rolle, wenn sie zur Verarbeitung von personenbezogenen Daten verwendet werden sollen.

Die teils aus den bisherigen Datenschutzgesetzen bekannten und teils neuen Anforderungen der DSGVO an datenverarbeitende Entitäten, werden durch ihre Kopplung an hohe Strafen von bis zu 4% des gesamten weltweit erzielten Jahresumsatzes, aber mindestens 20.000.000 Euro bei bestimmten Vergehen [26], zu ernstzunehmenden Aufgaben.

Aufgrund dieser, nur zunehmenden, Relevanz von Blockchain Anwendungen, soll die Bedeutung der DSGVO für sie geklärt werden. Dazu soll zunächst eine Sortierung der Artikel vorgenommen werden. Sie werden dahingehend untersucht, ob sie für

datenverarbeitende Unternehmen relevant sind, dann weiter ob sie für die datenverarbeitende Software relevant sind und abschließend, ob die daraus abgeleiteten Anforderungen an Programme eine besondere Bedeutung für die Blockchainthematik haben.

Um diese Identifizierung richtig vornehmen zu können, wird Blockchain zunächst auf die wichtigsten Grundlagen runter gebrochen, dazu werden einige notwendige kryptografische Techniken erläutert. Für die gefundenen besonderen Anforderungen wird dann ein Lösungsansatz theoretisch formuliert und anschließend praktisch auf einer Ethereum Blockchain in einem Beispiel umgesetzt. Zum Abschluss wird zusammengefasst inwieweit ein DSGVO konformes Arbeiten mit Blockchains tatsächlich möglich ist. Im Folgenden sind Erwähnungen von Daten und Datensätzen immer als personenbezogene Daten zu verstehen.

2 Grundlagen

Für die Blockchainthematik sind einige, teilweise komplexe Grundlagen und Annahmen sehr wichtig. Da die Arbeit den Fokus auf technischen Aspekten und eine mögliche Implementierung legt, ist ein richtiges Verständnis der Grundlagen unabdingbar. Weiterhin ist es wichtig festzulegen welche Annahmen gelten, weil eine genaue Definition von Blockchains nicht einfach festgelegt werden kann und es sehr viele verschiedene Möglichkeiten gibt, Blockchains zu implementieren. In der Arbeit wird allgemein von einer ethereumartigen Blockchain ausgegangen, doch viele Konzepte lassen sich auch auf andere Chains übertragen.

Des Weiteren wird die DSGVO auf ihre technischen Aspekte herunter gebrochen. Das Verständnis über die Anforderungen ermöglichen eine genau Einteilung der Artikel der DSGVO inwieweit sie die Implementierung von Software beeinflussen, bzw. Anforderungen an die Leistungen der Software stellen. Insgesamt soll das Kapitel die Vorbereitung für das Definieren von Konflikten zwischen typischen Blockchain Eigenschaften und den DSGVO Anforderungen im folgenden Kapitel bereitstellen.

2.1 Technische Grundlagen

Um die Herausforderungen von Datenschutzanforderungen an Blockchain Software zu verstehen, werden zunächst einige Grundlagen der Blockchaintechnologie und wie diese zusammen eine Blockchain bilden, erläutert. Anfangs werden einzelne, notwendige Technologien betrachtet, anschließend wird ein Überblick von Blockchains, anhand von Ethereum gegeben. Da eine genau Definition von Blockchains schwierig ist, werden als Ansatzpunkte entscheidende Merkmale anhand von bekannten Blockchains herausgearbeitet.

Aufgrund des beschränkten Umfangs dieser Arbeit wird ausschließlich auf die notwendigen Eigenschaften und Funktionsweisen eingegangen und auf eine umfassende Erläuterung der einzelnen Punkte verzichtet. Ein gewisses Grundlagenwissen von Kryptographie wird vorausgesetzt. Die im Folgenden verwendeten Ausdrücke "Nicht möglich" und "kann nicht gefunden werden" beziehen sich immer auf den Kontext aktuell verstandener Grenzen von Computern, was bedeutet, dass die Probleme zwar theoretisch lösbar, aber auf absehbare Zeit praktisch unmöglich zu lösen sind.

2.1.1 Hash Funktionen

Allgemein gesprochen, komprimieren Hash Funktionen unterschiedlich lange Bitfolgen auf eine bestimmte Länge. Bei diesem Vorgang gehen Informationen verloren, ein Rückschluss bzw. eine Rekonstruktion der ursprünglichen Nachricht ist nicht direkt möglich. Sie können in mehrere Kategorien unterteilt werden, von besonderer Relevanz für Blockchains sind kryptografische Hash Funktionen. Diese Funktionen können in modification detection codes (MDC) und message authentication codes (MAC) unterteilt werden. MDCs dienen dem Erkennen von Modifikationen von Nachrichten. Hierbei wird eine Nachricht zunächst gehasht und dann sowohl der Hash als auch die Nachricht übertragen. Der Empfänger kann die Nachricht mit dem gleichen Hashing Algorithmus hashen und das Ergebnis mit dem übertragenen Hash vergleichen. Stimmen sie überein wurde die Nachricht nicht verändert. Dieses simple Verfahren macht allerdings im Zusammenhang mit einem böswilligen Gegner keinen Sinn. Wenn er in der Lage ist die Nachricht zu verändern, wäre er auch in der Lage den Hash zu verändern. Er kann seine falsche Nachricht einfach selbst hashen und das Ergebnis weiterleiten. Der eigentliche Empfänger kann eine derartige Veränderung nicht feststellen, da für ihn Hash und Nachricht zusammen passen. [5]

Gelöst werden kann dieses Problem unter anderem mit MACs, da sie eine Authentifizierung der Quelle ermöglichen. Bei einem MAC wird sowohl die Nachricht, als auch ein geheimer Schlüssel gehasht. Ein böswilliger Zwischenmann, kann keinen gültigen Hash erstellen, da ihm der geheime Schlüssel fehlt. Voraussetzung hierfür

ist natürlich die Möglichkeit auf einem sicheren Weg den geheimen Schlüssel zu vereinbaren. Alternativ kann der Hash signiert werden um seine Echtheit zu garantieren. Darauf wird im Kapitel 2.1.2 Signaturen näher eingegangen. [5]

MDCs können bis zu drei besondere Eigenschaften haben, die im Folgenden erläutert werden. Preimage resistance, die besagt, dass für einen bekannten Hash h kein möglicher Input x gefunden werden kann, wenn noch kein möglicher Input bekannt ist, sodass $H(x) = h$. Second preimage resistance besagt, dass wenn für einen Hash h bereits eine mögliche Ursprungsnachricht x bekannt ist, es nicht möglich ist eine weitere Ursprungsnachricht x' zu finden, die den gleichen Hash h hat, also das $H(x) \neq H(x')$. Außerdem können die Funktionen noch collision resistant sein, das besagt, dass es nicht möglich ist zwei frei wählbare Nachrichten x und x' zu finden, die den gleichen Hash h haben, also $H(x) \neq H(x')$. Es lässt sich eine klare Abstufung hinsichtlich der Sicherheit erkennen. Second preimage resistance schließt preimage resistance automatisch ein und collision resistance geht sogar noch darüber hinaus. [5]

2.1.2 Signaturen

Digitale Signaturen dienen in erster Linie dem Authentifizieren und dem Sichern von Datenintegrität. Verschiedene Algorithmen zum Signieren können unterschiedliche Eigenschaften haben. Für ein vollständiges Signierschema muss mindestens die Möglichkeit zum Signieren und die des Verifizierens gegeben sein. Außerdem sollte es für einen böswilligen Aktor technisch nicht möglich sein die Signatur zu fälschen. Hierbei gibt es verschiedene grundlegende Angriffsmethoden, wie key-only oder message Angriffe. Bei key-only kennt der Angreifer nur den public key des Signierenden. Bei message Angriffen gibt es verschiedene Klassifizierungen, entweder der Angreifer kennt nur Signaturen für Nachrichten, die er nicht wählen kann, oder er kennt die Signaturen von selbst gewählten Nachrichten, die vor dem Versuch des Angriffs, erstellt wurden. Im Extremfall, kann er während eines Angriffes frei wählen, welche Nachrichten signiert werden. Je nach Anwendungsfall muss ein Algorithmus gewählt werden, der den Umständen entspricht. [6]

Erfolgreiche Angriffe werden in drei Kategorien unterteilt. Ein totaler Bruch ist der schlimmste anzunehmende Fall. Hierbei kann der Angreifer, entweder den privaten Schlüssel des Signierenden aus den Signaturen berechnen, und damit beliebige Nachrichten signieren, oder er findet einen effizienten Algorithmus der valide Signaturen ohne den privaten Schlüssel des Signierenden produziert. Nicht ganz so verehrend ist eine selective forgery, hierbei kann der Angreifer eine gültige Signatur für eine vor dem Angriff gewählte Nachricht erstellen. Der schwächste erfolgreiche Angriff ist die existential forgery. Dabei kann der Angreifende für wenigstens eine Nachricht, die er nicht frei wählen kann, eine gültige Signatur erstellen. [6]

Als Authentifizierungsmechanismus können Signaturen beispielsweise zur Erkennung einer Zugangsberechtigung verwendet werden. Ein System, das bestimmte öffentliche Schlüssel registriert hat, kann anhand von gesendeten Signaturen erkennen, ob der Sender eine geeignete Zugangsberechtigung hat. Alternativ ist es möglich zu erkennen, ob ein bestimmter Sender wirklich der ist, der vorgibt zu sein, wenn er seine Nachricht signiert. Das ist möglich, sofern vorher vereinbart wurde unter welchem öffentlichen Schlüssel er agiert. [6]

Ein anderer Anwendungsfall ist das Schützen der Integrität von Nachrichten. Meist wird das in Kombination mit Hashfunktionen umgesetzt. Zunächst wird die Nachricht gehasht, der Hash wird anschließend signiert, dann werden sowohl Hash als auch Nachricht übertragen. Der Empfänger kann dann die Signatur überprüfen, und kontrollieren ob sie mit der vorher vereinbarten Identität übereinstimmt. Ist das der Fall hasht er die Nachricht selbst und überprüft, ob das Ergebnis und der signierte Hash identisch sind. Ist auch das der Fall kann er davon ausgehen, dass die Nachricht sowohl vom korrekten Sender kommt, als auch unverändert ist. Es wird meist nur der Hash signiert, da die Signierung der gesamten Nachricht zu rechenintensiv ist. Es ist vorteilhafter die zusätzliche Rechenleistung in ein aufwändigeres Signierungsverfahren zu investieren, da bei der richtigen Wahl der Hashfunktion davon ausgegangen werden kann, dass der Hash ein nicht fälschbarer Fingerabdruck der Nachricht ist. [5]

2.1.3 Blockchain

Eine genaue Definition der Blockchain Technologie ist äußerst schwierig, da unterschiedliche Elemente auf verschiedene Arten und Weisen implementiert werden können. Hinzu kommt, dass Blockchain nicht alleine ein Informatik Thema ist, sondern Komponenten von Game-Theory, Ökonomie und Mathematik enthält. Außerdem, können je nach Wahl der einzelnen Elemente, Hardware oder soziale Interaktion eine erhebliche Rolle spielen.

Einige wenige öffentliche Blockchains heben sich stark von der Masse der Restlichen gemessen an der Anzahl von Transaktionen pro Zeiteinheit. Da das ein starker Indikator für tatsächliche Nutzung ist, werden sie als Richtwert genommen, um festzustellen was die Technologie in einer praktikablen Form selbst aus macht. Zu sehen ist das in Tabelle 2.1, Ethereum und Bitcoin stechen deutlich, um den Faktor 10-20, gegenüber den nächst größeren Chains hervor. Bitcoin Cash ist als direkte Fork von Bitcoin, mit wenigen Änderungen, technisch weitergehen identisch zu Bitcoin.

Netzwerk	Transaktionen/Tag
Ethereum	565,164
Bitcoin	206,898
Bitcoin Cash	103,532
Dogecoin	27,064
Litecoin	26,346
Dash	16,393

Table 2.1: Auswahl von Netzwerken mit vielen Transaktionen pro Tag vom 03.07.2018 [4]

Allgemein wird im Folgenden von öffentlichen, zugangsunbeschränkten Blockchains ausgegangen. Da sie in erster Linie der dezentralen Konsensfindung dienen, finden sie hauptsächlich in dezentralen Netzwerken Anwendung. Die Unveränderbarkeit und Sicherheit über die korrekte Ausführung von Berechnungen sind hauptsächlich in Netzwerken relevant, in denen die Vertrauensbasis der Akteure nicht gesichert ist.

Essentiell ist eine Datenstruktur bestehend aus sogenannten Blocks oder Blöcken, die in einer eindeutigen Abfolge aneinandergehängt werden. Eine besondere Rolle

spielen dabei Hashfunktionen, (siehe Kapitel 2.1.1), die die Blöcke unwiderruflich aneinanderkoppeln. Blöcke können grundsätzlich in ihren Header und ihre Transaktionen geteilt werden. Im Header werden je nach Blockchain unterschiedliche Daten gespeichert. Jedoch wird immer der Hash des vorherigen Blockheaders referenziert. Durch die Header entsteht also eine direkte Verknüpfung, da sie sich immer einer nach dem anderen beinhalten. Der Blockinhalt selbst wird in diese Verknüpfung, durch seine Referenzierung im Header, eingebunden. Das hat zur Folge, dass bei jedweder Veränderung eines der Blöcke der gespeicherte Hash im folgenden Block nicht mehr übereinstimmt. Eine Verifikation der Korrektheit der Blöcke kann also sehr einfach durchgeführt werden. Korrektheit bedeutet in diesem Fall, dass die Blöcke in der Kette, tatsächlich zu den Inhalten passen der vorgegeben wird. Alleine damit kann aber keine Aussage über die Korrektheit der Daten, bzw. der Statetransitions, getroffen werden, das muss separat erfolgen. Durch die Verkettung von Blöcken kann die Unveränderbarkeit der Daten sichergestellt werden. [22, p. 4]

Blockchains sind aber keine reinen Datenstrukturen. Es werden zusätzlich Berechnungen durchgeführt, die zur Erschaffung neuer Blöcke führen. Der Inhalt von Blöcken besteht aus sogenannte Transaktionen, die den State der Blockchain beeinflussen. Der aktuelle State der Chain kann beispielsweise aus unspend transaction ouputs (UTXO), wie in Bitcoin, bestehen [22, p. 3]. Neue Transaktionen beeinflussen den aktuellen State, der selbst allerdings nicht in den Blöcken steht. Werden die gesamten Transaktionen aller bisherigen Blöcke ausgeführt wird der aktuelle State erstellt. Die Korrektheit des Statetransitions kann durch alle Nodes, oder in manchen Implementierungen oder Protokollen, durch bestimmte, meist Master-Nodes genannte, Nodes verifiziert werden. Gewöhnlich werden inkorrekte Blöcke durch die Nodes abgelehnt, sie tragen also nicht zum State bei. Die Korrektheit von Berechnungen kann dementsprechend von der eignen Node überprüft werden. Der Umfang der möglichen Rechenoperationen hängt von der Implementierung ab. Ethereum hat beispielsweise eine Turing-Vollständige virtuelle Maschine, wohin gehen andere Chains nur bestimmte Stateveränderungen zulassen. [32]

Das Erstellen neuer Blöcke wird meist durch sogenannte Miner, je nach Protokoll auch Signer oder Sealer, durchgeführt. Ihre Aufgabe ist es, Transaktionen aus

dem Netzwerk zu sammeln, die Stateveränderungen zu überprüfen und die fertigen Blöcke in das Netzwerk zu senden. Je nach Protokoll zur Blockerstellung muss gegebenfalls noch eine bestimmte rechenintensive (Proof-of-Work (PoW)) [22, p. 3], oder speicherintensive(Proof-of-Capacity (PoC)) [28] Nonce gefunden werden, oder bestimmte Nodes wird die Berechtigung gegeben abwechselnd Blöcke zu erstellen (Proof-of-Authority (PoA)) [29]. Eine weitere Möglichkeit ist, dass Miner eine gewisse Kaution zahlen müssen, die als Sicherheit dient, falls ein Miner falsche Blöcke produziert (Proof-of-Stake (PoS)) [7]. Es gibt darüber hinaus noch einige andere Protokolle. Da den Miner eine gewisse Verantwortung obliegt, und sie eine Art von "Arbeit" aufwenden müssen um Blöcke erstellen zu können, bekommen sie eine Belohnung. Davon ausgenommen ist PoA, da hier keine Arbeit nötig ist. Die Arbeit ist notwendig damit bösartige Akteure, die in einem nicht zugangsbeschränktem Netzwerk nicht ausgeschlossen werden können, nicht mit falschen Blöcken das Netzwerk fluten. Dies würde sowohl dem Netzwerk als auch den einzelnen Nodes viel Overhead auferlegen, oder sie sogar lahmlegen. Zusätzlich verhindert der Aufwand der Berechnungen Long-Range-Attacken bei denen ein großer Teil der bestehenden Chain zurückgerollt wird [22, p. 6-7].

Die Belohnung der Miner hat eine netzwerkeigene Währung zur Konsequenz, die meist allen anderen Bestandteilen als Hauptfunktion von Blockchains vorangestellt wird. Das Kontenbuch dieser Währung ist meist ein essentieller, oder sogar der einzige, Bestandteil von Stateveränderungen in der Chain. Um das Netzwerk am laufen zu halten, ist es im Interesse der Netzwerkteilnehmer dieser Währung einen Wert zu verleihen, da mit ihr die Miner für ihre Arbeit zu belohnt werden. Dies kann beispielsweise durch das Kaufen der Währung geschehen. Eine Blockchain Währung, die verbreitet genug ist, kann auch alleine dadurch Wert haben, dass mit ihr direkt Güter oder Dienste gekauft werden können.

Das letzte grundlegende Element von Blockchains ist die Konsensfindung drüber, welche Kette von Blöcken die "Wahre" ist. Da es möglich ist unterschiedliche korrekte Blöcke auf einen bestimmten Block aufzubauen, können Verzweigungen entstehen, sogenannte Forks. Da ein Fork eine Spaltung des Netzwerks zur Folge hat, sollten sie vermieden werden. Zusätzlich muss der State für alle Beteiligten gleich sein, sonst könnten Double-Spends durchgeführt werden. Das einfachste Beispiel von zwei korrekten Blöcken ist ein Block mit einer korrekten Transaktion und ein

leerer Block, da der Miner entscheidet welche Transaktionen in die Blöcke gehen. Die Nodes müssen entscheiden, welche der beiden Chains die "Wahre" ist. Denn nach zwei unterschiedlichen Blöcken ist der State der jeweiligen Chain auch unterschiedlich, beispielsweise könnten die beiden Miner die Transaktionen im Block unterschiedlich angeordnet haben, daraus resultieren unterschiedliche Blockhashes. Doch selbst im einfachsten Fall, wenn zwei Miner beide leere Blöcke minen ist der State unterschiedlich, denn in jeder Chain bekommt ein anderer die Blockbelohnung. Da ein inkonsistenter State eigentlich durch Blockchains verhindert werden soll kann es nur eine "Wahre" Chain geben.

Für den Konsensmechanismuss gibt es wieder einige verschiedene Protokolle. Das wahrscheinlich einfachste ist die Wahl nach der längsten Chain in Bitcoin (Longest-Chain). Hierbei sind die Blöcke gültig, die die längste aufbauende Chain haben. Wenn auf Block n zwei verschieden Blöcke (n1' ,n1) aufsetzten, ist zu dem Zeitpunkt nicht sicher, welcher der Richtige ist (außer es wird lokal der erste eingetroffene, aber dann ist trotzdem noch unbekannt welchen der Rest des Netzwerks genommen hat). Wenn nun auf Block n1 ein weiterer Block n2 aufsetzt wird die Kette basierend auf n1 als die Wahre angenommen, sollte nun bevor n3 erstellt wird, n2' und n3' erstellt werden gilt die Chain auf n1' als die Wahre [18]. Dargestellt in Abbildung 2.2. Dieser Vorgang organisiert die Chain neu und wird auch Reorg genannt. Damit solche Reorgs nicht ständig passieren, existieren die oben genannten Blockfindungsmechanismen, sie verhindern das es einfach ist mehrere Alternativblöcke in der Zeit die ein neuer Block braucht erstellt werden könne. Das ist wichtig, da bei Reorgs alle Statetransitions bis zum Punkt des Fork zurückgerollt werden. Besonders bei Finanztranskationen ist sehr unerwünscht und wird hier als Double-Spend Attacke bezeichnet [22].

Ein anderes Protokoll ist zum Beispiel die Wahl nach dem Greedy Heaviest Observed Subtree (GHOST), das in vereinfachter Version in Ethereum verwendet wird. Hierbei kann die Blockzeit durch die Einführung sogenannter Uncles deutlich verkürzt werden. Uncles sind Blöcke mit validem Blockheader, die in validen Blöcken referenziert werden. Durch das nennen von Uncles in Blöcken erhält sowohl der Miner des Blocks, als auch der des Uncles eine extra Belohnung. [33]

Zusätzlich gibt es noch Mechanismen die einen größeren Reorg verhindern, beispiel-

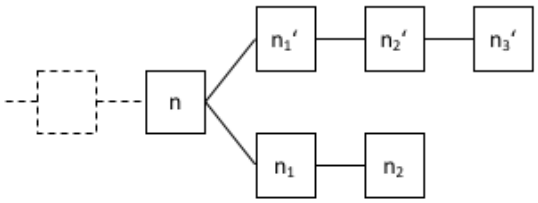

Figure 2.2: Einfache Darstellung der Longest Chain Regel. Selbst wenn n_2 vor n_2' gefunden wurde, ist er trotzdem ungültig, wenn n_3' vor n_3 gefunden wird.

sweise Casper FFG, welches nach bestimmten Abständen Blöcke finalisiert, die dann nicht mehr zurückgerollt werden können [31].

2.1.4 Ethereum

Die Ethereum Blockchain stellt mit ihren Turing-vollständigen Smart Contracts eine Plattform dar, auf der Datenverarbeitung grundsätzlich machbar ist. Durch verschiedene Eigenschaften, die Blockchains bieten, wie Transparenz, Sicherheit und Redundanz, ist Ethereum ein attraktives Ziel für kritische Programme.

Für den Aufbau der Chain ist die Keccak-256 Hashfunktion unerlässlich, da sie unter anderem die einzelnen Blöcke verlinkt. Zusätzlich ist die Funktion in Verbindung mit Merkle-Trees unerlässlich für den internen Aufbau der Blöcke [32]. Sie findet außerdem noch weitere Anwendungsfälle im Bezug zu Light-Clients [8], bei der Contract Erstellung [14] und bei dem Signieren von Transaktionen. Bei Letzteren wird nur der Keccak-256 Hash der Transaktion signiert [32]. Signaturen, die den Besitz eines Accounts beweisen, werden mit dem ECDSA Algorithmus auf der SECP-256k1 Kurve signiert [32].

Transaktionen werden in der stackbasierten Ethereum Virtual Machine (EVM) ausgeführt. Diese fast vollständige Turing-Maschine wird nur durch den Parameter Gas limitiert, der in Form des Blockgaslimits die maximale Anzahl von Berechnungsschritten pro Block definiert. Verschiedene Operationen kosten eine unterschiedliche Menge an Gas, zu sehen in Tabelle 2.3, was grob auf den tatsächlichen

Rechenaufwand eines CPUs zurückzuführen ist. Für jede Transaktion muss eine Gebühr entrichtet werden, die sich nach dem Gasverbrauch der Transaktion und einem vom Nutzer festgelegten Preis pro Gaseinheit ergibt. Auf der Blockchain können durch bestimmte Funktionen in der EVM Smart Contracts mit vorher festgelegten Bytecode erschaffen werden. Diese Contracts sind ab dem Zeitpunkt des Erstellens über ein Application Binary Interface (ABI) an ihrer Adresse ansprechbar. Da Bytecode und Speicher des Contracts Teil der Blockchain sind, kann dieser jederzeit von jedem eingesehen werden. Funktionsaufrufe im Contract können allerdings beschränkt werden. So ist es beispielsweise möglich, das Ausführen einer Funktion auf den Besitzer eines bestimmten privaten Schlüssels zu beschränken, der durch eine Signatur nachgewiesen werden muss. Auf eine genauere Einführung wird an dieser Stelle auf Grund des Umfangs verzichtet. [34]

Operation	Gaskosten
ADD	3
SUB	3
MUL	5
ADDMOD	8
BLOCKHASH	20
CREATE	32000

Table 2.3: Gaskosten für verschiedene Operationen [35]

Der Konsensmechanismus und die PoW Implementierung wird hier vernachlässigt, da sie keinen direkten Einfluss auf das Durchführen von Rechenoperationen in der EVM hat.

2.2 DSGVO

2 Jahre nach ihrer Verabschiedung trat die DSGVO am 25.05.2018 in Kraft. Sie ersetzt das bisher in Deutschland geltende Datenschutzgesetz und stellt eine umfassende Vereinheitlichung des gesetzlichen Datenschutzes in der EU dar. Bei einem so umfassenden Gesetz betreffen nicht alle Artikel direkt die Arbeitsweise der betroffenen Unternehmen oder Bürger. Ein Teil des Gesetzes legt auch Dinge fest, die weder Rechte für den Einzelnen, noch Pflichten für die Betroffenen zur Folge haben.

Selbst Teile des Gesetzes, die direkten Einfluss auf die Arbeitsweise von datenverarbeitenden Unternehmen haben, sind nicht immer relevant für die Implementierung von Software. [23]

In den nächsten Unterkapiteln wird eine Trennung vorgenommen, bis nur noch Artikel, die direkte Anforderungen an die technische Umsetzung von Anwendungen stellen, übrigbleiben. Diese relevanten Artikel werden dann, nach Möglichkeit, als Anforderungen an datenverarbeitende Software definiert.

2.2.1 Identifizierung anwenderrelevanter Artikel

Die 99 Artikel der DSGVO werden nach einem Ausschlussverfahren in zunächst zwei Gruppen geteilt. Alle Artikel, die weder Einzelpersonen Rechte gegenüber Datenverarbeitern geben, oder Datenverarbeitern Pflichten abverlangen zählen als nicht anwenderrelevant, da sie keinen direkten Einfluss auf ihr Handeln haben. Zunächst werden alle nicht anwenderrelevanten Artikel aufgezählt und es wird erklärt warum sie das sind.

Artikel 1 (Gegenstand und Ziele) ist lediglich eine Zusammenfassung über den Inhalt und den Zweck der Verordnung. Artikel 4 (Begriffsbestimmungen) ist zwar indirekt relevant, da wichtige Begriffe genauer in ihrer Bedeutung beschrieben werden, erfordert selbst aber keinen Handlungsbedarf, bei datenverarbeitender Software. Artikel 23 ermöglicht es der Union und ihren Mitgliedstaaten unter bestimmten Voraussetzungen gewisse Rechte und Pflichten zu beschränken [27, p. 10], folglich ist der Artikel im normalen Regelbetrieb irrelevant. Die Artikel 40/41 beschreiben ein vorgesehenes System zur Förderung von Verhaltensregeln, legt selbst jedoch keine fest [27, p. 12]. In den Artikeln 42/43 steht, dass Verantwortliche und Auftragsverarbeiter bestimmte Verarbeitungsvorgänge zertifizieren lassen können [27, p. 12]. Es ist also keine Pflicht, deshalb kann auch dieser Artikel aussortiert werden.

Die Kapitel 6-8 (Artikel 51-84) befasst sich ausschließlich mit unabhängigen Aufsichtsbehörden, die mit bestimmten Befugnissen zur Einhaltung der Verordnung beitragen sollen [27, p. 14]. Außerdem beschreiben sie die Zusammenarbeit zwischen den einzelnen Behörden [27, p. 15] und mögliche Strafen und Sanktionen, die

im Falle eines Verstoßes gegen die DSGVO verhängt werden können [27, p. 16]. Auch wenn Strafen einen direkten Einfluss auf betroffenen Datenverarbeiter haben, werden sie in dieser Arbeit vernachlässigt, da es ausschließlich um DSGVO konforme Anwendung geht. Eine etwaige Abschätzung, ob Strafen in gewisser Höhe verhängt werden könnten, wird in dieser Arbeit nicht getroffen.

In den Kapiteln 9-11 (Artikel 85-99) geht es um besondere Verarbeitungssituationen [27, p. 18], einige Pflichten der EU-Kommission und das geregelte Inkrafttreten der Verordnung [27, p. 18]. In der Arbeit wird von normalen Verarbeitungssituationen ausgegangen, außerdem sind Regelungen zur Arbeit der Kommission und zum Inkrafttreten der Regelung nur dahingehen relevant, dass sie den Gültigkeitsbeginn der Verordnung festlegen.

Nach dem eingangs beschriebenen Vorgehen, bleiben 56 Artikel die nicht relevant für Anwender sind und 43 Artikel die einen direkte Einfluss auf sie haben.

2.2.2 Trennung von Rahmen und technischer Relevanz

Diese 43 Artikel lassen sich wiederum in zwei Gruppen aufteilen. Es kann nach Artikeln, die direkt in einer technischen Lösung implementiert werden können bzw. beim Bau von Software berücksichtigt werden müssen und nach Artikeln die, beispielsweise für eine Firma die Daten verarbeitet, als notwendige Rahmenbedingen angesehen werden müssen, unterschieden werden. Es wird also zwischen dem tatsächlichen Verarbeiten von Daten und allen anderen Anforderungen der Verordnung an Datenverarbeiter unterschieden.

Artikel 3 (Räumliche Anwendung) legt fest in welchen Ländern die DSGVO gilt [27, p. 6], da sich die Arbeit mit dem Erfüllen der DSGVO befasst, sind Artikel, die die Pflicht des Erfüllens abgrenzen irrelevant. In den Artikeln 5-8 geht es um die Grundsätze und Rechtmäßigkeit der Verarbeitung von Daten, welche als Rahmenbedingung vorausgesetzt wird und kann beispielsweise per Einwilligung der betreffenden Person gegeben sein [27, p. 8]. Sollte Artikel 9 zutreffen, sind besondere Maßnahmen zu treffen. Diese betreffen aber auch wieder nur den Rahmen der Einwilligung, haben also keinen direkten Einfluss auf die Verarbeitung der Daten selbst [27, p. 9]. Zusätzlich beschreibt Artikel 10 unter welchen Voraussetzungen personenbezogene Daten

über strafrechtliche Verurteilungen und Straftaten verarbeitet werden dürfen [27, p. 8].

Artikel 11 legt fest, das Daten nicht bloß deshalb aufbewahrt werden müssen, um eine Person identifizieren zu können, was ebenfalls wieder eine organisatorische Frage für den Datenverarbeiter darstellt und damit keine Relevanz für die Datenverarbeitung selbst hat. Der Artikel 12 fasst die folgenden Artikel 13-15 zusammen, welche die Regelung der Datenschutzrechte, die einer betroffenen Person zukommen umfasst [27, p. 10]. Bei diesen Artikeln geht es um die Informationspflichten gegenüber eines Betroffenen, zum Beispiel unter welcher Rechtsgrundlage und zu welchem Zweck die Daten verarbeitet werden [27, p. 10]. Auch das ist wieder eine Rahmenbedingung, da der Verarbeiter technologieagnostisch in der Lage sein muss, zu wissen, wie und wieso er welche Daten verarbeitet. Artikel 19 verlangt das betroffene Personen im Zusammenhang mit der Berichtigung oder Löschung personenbezogener Daten oder der Einschränkung der Verarbeitung, nach den Artikeln 16-18 benachrichtigt werden. Wie dies geschieht ist egal, solange die betroffene Person tatsächlich in Kenntnis versetzt wird, weshalb der Artikel keinen direkten Einfluss auf datenverarbeitende Software ausübt.

In den Artikeln 26-29 wird die Rolle des Verantwortlichen genauer definiert, außerdem werden Auftragsverarbeiter, die nicht im Unionsgebiet niedergelassen sind, dazu verpflichtet einen Vertreter in einem Mitgliedsstaat zu benennen [27, p. 11]. Sie stellen also einen wichtigen Rahmen dar, den Datenverarbeiter erfüllen müssen. Das verpflichtende Verzeichnis von Verarbeitungstätigkeiten aus Artikel 30 [27, p. 11], ist wiederum ein organisatorisches Element. Der Artikel 31 legt außerdem fest, dass die Verantwortlichen aus den Artikeln 26-29 mit den Aufsichtsbehörden zusammenarbeiten sollen.

Die Artikel 33/34 verlangen eine Meldung von Verletzungen des Schutzes personenbezogener Daten an die Aufsichtsbehörde [27, p. 11]. Was wiederum durch übliche Kanäle erfolgt, also nichts mit der Datenverarbeitung selbst zu tun hat, außer der Notwendigkeit solche Datenbrüche zu erkennen. Gegebenenfalls muss eine Datenschutzfolgeabschätzung nach Artikel 35/36 in einem Konsultationsverfahren mit einer Aufsichtsbehörde durchgeführt werden. Auch das ist für die technologische

Umsetzung selbst irrelevant, der Artikel selbst beschreibt allein die Notwendigkeit der Kommunikation mit den Behörden unter bestimmten Bedingungen.

Durch die Artikel 37-39 wird der Datenschutzbeauftragte definiert. Dieser muss unter Umständen durch den Datenverarbeiter gestellt werden. Somit haben auch diese Artikel nichts mit der Datenverarbeitung selbst zu tun, es muss lediglich bekannt sein wie die Daten verarbeitet werden [27, p. 12]. Abschließend sind noch die Artikel 44-46 und 49 für korrekte Rahmenbedingen relevant, da sie den Datenverkehr mit Empfängern in Drittstaaten oder internationalen Organisationen regeln [27, p. 13].

Insgesamt sind also 33 Artikel für die Rahmenbedingungen relevant, welche ein Datenverarbeiter auf jeden Fall leisten muss. Übrig bleiben 10 anwenderrelevante Artikel die einen direkten Einfluss auf mögliche Implementierung von datenverarbeitendender Software haben. Diese werden anschließend näher besprochen.

2.2.3 Relevante Artikel

Die 10 Artikel, die eine direkte Relevanz für weiterverarbeitende Software haben, wurden im Folgenden zusammengefasst. Ihr direkter Wortlaut ist aus der DSGVO [11] zu entnehmen.

Art. 2 Sachlicher Anwendungsbereich Der Anwendungsbereich der Verordnung betrifft ganz oder teilweise automatisierte Verarbeitung von personenbezogenen Daten, die in einem Dateisystem gespeichert werden.

Art. 5 Grundsätze für die Verarbeitung personenbezogener Daten Wichtige zu nennende Grundsätze sind, dass die Daten rechtmäßig und für die betroffenen Personen nachvollziehbar verarbeitet werden, sie für festgelegte, eindeutige und legitime Zwecke erhoben werden, auf das notwendige Maß beschränkt sind, sachlich richtig und auf dem neusten Stand sind und auf eine Weise verarbeitet werden, die eine angemessene Sicherheit der Daten gewährleistet, einschließlich dem Schutz vor unbefugter oder unrechtmäßiger Verarbeitung, Verlust oder Vernichtung.

Art. 16 Recht auf Berichtigung Dieser Artikel gibt betroffenen Personen das Recht, von dem Verantwortlichen unverzüglich die Berichtigung unrichtiger personenbezogener Daten zu verlangen.

Art. 17 Recht auf Löschen Beschreibt das "Recht auf Vergessenwerden", das einer betroffenen Person das Recht gibt, die sie betreffenden personenbezogenen Daten, unter bestimmten Gründen, unverzüglich löschen zu lassen.

Art. 18 Recht auf Einschränkung der Verarbeitung Dieser Artikel gibt einer betroffenen Person, unter bestimmten Voraussetzungen, das Recht, von dem Verantwortlichen die Einschränkung der Verarbeitung zu verlangen.

Art. 20 Recht auf Datenübertragbarkeit Dieser Artikel gibt betroffenen Personen das Recht, die sie betreffenden Daten, in einem strukturierten, gängigen und maschinenlesbaren Format zu erhalten, oder direkt die Übertragung an einen anderen Verantwortlichen zu veranlassen.

Art. 21 Widerspruchsrecht Das Widerspruchsrecht legt fest, dass es einer betroffenen Person gestattet ist, aus Gründen, die sich aus ihrer besonderen Situation ergeben, jederzeit gegen die Verarbeitung sie betreffender personenbezogener Daten Widerspruch einzulegen.

Art. 22 Automatisierte Entscheidungen im Einzelfall einschließlich Profiling Beschreibt das Recht einer Person auf eine, nicht auf automatisierte Verarbeitung beruhende, Entscheidung, sofern diese ihr gegenüber eine rechtliche Wirkung entfaltet, oder sie in ähnlicher Weise erheblich beeinträchtigt.

Art. 25 Datenschutz durch Technikgestaltung und durch datenschutzfreundliche Voreinstellungen In diesem Artikel wird festgelegt, dass der Verarbeiter geeignete technische und organisatorische Maßnahmen zur Umsetzung von Datenschutzgrundsätzen trifft und geeignete Voreinstellungen für betroffene Personen festlegt.

Art. 32 Sicherheit der Verarbeitung Die Verantwortlichen haben geeignete technische und organisatorische Maßnahmen zu treffen, um ein dem Risiko angemessenes Schutzniveau für die personenbezogenen Daten zu gewährleisten, unter

> anderem durch Verschlüsselung und Pseudonymisierung und vertrauliche, integere, verfügbare und belastbare Systeme und Dienste im Zusammenhang mit der Verarbeitung der Daten zu schaffen. Außerdem sind bei der Beurteilung des angemessenen Schutzniveaus insbesondere die Risiken zu berücksichtigen, die mit der Verarbeitung verbunden sind.

Die Anforderungen, die die relevanten Artikel stellen, werden im Folgenden näher betrachtet. Laut Artikel 2 sind sämtliche automatisierte und teilweise automatisierte Verarbeitungsprozesse von der DSGVO betroffen. Folglich muss sie, bei jeder Software, die personenbezogene Daten verarbeitet, berücksichtigt werden. Artikel 5 beschreibt wichtige Grundlagen zur Verarbeitung der Daten. Die beschriebenen Anforderungen werden in den Artikeln 16-18, 25 und 32 weiter ausgeführt. So stellt beispielsweise Artikel 16 die Anforderung, dass es möglich sein muss, bestehende Daten unverzüglich zu ändern, bzw. zu berichtigen. Um Artikel 17 einzuhalten, muss es möglich sein, bestehende Datensätze zu löschen. Weiterhin wird beschrieben unter welchen Voraussetzungen das Löschen geschehen kann. Das wiederum ist für die reine Softwarelösung nicht von Relevanz, da für die Einhaltung der Regelung auf jeden Fall davon ausgegangen werden muss, dass die Artikel Anwendung finden werden. Ob der Antrag auf Löschung rechtmäßig ist, kann entweder durch eine separate Software, oder aber manuell bestimmt werden. Wichtig für die datenverarbeitende Software ist nur die Möglichkeit der Löschung. Laut Artikel 18 muss es der Software möglich sein, die Verarbeitung der Daten einzuschränken, es geht also nicht um direktes Löschen, sondern darum, die Daten weiterhin zu speichern, aber nicht mehr zu verarbeiten, bzw. zu benutzen. Ergänzend dazu beschreibt Artikel 21 das Widerspruchsrecht, welches es betroffenen Personen ermöglicht, jederzeit aus bestimmten Gründen Widerspruch gegen die Verarbeitung ihrer Daten einzuheben. Sollte der Widerspruch rechtens sein, muss der Verantwortliche entsprechend beispielsweise die Verarbeitung der Daten stoppen. Für die Software bedeutet das, dass jederzeit eine Änderung in der Art und Weise der Verarbeitung auftreten kann. [25]

Das in Artikel 20 beschriebene Recht auf Datenübertragbarkeit ermöglicht es, betroffenen Personen mit all ihren Daten zu einem anderen Dienst zu wechseln. Das kann durch die Person selbst erfolgen, es müssen also ihr alle Daten zukommen.

Eine andere Möglichkeit ist die direkte Übermittlung der Daten an den anderen Dienst auf Veranlassung der Person. Der Artikel selbst spricht von einem "maschinenlesbaren Format", es muss der betreffenden Software also möglich sein, sämtliche Daten einer bestimmten Person zu sammeln und in ein gängiges Format zu übersetzten. Alternativ wäre es auch möglich mit einer anderen Software die Daten auszulesen, sofern eine gewisse Architektur vorhanden ist, doch auch das setzt ein die Möglichkeit voraus alle Daten zu sammeln. Artikel 22 gibt betroffenen Personen in bestimmten Fällen die Möglichkeit eine nicht automatische Entscheidungsfindung zu verlangen. Sollte die Software also, für den Artikel zutreffende Entscheidungen treffen, muss es die Möglichkeit eines manuellen Overrides gegeben, also das ein menschlicher Entscheider der Software die Entscheidung abnimmt, gegebenenfalls muss die Software auch mit dieser Entscheidung weiterarbeiten. Im letzteren Fall ist es notwendig, dass es eine Möglichkeit gibt der Software die Entscheidung mitzuteilen, um den Regelbetrieb aufrecht zu erhalten. [25]

Artikel 25 hat nach einem vorläufigen Kommentar des Europäischen Datenschutzbeauftragten [9], mehrere wichtige Dimensionen. So sollen wenigstens teilweise automatisierte datenverarbeitende IT Systeme als Ergebnis eines Design Projektes angesehen werden [9, p. 11]. Die Software muss also schon in ihrer Konzeption angemessene Sicherheit für Daten bieten. Außerdem ist ein angemessenes Risikomanagement wichtig, bei dem Maßnahmen gewählt und implementiert werden sollen, um die schützenswerten Daten von Einzelpersonen ausreichend zu schützen [9, p. 11]. Von besonderer Bedeutung für die betroffene Software, ist die tatsächliche Integration der für wichtig befundenen Maßnahmen.

Abschließend muss Artikel 32 berücksichtigt werden. Er schreibt vor, dass unter Berücksichtigung des Stands der Technik und weiteren Faktoren, geeignete Maßnahmen getroffen werden, um ein dem Risiko angemessenes Schutzniveau zu gewährleisten. Dieser Schutz kann und soll unter anderem durch Pseudonymisierung und Verschlüsselung personenbezogener Daten erreicht werden. Die Fähigkeit, die Vertraulichkeit, Integrität, Verfügbarkeit und Belastbarkeit der Systeme und Dienste im Zusammenhang mit der Verarbeitung auf Dauer sicherzustellen ist ebenso wichtig. Außerdem soll ein Verfahren zur regelmäßigen Überprüfung, Bewertung und Evaluierung der Wirksamkeit der technischen und organisatorischen Maßnahmen zur Gewährleistung der Sicherheit der Verarbeitung eingeführt werden. [24]

Kurz zusammengefasst ergeben sich folgende Anforderungen:

- Daten müssen vor unbefugtem Zugriff geschützt werden
- Daten müssen veränderbar sein
- Daten müssen löschbar sein
- Die Verarbeitung von Daten muss einschränkbar oder sogar stopbar sein
- Es muss möglich sein, bestimmte Entscheidungen unabhängig von der Software zu treffen
- Die Daten müssen vollständig in einem maschinenlesbaren Format ausgegeben werden können
- Es muss ein durchdachtes und angemessenes Konzept für die Sicherheit der Datenverwaltung existieren
- Es soll mit möglichst minimalen Datensätzen gearbeitet werden, bzw. die Verwendung von möglichst wenigen Daten muss für den Nutzer voreingestellt sein

3 Zielstellung des Programms

Basierend auf den herausgearbeiteten relevanten Punkten der DSGVO und den Blockchain Grundlagen werden potentielle Konfliktpunkte besprochen, die im letzten Teil der Arbeit bearbeitet werden. Die einzelnen Anforderungen aus Kapitel 2.2.3 werden dahingehend analysiert, ob sie in direkter Relevanz zu Blockchainanwendungen stehen.

Anforderung: Daten müssen vor unbefugtem Zugriff geschützt werden
Konflikt: Blockchain ist allgemein zugangsunbeschränkt und öffentlich
Beschreibung: Die Öffentlichkeit von Blockchains bedeutet, dass jeder alle Daten der Chain aus dem Netzwerk anfordern kann. Zusätzlich werden Teilnehmer des Netzwerkes nicht beschränkt, das heißt jeder kann grundsätzlich Transaktionen an jede Adresse senden.

Anforderung: Daten müssen veränderbar sein
Konflikt: Blockstruktur ist unveränderbar
Beschreibung: Durch die Hashverkettung der einzelnen Blöcke können vorangegangene Blöcke nicht verändert werden, ein Ersetzen von Daten ist somit nicht möglich.

Anforderung: Daten müssen löschbar sein
Konflikt: Blockstruktur ist unveränderbar
Beschreibung: Durch die Hashverkettung der einzelnen Blöcke können vorangegangene Blöcke nicht verändert werden, ein direktes Löschen von Daten ist somit nicht möglich.

Anforderung: Es muss möglich sein, bestimmte Entscheidungen unabhängig von der Software zu treffen
Konflikt: Normales Softwareproblem
Beschreibung: Entscheidungsprozesse aus Software auszulagern ist ein normales Problem in der Softwareentwicklung, hierbei geht es um die Architektur. Es muss möglich sein, über bestimmte Funktionen extern getroffene Entscheidungen der Software zu übergeben. Eine Implementierung solcher Funktionen in Smart Contracts ist ohne besondere Probleme möglich.

Anforderung: Die Daten müssen vollständig in einem maschinenlesbaren Format ausgegeben werden können
Konflikt: Normales Softwareproblem
Beschreibung: Das Ausgeben von Daten aus einer Datenbank oder ähnlichen ist auch ein normales Software Problem. Wichtig bei der Umsetzung ist es einen Überblick zu haben, wo welche Daten gespeichert sind.

Anforderung: Die Verarbeitung von Daten muss einschränkbar oder sogar stoppbar sein
Konflikt: Normales Softwareproblem
Beschreibung: Das Stoppen der Verarbeitung von Daten ist wieder ein klassisches Softwareproblem. Solange Kontrolle über den Verarbeitungsprozess gewahrt wird, ist das Stoppen desselben möglich. Blockchain bietet allerdings die Möglichkeit unstoppbare Applikationen zu schreiben, hierbei werden einfach Kontrollmechanismen, wie eine Funktion zum stoppen des Contracts weggelassen. Das Implementieren solcher Funktionen ist allerdings keine besonderer Hürde.

Anforderung: Es muss ein durchdachtes und angemessenes Konzept für die Sicherheit der Verwaltung der Daten existieren
Konflikt: Externes Problem
Beschreibung: Das Sicherheitskonzept muss vor dem eigentlichen Entwickeln der Software erstellt werden, beispielsweise beschrieben in [13]. Außerdem muss es an die Blockchaingegebenheiten angepasst sein.

Anforderung: Es soll mit möglichst minimalen Datensätzen gearbeitet werden, bzw. die Verwendung von möglichst wenigen Daten muss für den Nutzer voreingestellt sein
Konflikt: Normales Softwareproblem
Beschreibung: Das Herausarbeiten, welche Daten tatsächlich notwendig sind, und so zu den minimalen Datenmengen, die gebraucht werden, anzukommen, ist ein Problem das auch von nicht Blockchainsoftware bewältigt werden muss. Es ist demnach ein normales Softwareproblem. Gute und verständliche Einstellungsmöglichkeiten für Nutzer ist eine übliche User-Experience-Design Aufgabe.

Anforderungen, die besondere Blockchain relevante Gründe haben, werden auf potentielle Lösungen als Ziel der Arbeit in Kapitel 4 untersucht. Anforderungen die unabhängig davon auf jede Software, die personenbezogene Daten verarbeitet, zutreffen, werden nicht extra berücksichtigt, da hier bereits Lösungen bestehen, welche mehr oder weniger direkt für Smart Contracts in Ethereum übernommen werden können. Denn bis auf einige bestimmte Eigenschaften von Smart Contracts, sind sie im Grunde genommen nur gewöhnliche Programme, wie sie auch außerhalb von Blockchains bestehen.

4 Umsetzung

Wie im vorherigen Kapitel betrachtet, gibt es insgesamt drei Anforderungen, die die DSGVO stellt, die bei Blockchain Anwendungen eine besondere Herausforderung darstellen. Das sind:

1. Schutz von Daten in einem öffentlichen System

2. Veränderung von Daten in einem nicht überschreibaren System

3. Dauerhaftes Löschen von Daten in einer nicht veränderbaren Datenstruktur

Im folgenden Ansatz wird ein bestimmtes Szenario betrachtet, bei dem es um die Verwaltung von personenbezogenen Daten durch ein Unternehmen geht. Das Unternehmen handelt im Auftrag von Einzelpersonen und verwaltet für sie Dinge von Wert, zum Beispiel Token auf der Blockchain. Das Unternehmen ist in der Verantwortung gegenüber einer Aufsichtsbehörde oder den einzelnen Kunden nachzuweisen, über welchen Wert sie aktuell verfügen, bzw. wie sich dieser Wert über die Zeit entwickelt hat, ohne dass ein Dritter in der Lage ist herauszufinden wer die Personen sind. Ohne diese, oder ähnliche, Garantien macht die Verwendung einer Blockchain wenig Sinn, da kein externer in der Lage ist, das Handeln des Unternehmens nachzuvollziehen. Es wäre also einfacher für das Unternehmen eine eigene interne Datenbank zu führen. Die Blockchain würde keinerlei Mehrwert darstellen, da bei nicht nachzuvollziehenden Vorgängen das Unternehmen manipulieren könnte. Bei der Verwaltung von Daten durch Unternehmen stellt die Blockchain hauptsächlich ein Mittel für Transparenz und Vertrauenswürdigkeit dar. Das es zu verhindern gilt, dass Dritte die Daten einsehen können, ergibt sich schon aus den Anforderungen der DSGVO.

Geschäftsprozesse sollen nicht abgebildet werden, um die Vorgänge der Verwaltung von Daten in Verbindung mit einer Blockchain klar hervorzuheben. Es wird sich auf einen einfachen, aber definitiv personenbezogenen, Datensatz beschränkt. Die Vorgänge und Techniken sollten sich leicht auf mehr, oder andere, Daten erweitern lassen. Der Datensatz enthält:

- Name
- Vorname
- E-Mail Adresse
- Telefonnummer

Als Zielplattform wird eine Ethereum Blockchain gewählt, da sie die größte Blockchain mit Turing-vollständigen Smart Contracts, die in High-Level Sprachen geschrieben werden können, ist. Dadurch können die anderen nicht Blockchain spezifischen Anforderungen der DSGVO durch herkömmliche Software Paradigmen fast problemlos gelöst werden.

4.1 Grundlegender theoretischer Ansatz

Durch die strikte Formulierung des "Rechts auf Vergessen" in Artikel 17 das Daten unverzüglich gelöscht werden, und der Verantwortliche verpflichtet ist, personenbezogene Daten unverzüglich zu löschen [11] bleibt kein Spielraum für die Umsetzung. Im Gegensatz zu beispielsweise beim verlangten Schutz der Daten, der wenigstens angemessen sein muss [11]. Dass heißt, es ist ein tatsächliches, vollständiges Löschen der Daten durchzuführen, eine Verschlüsselung der Daten, mit anschließender Löschung des Schlüssels wird vermutlich nicht ausreichen. Denn dabei sind die Daten immer noch vorhanden, und beispielsweise durch ein Brechen des Verschlüsselungsalgorithmus wiederherstellbar. Das ist bei einem Blockchain Netzwerk deshalb ein Problem, da es die Daten auf viele unbekannte Rechner überträgt. Diese könnten gegebenenfalls die verschlüsselten Daten speichern, bis in Zukunft eine Methode zum Brechen der Verschlüsselung gefunden wird. Folglich garantiert diese Methode keine sichere Einhaltung der DSGVO.

Ein alternativer Ansatz ist, keine direkt personenbezogene Daten auf der Blockchain zu speichern, sondern Referenzen auf lokal vorgehaltene Daten. Hierbei wird eine eindeutige Referenz in der Blockchain gespeichert und gilt stellvertretend für die Einheit, der eine einzelne Person betreffenden Daten. Das im Kapitel 4 beschriebene Szenario eignet sich für eine solche Lösung. Da das Unternehmen für einzelne Personen auf der Blockchain agiert, zum Beispiel als Verwalter von Investments, die Personen selbst aber nicht für, beispielsweise einen Handel, wichtig sind.

Bei dieser Methode wird ein eindeutiger Hash der personenbezogenen Daten erstellt und in der Blockchain als Identität abgelegt. Dadurch kann sowohl das Problem des Löschens, als auch das Problem des Aktualisierens umgangen werden. Da die tatsächlichen Daten weiterhin nur intern und off-chain gespeichert werden, können sie, wie bei gewöhnlichen Anwendungen, verändert werden. Für die Sicherheit der Daten können bereits bekannte IT-Techniken verwendet werden. Lediglich der Hash in der Blockchain bedarf besonderer Beachtung, um eine Verbindung von Personen und ihrem auf der Blockchain gespeicherten Wert (zb. Token) zu erschweren.

4.2 Erweiterung Secret

Ohne zusätzliche Maßnahmen ist es vergleichsweise leicht Hashe für Daten zu finden, die eine bestimmte, bekannte Form haben. Beispielsweise gibt es nur ein relativ kleines Feld an möglichen Namen für Personen, wenn jede lebende Person auf der Erde einen einzigartigen Namen hat, gibt es insgesamt 7,6 Milliarden [12] mögliche Namen, die sich hinter einem Hash verbergen könnten. Aktuelle Grafikkarten können etwa 600 Millionen double-SHA-256 pro Sekunde errechnen [1], das Finden des richtigen Namens würde somit nur etwa 12 Sekunden dauern.

Um ein einfaches Erraten der Daten zu verhindern, muss ein Secret mit gehasht werden, es sollte aus einer guten Zufallsquelle stammen. Durch das Secret ist für Externe nicht mehr ersichtlich, welcher Hash sich auf welche Person bezieht, da nun nicht mehr bekannte, erratbare Werte probiert werden können, sondern lange,

zufällige Zeichenfolgen gefunden werden müssen. Wichtig hierbei ist, dass das Secret geheim gehalten wird. Bei einem 64bit Secret gibt es 2^{64} mögliche Zusammensetzungen. Wenn die Komplexität des Namens vernachlässigt wird, muss also maximal 2^{64} gehasht werden, um sicher das korrekte Secret zu finden. Die Anzahl der Namen war etwa 2^{33}, das Finden des Hashes dauert also maximal 2^{31} mal länger.

Gleichzeitig ergibt sich dadurch die Notwendigkeit für eine Methode, mit der das Unternehmen der betreffenden Person beweisen kann, dass ein bestimmter Hash tatsächlich sie repräsentiert. Denn jetzt kann die Person nicht mehr einfach ihre Daten selbst hashen und dann diesen Hash in der Blockchain kontrollieren. Es muss also ein Weg gefunden werden, mit dem das Unternehmen beweisen kann, das ein bestimmter Hash eine bestimmte Person ist. Sonst muss der Kunde dem Unternehmen voll vertrauen, die Blockchain würde also keinen Mehrwert liefern.

Die einfachste Möglichkeit zu beweisen, dass ein bestimmter Hash eine bestimmte Person repräsentiert ist, der betreffenden Partei das Secret auszuhändigen. Wenn die Partei die relevanten, gehashten Daten kennt, kann sie den Endhash rekonstruieren. Ein nachträgliches Fälschen des Endhashes durch das verwaltende Unternehmen ist bei Verwendung eines Kollisionsresistenten Algorithmus, beschrieben in Kapitel 2.1.1, nicht möglich.

Es ist allerdings zu beachten, dass das Secret auf einem sichern Kanal übermittelt wird, da sonst Dritte Informationen über die betreffende Person erlangen können. Mit dem bekannten Secret vereinfacht sich die Arbeit für den Angreifer wieder auf die zu Beginn berechneten 12 Sekunden mit Standard Hardware.

4.3 Erweiterung Merkle-Tree

Dieses einfache Erraten bei bekanntem Salt kann durch die Verwendung eines Merkle-Trees [21] verhindert werden. Der Tree wird aus Leafs aufgebaut, die jeweils der Hash aus einer der personenbezogenen Daten, zum Beispiel der Name und einem Secret sind. Die Merkle-Root wird als Referenz in der Blockchain gespeichert. Ein weiterer Vorteil ist, dass die zu beweisenden Daten selektiv bestimmt werden können. Es

kann beispielsweise nur der Name bewiesen werden ohne die Telefonnummer kennen zu müssen.

Ein Beweis für einen der Datensätze würde wie folgt ablaufen. Das Unternehmen offenbart dem Kunden das Secret für den Datensatz, das zur gleichen, höher liegenden Node gehörende Leaf und alle weiteren neben liegenden Nodes auf dem direkten Pfad zur Root. Der Kunde kann nun den ihm bekannten Datensatz mit dem Secret Hashen und den Tree bis zur Root vervollständigen. Am Ende sollte er die gleiche Root erhalten, die auch in der Blockchain gespeichert ist.

Bei einem Verlust des Secret und aller Node-Hashe an einen böswilligen Akteur, kann dieser nur deutlich schwerer Informationen über den Datensatz gewinnen, da er nicht den direkt zu erratenden Hash kennt. Er muss für den gesamten Pfad korrekte Hashe finden. Bei einem Merkle-Tree mit drei Ebenen verdreifacht sich sein Rechenaufwand, da er immer drei weitere Male Hashen muss, bis er weiß ob sein Ursprungswert der Richtige ist.

Noch aufwändiger wird es für den Angreifer, die nicht bewiesenen Datensätze zu erraten, sofern für alle ein einzigartiges Secret verwendet wird. Wird kein einzigartiges Secret benutzt, ist es für den Angreifer unter Umständen sogar einfacher nicht das direkt bewiesene Leaf anzugreifen, da er vom Nachbar Leaf bereits den direkten Hash kennt. Somit muss er nicht mehr den gesamten Pfad durch den Tree durchwandern.

4.4 Praktische Umsetzung

Für die praktische Umsetzung wurde mit dem Javascript Framework vue.js [36] und dem CSS Framework bulma [30] eine minimale Anwendung geschrieben, die als Benutzeroberfläche dient und sich in modernen Browsern benutzen lässt. Das Programm kann sich mittels der web3.js Bibliothek [19] mit einer Ethereum Blockchain verbinden. Zu Demonstrationszwecken ist ein Standard Account eingerichtet, der bereits das, für Transaktionen nötige, Ether besitzt. In diesem Modus wird der Infura Dienst [2] genutzt, um die Verbindung zur Blockchain herzustellen. Außerdem

lässt sich die Anwendung auch mit der Metamask Browsererweiterung [3] und einem eignen Account, den Besitz von Test-Ether vorausgesetzt, benutzen.

Auf Blockchainseite wurde ein einfacher Smart Contract geschrieben, der, ähnlich dem bekannten ERC-20 Standard [16], über verschiedene Kontostände, für verschiedene Accounts verfügt. Accounts sind in diesem Fall die Hashreferenzen der persönlichen Daten der Kunden des Unternehmens. Der Contract verfügt über verschiedene Funktionen zum Hinzufügen, Ändern und Löschen von Kunden. Löschen bedeutet in diesem Fall eine Invalidierung des Eintrags der Referenz, der Hash selbst kann effektiv nicht gelöscht werden, ist für sich alleine gestellt aber keine personenbezogene Information. Außerdem gibt es noch eine einfache Transfer Funktion um das agieren im Namen der Kunden zu demonstrieren.

Der Zugriff auf die Funktionen wird mittels modifier, die digitale Signaturen überprüfen, angemessen beschränkt. So kann ein Kunde nur von dem Unternehmen verändert, gelöscht oder in seinem Namen Wert verschickt werden, das ihn auch als Kunden im Smart Contract registriert hat. Um Inkonsistenzen und exploits zu verhindern, werden mit Hilfe von mehreren require außerdem noch einige Werte überprüft. Beispielsweise kann ein Hash nicht ein zweites Mal registriert werden, solange er aktiv ist. Andernfalls könnte ein bösartiger Akteur die Transaktionsparameter einer Kundenerstellung auslesen, eine eigene Transaktion mit den gleichen Werten durchführen und so die Kontrolle über die Referenz und damit den Wert oder die Token erlangen. Außerdem wird bei transfers sichergestellt, dass das Konto genügend gedeckt ist.

Auf der Blockchain wird der Contract durch das embark Framework [20] geschaffen, das automatisch eine JSON-Datei erstellt in der die für das Ansprechen des Contracts wichtige ABI und seine Adresse liegen. Diese Datei wird direkt in die Javascript Anwendung eingebunden. Anschließend wird mittels web3.js ein Contract Objekt erstellt mit dem interagiert werden kann. Die lokale Datenbank des Unternehmens wurde mittels der LocalStorage des Browsers umgesetzt. Persistenz zwischen Seitenaktualisierungen und die Speicherkapazität sind für das Beispiel ausreichende Eigenschaften. Mittels verschiedener Funktionen zum Interagieren mit der Blockchain durch das Contract Objekt, der LocalStorage und der Benutzeroberfläche wurden die einzelnen Komponenten zusammengeführt.

Für die Hashgenerierung wurde der, auch von Ethereum verwendete, Keccak-256 Algorithmus verwendet. Das für den Schutz der Daten nötige Secret wird mittels der web3.js Bibliothek erstellt und neben den personenbezogenen Daten in der LocalStorage des Browsers aufbewahrt. Der Beweis über die Kenntnis der Daten kann außerdem in der Benutzeroberfläche erstellt werden. Unter einem separaten Tab, der nicht auf die LocalStorage zugreift, kann der Beweis überprüft werden.

5 Zusammenfassung und Ausblick

Im Verlauf der Arbeit wurden zunächst einige essentielle Grundlagen über die Zusammensetzung und Funktionsweise von Blockchains gesammelt. Anschließend wurden die 99 Artikel der DSGVO zunächst auf Anwenderrelevanz, dann auf Softwarerelevanz und abschließend auf besondere Relevanz in der Blockchainthematik untersucht. Mit den drei resultierenden Anforderungen wurde sich auseinandergesetzt, um eine theoretischen Methode zu finden, um diese Anforderungen zu erfüllen. Durch das Umgehen der Konfliktpunkte konnte eine grundlegende Lösung gefunden werden, die anschließend durch weitere Techniken verbessert wurde. Abschließend wurde die theoretische Lösung als Beispielprogramm umgesetzt, das alle minimalen Aufgaben und Anforderungen erfüllen kann.

Der in dieser Arbeit beschriebene Lösungsansatz ist nicht nur im Hinblick auch die DSGVO sinnvoll. Auch wenn es nicht darum geht Gesetze einzuhalten, macht Datenschutz Sinn. Die Sicherheit der Blockchain nutzen zu können, ohne private Daten der breiten Öffentlichkeit mitzuteilen, ist sicherlich wünschenswert.

Weiterführend würde ein Ausbau des Ansatzes Sinn machen. Dass das Secret verraten werden muss, um einen Beweis zu liefern, ist definitiv noch eine Schwachstelle, die sich alleine dadurch ergibt, dass danach jeder der in Besitz des Secrets gelangt, vorgeben könnte der Verwalter des Kontos zu sein. Das ließe sich durch eine Interaktion mit der Blockchain vermeiden. Hier müsste jedes Mal der Verwalter eine bestimmte durch Signaturüberprüfung gesicherte Funktion aufrufen, um zu signalisieren das er wirklich der Verwalter ist. Das Problem der Vereinfachung des Hashfindes durch einen Angreifer, wie in Kapitel 4.2 beschrieben, bleibt aber weiter bestehen. Potentiell könnte durch zero-knowledge-proofs ein Beweis erstellt werden, mit dem bewiesen wird, dass ein bestimmter Input zu der in der Blockchain hinterlegten Merkle-Root führt. So könnte das Secret weiterhin geheim bleiben.

Bibliography

[1] Non-specialized hardware comparison. URL https://en.bitcoin.it/wiki/Non-specialized_hardware_comparison.

[2] Infura. URL https://infura.io.

[3] Metamask. URL https://metamask.io.

[4] Transactions historical chart. URL https://bitinfocharts.com/index_v.html.

[5] Scott A. Vanstone Alfred J. Menezes, Paul C. van Oorschot. Handbook of applied cryptography. chapter 9 Hash Functions. 1996.

[6] Scott A. Vanstone Alfred J. Menezes, Paul C. van Oorschot. Handbook of applied cryptography. chapter 11 Digital Signatures. 1996.

[7] Vitalik Buterin. Proof of stake faqs, . URL https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQs.

[8] Vitalik Buterin. Light client protocol, . URL https://github.com/ethereum/wiki/wiki/Light-client-protocol.

[9] Giovanni Buttarelli. Preliminary opinion on privacy by design. 2018.

[10] coinmarketcap.com. Percentage of total market capitalization. URL https://coinmarketcap.com/charts/#dominance-percentage.

[11] Council of European Union and the Parliament. Council and parliament regulation (EU) no 2016/679, 2016. https://eur-lex.europa.eu/legal-content/en/TXT/?uri=CELEX:32016R0679.

[12] U.N. department of economic and social affairs. World population prospects: The 2017 revision. URL https://www.un.org/development/desa/publications/world-population-prospects-the-2017-revision.html.

[13] Konferenz der unabhängigen Datenschutzbehörden des Bundes und der Länder. Das standard- datenschutzmodell. 2018.

[14] eth. How is the address of an ethereum contract computed? URL https://ethereum.stackexchange.com/questions/760/how-is-the-address-of-an-ethereum-contract-computed.

[15] etherscan.io. Ethereum transaction chart. URL https://etherscan.io/chart/tx.

[16] Vitalik Buterin Fabian Vogelsteller. Erc-20 token standard. URL https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md.

[17] handelsblatt.com. Plus sieben prozent – bitcoin auf erholungskurs. URL https://www.handelsblatt.com/finanzen/maerkte/devisen-rohstoffe/newsblog-blockchain-bitcoin-ripple-plus-sieben-prozent-bitcoin-auf-erholungskurs/20913326.html.

[18] Aggelos Kiayias Juan A. Garay. The bitcoin backbone protocol: Analysis and applications. 2017.

[19] Fabian Vogelsteller Marek Kotewicz. Ethereum javascript api. URL https://github.com/ethereum/web3.js.

[20] Iuri Matias. embark. URL https://embark.status.im/.

[21] Ralph C. Merkle. A certified digital signature. 1979.

[22] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[23] Axel von dem Bussche Paul Voigt. Eu-datenschutzgrundverordnung praktiker-handbuch. chapter 1. Einleitung und Checkliste. 2018.

[24] Axel von dem Bussche Paul Voigt. Eu-datenschutzgrundverordnung praktiker-handbuch. chapter 3. Anforderungen an die Datenschutzorganisation. 2018.

[25] Axel von dem Bussche Paul Voigt. Eu-datenschutzgrundverordnung praktikerhandbuch. chapter 5. Rechte der betroffenen Personen. 2018.

[26] Axel von dem Bussche Paul Voigt. Eu-datenschutzgrundverordnung praktikerhandbuch. chapter 7. Rechtsdurchsetzung und Sanktionen. 2018.

[27] Dr. Matthias Schmidl. Verordnung (eu) 2016/679 – Datenschutz-Grundverordnung Leitfaden. 2018.

[28] Robert Stadler Seàn Gauld, Franz von Ancoina. The burst dymaxion. 2017.

[29] Péter Szilágyi. Clique poa protocol & rinkeby poa testnet. URL `https://github.com/ethereum/EIPs/issues/225`.

[30] Jeremy Thomas. Bulma. URL `https://bulma.io`.

[31] Virgil Griffith Vitalik Buterin. Casper the friendly finality gadget. 2018.

[32] Dr. Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. chapter 4. Blocks, State and Transactions. Byzantium version fadb37b edition, 2018.

[33] Dr. Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. chapter 10. Blocktree to Blockchain. Byzantium version fadb37b edition, 2018.

[34] Dr. Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. chapter 9. Execution Model. Byzantium version fadb37b edition, 2018.

[35] Dr. Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. chapter Appendix G. Fee Schedule. Byzantium version fadb37b edition, 2018.

[36] Evan You. The progressive javascript framework. URL `https://vuejs.org/`.

Anlagen

Das unter Kapitel 4 beschriebene Programm befindet sich auf der beigefügten CD-ROM. Im Ordner Quellcode ist das Projekt selbst zu finden. Der Ordner Anwendung enthält das betriebsbereite Programm. Beide Ordner enthalten readme Dateien, die die Anwendung und den Umgang mit den Dateien beschreiben. Die Dokumentation kann dem Quellcode entnommen werden.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Ort, Datum	Unterschrift