Information-Security

Kryptografisches Zugriffskontrollsystem für Blockchains

Diplomarbeit

zur Erlangung des akademischen Grades

Diplom-Ingenieur

eingereicht von

Ing. Robert Prinz, BSc

is171826

im Rahmen des

Studienganges Information-Security an der Fachhochschule St. Pölten

Betreuer: Prof. FH Univ.-Doz. D.I. Dr. Ernst Piller

St. Pölten, 16. Juni 2019

(Unterschrift Verfasser/in)

(Unterschrift Betreuer/in)

Fachhochschule St. Pölten GmbH, Matthias Corvinus-Straße 15, A-3100 St. Pölten,T: +43 (2742) 313 228, F: +43 (2742) 313 228-339, E:office@fhstp.ac.at, I:www.fhstp.ac.at

Ehrenwörtliche Erklärung

Hiermit erkläre ich, dass

- ich die vorliegende Diplomarbeit selbständig verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und mich sonst keiner unerlaubten Hilfe bedient habe.
- ich dieses Diplom-Arbeitsthema bisher weder im Inland noch im Ausland einer Begutachterin/ einem Begutachter zur Beurteilung oder in irgendeiner Form als Prüfungsarbeit vorgelegt habe.
- diese Arbeit mit der von der Begutachterin/vom Begutachter beurteilten Arbeit übereinstimmt.

Der Studierende/Absolvent räumt der Fachhochschule St. Pölten das Recht ein, die Diplomarbeit für Lehre- und Forschungstätigkeiten zu verwenden und damit zu werben (z.B. bei der Projektvernissage, in Publikationen, auf der Homepage), wobei der Absolvent als Urheber zu nennen ist. Jegliche kommerzielle Verwertung/Nutzung bedarf einer weiteren Vereinbarung zwischen dem Studierenden/Absolventen und der Fachhochschule St. Pölten.

Ort, Datum *Unterschrift*

Gender Erklärung

Aus Gründen der besseren Lesbarkeit wird in dieser Diplomarbeit auf die gleichzeitige Verwendung männlicher und weiblicher Formulierungen verzichtet. Der Autor möchte jedoch ausdrücklich darauf hinweisen, dass sämtliche Personenbezeichnungen für beide Geschlechter zu verstehen sind.

Zusammenfassung

Im Jahr 2019 setzt sich der seit einigen Jahren beobachtete Trend, Blockchain-Technologie einzusetzen, fort. Mitverantwortlich für diesen Höhenflug ist die Anwendung der Blockchain bei Kryptwährungen. Dabei darf die Sicherheit und Integrität der sensiblen Daten in einer Blockchain nicht außer Acht gelassen werden.
Wissenschaftliche Studien zeigen aber, dass das Vertrauen und die Authentizität der Daten nicht immer gegeben sind. Des Weiteren kann durch die Unveränderbarkeit der Blockchain die Erfüllung der Europäischen Datenschutz-Grundverordnung (DSGVO) in Bezug auf „Recht auf Löschung“ nicht eingehalten werden. Auf Grund der enormen Auswirkungen, die durch eine Kompromittierung bei einer Blockchain entstehen, beschäftigt sich diese Arbeit mit der Implementierung von kryptografischen Zugriffskontrollsystemen für Blockchains. Dabei stellt sich die Frage, wie eine Blockchain Daten sicher abspeichern kann, sodass sie von nicht berechtigten Zugriffen geschützt ist. Die Einhaltung der DSGVO ist dabei essenziell. Außerdem ist zu berücksichtigen, wie die *Blockchain for Data Security*, die Blockchain der Zukunft, aussehen muss, damit ein sicherer Einsatz möglich ist. Aus diesem Grund werden bereits bestehende kryptografische Verfahren, wie *Shamir Secret Sharing* und die auf dem *Rabin-Kryptosystems* basierende Methode, *Cryptographic Access Control Solution for Smart Phones* (CASSP), für den Einsatz in der Blockchain überprüft.

Durch die Untersuchung der beschriebenen Verfahren werden neue *Service Nodes* definiert, welche die Aufgabe der „Miner“ von Blöcken übernehmen. Sie erfüllen und setzen die Anforderungen der DSGVO um. Zusätzlich werden zwei neue Konsensverfahren, *proof of cryptography* und *proof of signature*, entwickelt. Somit kann gewährleistet werden, dass nur berechtigte Teilnehmer Blöcke in die Blockchain hinzufügen können. Mit den ausgearbeiteten Verfahren, welche auf kryptografische Zugriffskontrollsysteme aufbauen, können Lesezugriffe auf Blöcke gesteuert werden.

Keywords: Blockchain, Blockchain for Data Security, Kryptografie, Zugriffskontrolle, kryptografisches Zugriffskontrollsystem, Datenschutz-Grundverordnung, DSGVO, *Trusted Computing Platform*, CASSP, *Rabin-Kryptosystem*, *Shamir Secret Sharing*

Abstract

It has been observed that the trends in the year 2019 and in the past few years of using Blockchain technology shall continue. One of the main reasons for high trend and increase in adoption of this technology is the application of Blockchain in crypto currency. Therefore, the security and integrity of the sensitive data in Blockchain application cannot be taken for granted, nor must it be ignored.

Scientific studies show that the trust and authenticity of the data is not always given. Furthermore, the immutability of the Blockchain means that compliance with the General Data Protection Regulation (GDPR) with regard to the 'right to be forgotten' is near impossible. Because of to its enormous impact on all aspects of Blockchain technology, this work addresses the complex issue of Blockchain security compromise. It also deals with one of the ways to mitigate the risks associated with Blockchain technology which is cryptographic access controls. This raises the question of how a Blockchain can store data securely so that it is protected from unauthorized access. Compliance with the GDPR is essential. In addition, it has to be taken into account how the future *Blockchain for Data Security* would look like so that it can be used safely. For this reason, the already existing cryptographic methods such as *Cryptographic Access Control Solution for Smart Phones* - CASSP which is based on the *Rabin cryptosystem* and *Shamir Secret Sharing*, would be evaluated and considered for use in securing Blockchain.

By examining the methods described above, new *service nodes* have been defined which take on the task of 'mining' of blocks. They fulfill and implement the requirements of the GDPR. In addition, the two new consensus procedures, *proof of cryptography* and *proof of signature*, are all being developed. All of these ensure that only authorized users can add blocks to the Blockchain. With the elaborated methods which are based on cryptographic access control systems, additional read access to blocks can be controlled.

Keywords: Blockchain, Blockchain for Data Security, cryptography, access control, cryptographic access control, CAC, *GDPR*, *Trusted Computing Platform*, CASSP, *Rabin cryptosystem*, *Shamir Secret Sharing*

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

Die wachsende Datenmenge und Handhabung auf dezentraler Ebene in IT-Systemen benötigen neu geschaffene Speichermöglichkeiten. Um diese bewältigen zu können, kommt der *Blockchain* als Speichermedium eine sehr große Bedeutung zu. Die Blockchain, namensgebend von verketteten Blöcken [1, S.11], wird bei digitalen Währungen, wie Bitcoin, eingesetzt. Das Konzept dieses Datenbank-Management-Systems wurde von dem japanischen Entwickler Satoshi Nakamoto [2] im Jahre 2008 zum ersten Mal erwähnt. Mit dieser Entwicklung lassen sich große Mengen an Daten miteinander verketten und dezentral ablegen. Die Blockchain lässt sich nicht nur in der Finanzbranche einsetzen, sondern auch in jedem anderem Ökosystem, wie im Risikomanagementbereich, bei *internet of things* (IoT) sowie in öffentlichen und sozialen Bereichen. [3, S.352] In dieser Diplomarbeit werden die Möglichkeiten der Absicherung einer Blockchain beleuchtet.
Der Autor hat zum Anlass genommen, die Anwendbarkeit der kryptografischen Zugriffskontrolle bei einer *Blockchain* genauer zu erforschen, die derzeit noch nicht vollständig behandelt wurde. [4, S.3] Dabei wird der Frage nachgegangen, wie Lesezugriff und Schreibzugriff auf eine Blockchain mit herkömmlichen, bereits etablierten kryptografischen Zugriffskontrollen, sicher gemacht und damit unerwünschte Zugriffe oder Veränderungen der Daten vermieden werden können. Zusätzlich wird die neue Regulation 2016/679 der europäische Union (EU), die EU-Datenschutz-Grundverordnung (DSGVO), in dieser Arbeit auch *general data protection regulation* (GDPR) genannt, nicht außer Acht gelassen. Diese gibt jedem das Recht auf Schutz von personenbezogenen Daten sowie Löschung der eigenen Daten. [5,] [6,] Somit muss in einer Blockchain nicht nur der Lese- und Schreibzugriff, sondern auch das Löschen der Daten nach DSGVO berücksichtigt werden.

1.1. Problemstellung

In den meisten Fällen wird der Begriff Blockchain mit der Kryptowährung Bitcoin in Verbindung gebracht. Zwischenzeitlich zieht das Interesse immer größere Kreise. In im Jahr 2016 veröffentlichten Gartner *Hype Cycle for Emerging Technologies 2016* wurde die Blockchain als eine der zukunftsträch-

tigsten Technologien der aktuellen Zeit gewertet. Allerdings werden das Potenzial und der produktive Einsatz erst in 5-10 Jahren gesehen. [7, S.2] Dies ist darauf zurückzuführen, dass die Technologie und die Sicherheit der Daten noch nicht vollständig erforscht sind [8, S.978]. Die Blockchain wird trotz ihrer Dezentralität als fälschungssicher eingestuft. Der Wegfall eines zentralen Betreibers macht sie gegen viele klassische Angriffsmethoden robust.

Trotzdem gibt es andere Ansätze von Angriffsvektoren. Das Vertrauen, ob die Blockchain wirklich sicher ist, hängt von den eingesetzten kryptografischen Verfahren ab. [9, S.25-26] Da jeder Benutzer seine Blöcke selbstständig in die Blockchain schreibt, muss er auch selbst darauf achten, dass dies durch einen sicheren Mechanismus geschieht. Um die Transaktion zu signieren, wird der eigene private Schlüssel für die Kryptografie verwendet. [9, S.26] Erhält ein Angreifer Zugriff auf diesen Schlüssel, kann er damit ebenfalls Transaktionen durchführen und somit die Echtheit fälschen. Um das zu verhindern, muss jeder Benutzer seinen privaten Schlüssel sicher aufbewahren. Die sicherste Aufbewahrung der privaten Schlüssel, wird durch ein Hardware-Security-Modul gewährleistet. [10]

Des Weiteren gibt es den sogenannten 51%-Angriff. Dieser kann dann auftreten, wenn ein Angreifer statistisch gesehen, mehr als die Hälfte der Rechenleistung einer Blockchain unter seine Kontrolle bringt. Dieses Szenario ist bei kleinen Blockchain-Netzwerken realistisch. Dadurch können neue Blöcke in die Blockchain hinzugefügt, eine Gabelung erzeugt und falsche Daten geschrieben werden, die anschließend von mehr als der Hälfte der bösartigen *Nodes* fälschlicherweise freigegeben werden können. Auch wenn diese Angriffsmethode nur sehr schwer in der Praxis umzusetzen ist, darf sie nicht außer Acht gelassen werden. Allerdings nimmt die Gefahr dieses Angriffs mit der Größe der Blockchain ab. [9, S.26]

Aus dem 51%-Angriff lässt sich bereits ablesen, desto größer die Macht, umso mehr Schaden kann angerichtet werden. Dies lässt sich ebenfalls auf einen bösartigen User umlegen. Kann dieser zu viel Einfluss, z.B. durch Prozessorleistung oder in Form von einem Teil im System, erlangen, können die Folgen gravierend sein. Damit können ganze Blöcke von anderen Benutzern verweigert oder eine Abzweigung in der Blockchain erstellt werden. Sobald diese größer als die ursprüngliche nicht bösartige Kette ist, schaltet der vorgegebene Blockchain Prozess automatisch auf diese als Hauptkette, mit den bösartigen Daten, um. Dieses Verhalten ist im Blockchain Protokoll festgelegt. [11, S.37]

Double spending stellt einen weiteren Angriffsvektor dar. Diese Art tritt vor allem bei Krypto-Währungen auf. Hier wird durch doppelte Transaktionen, die parallel durchgeführt werden, ein weiterer Nutzer getäuscht. Der Betrag wird auf den Angreifer und nicht auf den wahren Empfänger übertragen. [12, S.4] Dieser Angriff kann zusätzlich durch einen *fork* verstärkt werden. Dabei verweisen mehrere neue Blöcke auf ein und demselben Vorgängerblock. Wird ein *fork* dabei länger als die ursprüngliche Kette, wird dieser als allgemeingültig angenommen. [13, S.4] Die davor durchgeführte Transaktion aus dem nicht mehr

Kryptografische Algorithmen	**Typen**	**Funktionen**	**Einfluss durch *Quanten-Computer***
AES	Symmetrische Schlüssel	Verschlüsselung	Längerer Key notwendig
SHA-2, SHA-3	N/A	Hash Funktion	Längerer Output nötig
RSA	*Public key*	Signatur, Verschlüsselung	Nicht länger sicher
ECDSA, ECDH (Elliptic Curve Cryptography)	*Public key*	Signatur, Schlüsselaustausch	Nicht länger sicher
DSA (Finite Field Cryptography)	*Public key*	Signatur, Schlüsselaustausch	Nicht länger sicher

Tabelle 1.1.: Einfluss von Quanten-Computer auf gebräuchliche kryptografische Algorithmen [14, S.2]

gültigem *fork* wird somit ungültig.

Abgesehen von den bereits angeführten Angriffsvektoren entstehen durch die aufkommenden *Quanten-Computer* neue Probleme. Sie erhöhen die Gefahr, dass traditionelle asymmetrische Verschlüsselungsalgorithmen aufgebrochen werden können. [8, S.978] Wie in Tabelle 1.1 ersichtlich ist, sind einige weitverbreitete kryptografische Algorithmen dadurch gefährdet. Dies stellt für die Blockchain ebenfalls ein großes Sicherheitsrisiko dar. Verschlüsselte Information können gelesen und verändert werden, womit die Datenqualität nicht mehr gewährleistet ist.
Die fehlerhafte Implementierung eines kryptographischen Algorithmus kann zu Sicherheitslücken führen. Der Signaturalgorithmus *Elliptic Curve Digital Signature Algorithm* (ECDSA) dient zur Überprüfung auf Echtheit und dazu, dass nur der rechtmäßige Besitzer Zugriff auf einen Block besitzt. Dabei wird eine Signatur für eine Transaktion mittels privaten Schlüssels und einer Zufallszahl erzeugt. Die Zufallszahl muss für jede Transaktion zufällig gewählt werden und darf nicht vorhersagbar sein. Sicherheitsrisiken treten dann auf, wenn dieser Zufallswert durch eine falsche Implementierung erraten werden kann. [12, S.4] Wie bereits besprochen, kann durch *Quanten-Computer* der asymmetrische kryptografische Algorithmus RSA (Rivest–Shamir–Adleman) [15], welcher ebenfalls bei einer Blockchain zum Einsatz kommt, durch eine falsche Implementierung zu schweren Folgen, wie z.B. Datenverlust, führen. [8, S.978]

Die Blockchain gilt im Bereich DSGVO als sicher, da Benutzer anstelle von echten Identitäten mit generierten Adressen Transaktionen erstellen. Sie können zusätzlich unterschiedlichste Adressen generieren, um die Rückverfolgbarkeit weiter zu verschleiern. Da die *public keys* der einzelnen Benutzer allerdings öffentlich sichtbar sind, wurde bei Meiklejohn et al. 2013 [16, S.12] oder Biryukov et al. 2014 [17] festgestellt, dass es keine Garantie gibt, den Datenschutz aller User sicherzustellen. [18, S.20] Der pseudonymisierte Link lässt sich auf eine IP-Adresse zurückrechnen und ist anschließend direkt einem Benutzer zuweisbar. Zusätzlich können die einzelnen *Nodes*, die der Benutzer verwendet, festgestellt werden. Somit lässt sich der Ersteller der Transaktion genau ermitteln. [3, S.367-368] Laut DSGVO reicht eine Pseudonymisierung der Daten einer Person nicht aus. Daten müssen anonymisiert werden. Dabei darf ein Rückschluss auf die echten Daten nicht möglich sein. [18, S.19]
Des Weiteren ist in der DSGVO festgelegt, dass jede natürliche Person per Anfrage das Recht auf Löschung ihrer Daten hat. Dies wird als „*right to be forgotten*“ oder „Recht auf Vergessenwerden“ bezeichnet. [5, ErwG 66] Dabei reicht es bereits aus, den Personenbezug von personenbezogenen Daten zu entfernen, und somit eine „Anonymisierung“ durchzuführen. Allerdings stellt genau dies ein sehr großes Problem bei der Blockchain dar. Denn in der derzeitigen Implementierung bestehen keine Mechanismen, personenbezogene Daten aus der Blockchain zu verändern oder zu löschen. [19, S.12] Die Löschung eines Blockes macht die komplette Logik der Rückverfolgbarkeit von Transaktionen zunichte und zerstört die Blockchain vollständig. [18, S.25] Auf Grund der Dezentralität und fehlenden zentralen Kontrollstelle einer Blockchain ist die „Informationspflicht“ [5, Art. 13, Art. 14] schwer bis nicht realisierbar. Ist es trotzdem mit einer Technik möglich, Auskunft über Information zu geben, haben *Nodes* nicht zwingendermaßen die Verpflichtung, die Fragen zu beantworten. Aus aktueller Sicht zählt dies nicht zu deren Aufgabe. [18, S.25] Somit gibt es derzeit keine DSGVO-konforme Art einer Blockchain, die natürlichen Personen ausreichenden und vollständigen Schutz bietet. [18, S.4]

Durch die oben angeführten Schwachstellen ist bereits erkennbar, dass die Blockchain noch in ihrer Erforschungsphase steckt. [8, S.977] Aus diesem Grund wurde im Jahr 2018 der neue Gartner *Hype Cycle for Emerging Technologies 2018* publiziert. Daraus ist zu entnehmen, dass eine neue Innovation der klassischen Blockchain, die *Blockchain for Data Security*, bereits als neuer Trend erkennbar ist. Diese neuartige Blockchain soll die derzeitigen Sicherheitsmerkmale, die neue DSGVO und viele weitere Features beinhalten, damit die Blockchain für alle Anwendungsgebiete mit höchster Sicherheit zum Einsatz kommen kann. Die Umsetzung der *Blockchain for Data Security* wird weitere 5-10 Jahre Entwicklung benötigen. [20, S.2] Daher muss ein Rahmenwerk, in dieser Arbeit als *Framework* bezeichnet, geschaffen werden, welches allen Anforderungen gerecht wird.

1.2. Ziel der Arbeit

Das Ziel dieser Arbeit ist, eine Lösung für kryptografische Zugriffskontrollen zu finden, welche einer Blockchain eine höhere Sicherheit gewährleistet. Dadurch soll das Vertrauen in Blockchain als Speichermedium gesteigert werden und deren Einsatz zum neuen Standard heranwachsen. Zusätzlich wird berücksichtigt, dass die personenbezogene Datenverarbeitung in einer Blockchain ebenfalls der neuen DSGVO entspricht. Dabei dient das hier erarbeitete Framework, als Fundament für die *Blockchain for Data Security*.

1.3. Aufbau der Arbeit

Diese Diplomarbeit beruht auf einer Literaturanalyse, die sich auf aktuelle Fachjournale und Artikel bezieht und ist in vier Kapitel gegliedert. Den Beginn bilden die Einleitung mit Problemstellung, Forschungsziel und Fragestellungen. Kapitel 2 widmet sich der Literatur der Blockchain, den Zugriffskontrollen und der Kryptografie sowie der Sicherheit von Daten in Cloud Speicher und der europäischen Datenschutz-Grundverordnung. Dabei wird nicht näher auf das Thema Kryptowährung eingegangen. Anschließend wird mittels empirischen Verfahrens die Untersuchungsmethode begründet und die methodische Vorgehensweise erläutert. Fazit und Ausblick bilden den Abschluss dieser Arbeit.

1.4. Forschungsfragen

Die Datensicherheit und Integrität haben in einer Blockchain einen sehr hohen Stellenwert. Die zunehmende Komplexität der Daten in einer Blockchain stellt bei unzureichender Sicherheit ein Risiko dar. Aus diesem Grund geht diese Diplomarbeit den folgenden Fragestellungen und genaueren Betrachtungen nach:

- **Wie kann eine Blockchain Daten sicher abspeichern, so dass sie von nicht berechtigten Zugriffen geschützt ist?**
  Kryptografische Zugriffskontrollen sollen so eingesetzt werden, dass Daten vor fehlerhaften oder nicht gewünschten Zugriffen sicher aufbewahrt sind.
- **Wie muss die *Blockchain for Data Security*, die Blockchain der Zukunft, aussehen, damit sie in allen Anwendungsgebieten zum Einsatz kommen kann?**
  Daten müssen so aufbewahrt werden, dass sie die *Privacy* nach DSGVO erfüllen, die Daten sicher vor unerwünschten Zugriffen sind und verlustfrei abgespeichert werden können, sowie ihre Integrität gewährleistet ist.

- **Welche Art an Mechanismen oder Technik muss eine Blockchain beinhalten, damit die DSGVO erfüllt ist?**

  Dabei soll das „Recht auf Vergessenwerden“ trotz der Unveränderbarkeit einer Blockchain nicht im Widerspruch stehen. Zusätzlich soll die „Informationspflicht“ für natürliche Personen berücksichtigt werden.

2. Literaturanalyse

Um ein Verständnis für die Funktionalität, dem Aufbau und dem Einsatzzweck einer Blockchain zu erhalten, bildet die Literaturanalyse die Grundlage dieser Forschungsarbeit. Des Weiteren wird eine Übersicht verschiedener Zugriffskontrollen dargestellt, die in einer Blockchain zum Einsatz kommen können. Zum Abschluss wird in diesem Kapitel auf das europäische Datenschutzgesetz eingegangen, das die Wichtigkeit einer sicheren Blockchain untermauert.

2.1. Blockchain

In diesem Kapitel wird das Wachstum und das Einsatzgebiet der Blockchain beleuchtet. Zusätzlich wird die Architektur näher betrachtet und die Berechtigungsstruktur dargestellt.

2.1.1. Wachstum und Einsatzbereiche von Blockchain

Auf Grund ihrer Transparenz, öffentlichen Datenbank und der Verteilung wird der Blockchain ein großes Wachstum vorausgesagt. Durch das Offenlegen der digitalen Inhalte erhält die Blockchain Transparenz. Das Vertrauen in die Sicherheit wird durch den Einsatz von kryptografischen und mathematischen Algorithmen gewonnen. Diese garantieren derzeit eine sichere Verwahrung der Daten. Die Blockchain unterscheidet sich von herkömmlichen Datenbanksystemen durch ihre dezentrale Verwaltung. [8, S.976] Diese Revolution ist deshalb so besonders, weil sie in der Information und Computer Technologie die Technik hinter der Blockchain ganzheitlich offenlegt. [21, S.1] Für die Aufzeichnung und Sammlung von Transaktionen wird die sehr alte Technologie *Ledger* verwendet. Diese kommt allerdings in herkömmlicher Weise als zentrale *Ledger* zum Einsatz. Durch Weiterentwicklung ist die *distributed ledger technologie* (DLT) entstanden. Diese unterscheidet sich zu *Ledger* darin, dass sie verteilt ist. Die Blockchain stellt eine spezielle Type einer DLT dar, deren Bedeutung im nächsten Kapitel 2.1.2 erläutert wird.

Die Blockchain ist eine Anwendung, die alte, ausgediente Mechanismen mit mehr Effizienz sowie mit kostenoptimierten, effektiven und dezentralen Lösungen durchsetzt. Das macht sie als Einsatzgebiet sehr interessant und sie ist auf Wachstum ausgelegt. Die Blockchain findet derzeit vorrangig bei Währungen,

Verträgen und Speicherung von Identitäten sowie Lieferketten und Datenspeicherung Verwendung. Sie kann als neue Basistechnology-Innovation betrachtet werden, die gerade die Phase der Adaption von Technologien durchschritten hat. Als institutionelle Alternative für die Koordination von Personengruppen konkurriert sie mit Firmen und Märkten. [21, S.18]

In der Grafik 2.1 ist erkennbar, dass die Blockchain aktuell in Banken und der Finanzbranche am häufigsten zum Einsatz kommt. Darüber hinaus wird sie oft im Government und öffentlichen Bereich eingesetzt.

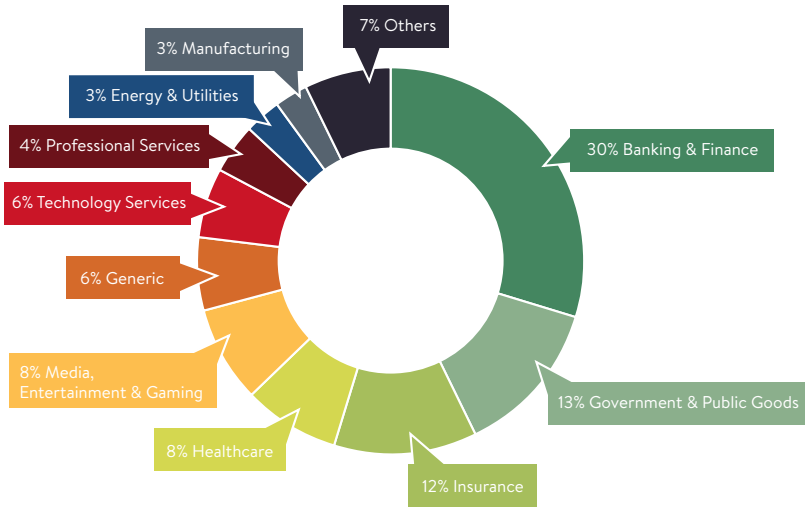


Abbildung 2.1.: Anwendungsbereiche Blockchain [22, S.37]

Dort findet die Blockchain, z.B. als eVoting, eine elektronische Stimmabgabe bei Wahlen, Anwendung. Des Weiteren liegt ihr Potenzial in regulatorischen Erfüllungen und auch im Steuerwesen. Bei Versicherungen kann mit der Blockchain die Steuerung von Verträgen abgewickelt werden. Wie in der Grafik 2.2 ersichtlich, kann etwa durch eine Abfrage der Vertrags-Blockchain durch das Auto ermittelt werden, ob die Versicherung bezahlt wurde und gegebenenfalls der Zündvorgang unterbunden werden. Im Finanzbereich kann die Blockchain für Geldüberweisungen oder Krediten zwischen zwei Kunden verwendet werden. Das Gesundheitswesen kann die Blockchain, z.B. im Bereich der elektronischen Patientenakte oder aber auch zur Identifikation von Zugriffen, einsetzen. Darüber hinaus kann im Industriezweig eine vollständige Lieferkette abgebildet und die Zurückverfolgbarkeit gewährleistet werden.
Damit ist erkennbar, wie durch den Einsatz einer Blockchain viele digitale Business-Prozesse gesteigert werden können. Dabei wird das Ziel verfolgt, ein verteiltes Business-Model zu verwenden, bei dem eine *peer-to-peer* Transaktion durchgeführt wird, die eine zentrale Verwaltung überflüssig macht. [23, S.1]

Nachdem das Wachstum und die Einsatzgebiete einer Blockchain betrachtet wurden, wird im nächsten

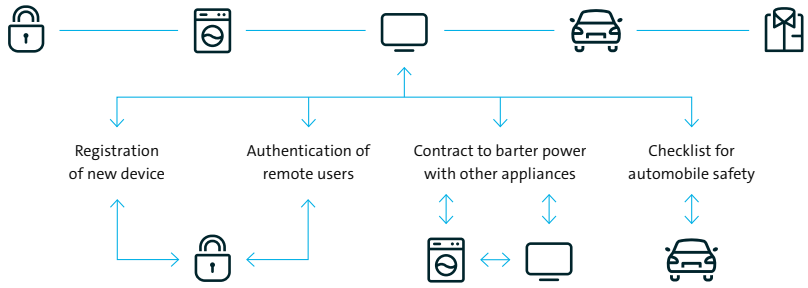


Abbildung 2.2.: IoT Einsatzgebiet für Blockchain [1, S.35]

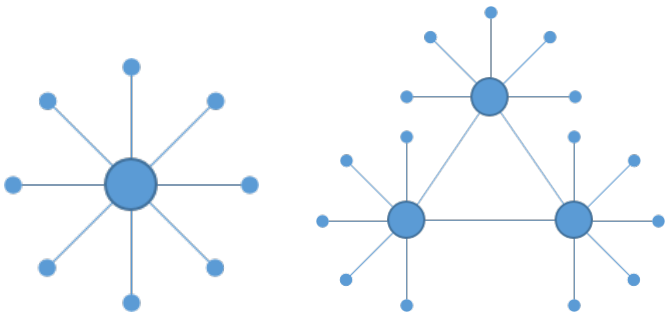
Kapitel die Architektur und die Funktionsweise geschildert.

2.1.2. Architektur und Funktionsweise einer Blockchain

Beim Einsatz einer Blockchain wird, im Vergleich zu herkömmlichen Datenbanken, dasselbe Ergebnis – eine Momentaufnahme der letztgültigen Einträge – retour geliefert. Darüber hinaus werden alle veralteten Einträge abgelegt und stehen jederzeit weiterhin zur Verfügung. Dadurch entsteht eine vollständige Historie aller zurückliegenden Aktivitäten und Transaktionen. Durch die dezentrale Verwaltung auf mehrere verteilten Plätze, entfällt bei einer Blockchain die zentrale Kontrolle und Administration, die bei einem gewohnten Datenbanksystem notwendig wären.
Im Wesentlichen kann die Blockchain als eine sehr spezielle Art einer Datenbank betrachtet werden. Sie verwendet altbekannte Technologien, wie z.B. das Netzwerk, die Kryptografie und Bestandsaufzeichnungen. Durch das gemeinsame Verwenden werden diese Technologien zu dem, was eine Blockchain heute ausmacht. Dadurch können viele neue Funktionen angeboten werden. [11, S.46] Darüber hinaus bietet die Blockchain neue Konsensverfahren an, welche für die Übereinstimmung der Daten verantwortlich sind. Sie verbraucht allerdings auch sehr viele Ressourcen, Bandbreiten und Speicher, um die Daten sicher ablegen zu können. [8, S.976] Die Blockchain gilt im Prinzip als transparente, verteilte und offene Technologie, die kryptografische und mathematische Algorithmen benutzt. [8, S.976]
Die Blockchain besitzt vier Haupteigenschaften [3, S.357], die in diesem Kapitel erläutert werden:

- **Dezentralität**
- **Unveränderbarkeit**
- **Anonymität**
- **Auditierung**

Dezentralität ist auf Grund ihrer Implementierung die Grundlage einer Blockchain. Die Daten werden nicht auf einem zentralen Platz abgelegt, sondern weitläufig im Netzwerk ausgebreitet und dadurch auf alle beteiligten *Nodes* verteilt. Ein *Node* ist ein individuelles System, Gerät oder Teilnehmer und stellt Teile des Blockchain-Netzwerkes dar. Diese werden zumeist von Benutzer betrieben. [11, S.53] *Nodes* können unterschiedliche Rollen einnehmen, die später unter 2.1.6 näher betrachtet werden. In der Abbildung 2.3 ist der Unterschied zwischen einem herkömmlichen Datenbanksystem auf der linken Seite und der Blockchain auf der rechten Seite ersichtlich. Bei der Variante (a) kommuniziert jeder mit einer zentralen Autorität. Bricht dieses System zusammen, sind alle Beteiligten davon betroffen. Dem gegenüber steht das dezentrale Netzwerk (b). Hier befinden sich viele Autoritäten im Netzwerk. Fällt eine dieser Komponenten aus, betrifft der Ausfall nicht das vollständige System.



(a) zentrales Netzwerk (b) dezentrales Netzwerk

Abbildung 2.3.: Zentrales und dezentrales Netzwerk [11, S.51-52]

Darüber hinaus ist die Blockchain, wie in Abbildung 2.4 dargestellt, auf ein verteiltest Netzwerk verstreut. Das hat zum Vorteil, dass die unterschiedlichen Autoritäten direkt miteinander verbunden sind. Somit können auch Teile des Netzwerkes zusammenbrechen und das System funktioniert weiterhin.

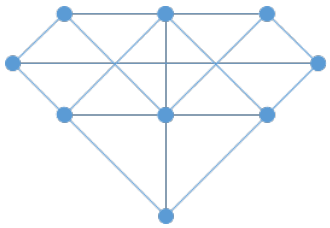


Abbildung 2.4.: Verteiltes Netzwerk [11, S.51]

Durch das verteilte Netzwerk ist es für einen Angreifer sehr schwer bis unmöglich, die Blockchain durch Attackieren eines einzelnen *Nodes* und somit das ganze System lahm zu legen. Dadurch fällt der *single point of contact* weg, um sie zu hacken. Das macht das ganze System sicher.

Wie in 2.1.1 bereits erwähnt, ist eine Blockchain eine *distributed ledger technology* (DLT). Eine DLT umschließt viele einzelne Blöcke. Jeder Block hat in sich viele Attribute, wie z.B. *Header*, *metadata* über den Block, Transaktionsinformationen und andere Daten. Dies wird in der Grafik 2.5 veranschaulicht. Die DLT ist ein digitales System, in dem Transaktionen und die Details dazu an vielen Plätzen abgespeichert werden, ohne einen zentralen Administrator, hier als *third party* bezeichnet, zu besitzen. Das übernehmen *Nodes* mit speziellen Rechten. Die einzelnen Blöcke werden dann in einer wachsenden Kette miteinander verbunden, siehe Abbildung 2.5, daher die Namensgebung *Blockchain*, und im dezentralen *peer-to-peer* Netzwerk verteilt. [19, S.26] Sie sind chronologisch angeordnet. Somit ist es möglich, auf alle Datenstände ständig zuzugreifen.

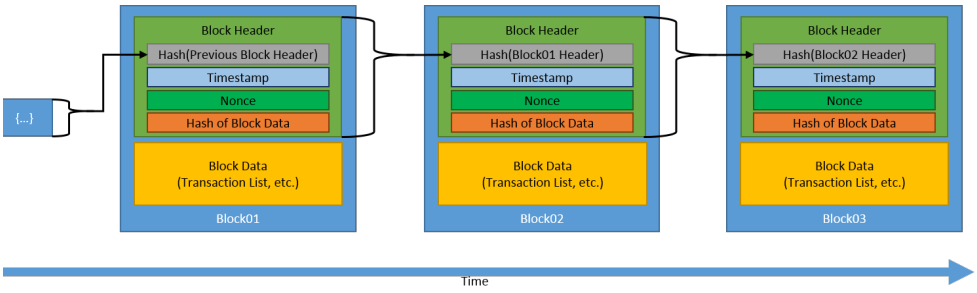


Abbildung 2.5.: Verkettung von Blöcken [11, S.17]

Wie in der schematischen Darstellung 2.6 erkennbar, besitzt jede Blockchain einen Ursprung. Dieser erste Block wird *Genesis Block* genannt. Danach kann es viele Abzweigungen geben. Allerdings besitzt die längste Kette Gültigkeit. Das ist jene Kette, in der die meiste Rechenleistung hineingeflossen ist. [1, S.13] Die restlichen Zweige wurden in die Blockchain hinzugefügt, enden aber einfach.

Unveränderbarkeit ist eine weiter Eigenschaft der Blockchain. Da die Transaktionen im gesamten Netzwerk veröffentlicht und anschließend auf ihre Gültigkeit geprüft werden, finden sie nicht genügend *Nodes*, die diese durch ein Konsensverfahren, wie unter 2.1.5 abgebildet, bestätigt haben, um gültige Blöcke zu werden.

Zum Beispiel passiert das durch Vergleichen der digitalen Signatur des Initiators der Transkation mit deren Adresse. Dabei wird der Block durch einen Hash-Algorithmus vor seiner Veränderung geschützt und mit dem Hash des vorhergehenden Blockes verbunden. Die Blöcke können anschließend nicht mehr verändert werden und sind daher vor Manipulation geschützt. Der Aufwand einen Block in der Vergangenheit zu verändern ist so groß, dass bis zum letztgültigen Block alle Blöcke verändert werden

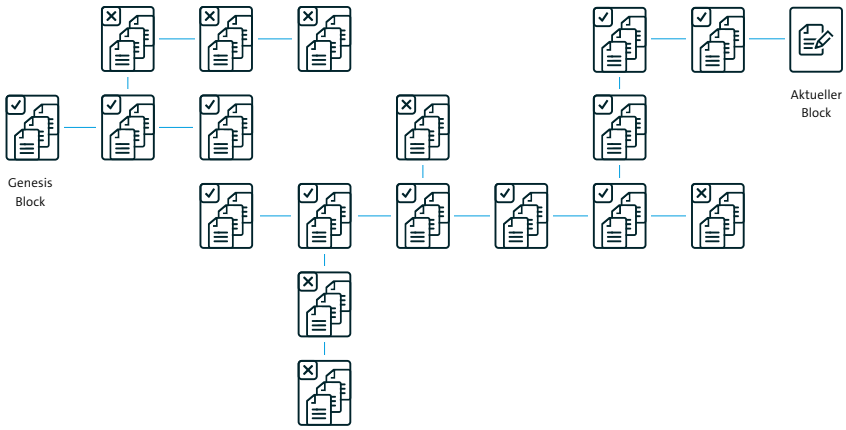


Abbildung 2.6.: Schematische Darstellung einer Blockchain [1, S.13]

müssen, damit dieser gültig wird. Erst dann können theoretisch die Blöcke ausgetauscht und den anderen *Nodes* zur Bestätigung vorgelegt werden. Allerdings haben diese in ihrer Blockchain bereits einen anderen gültigen Block, wodurch die Veränderung niemals bestätigt werden kann. Dieses Verfahren macht die Blockchain zu einer besonderen Revolution. Keine andere Technik bringt blockweises Prüfen, Verkettung von Blöcken und dezentrales Management mit sich. Da diese Funktionen von vielen Akteuren parallel durchgeführt werden, lässt sich die Blockchain nicht von einer zentralen Stelle kompromittieren. [1, S.11]

Anonymität wird durch das Verschleiern von Informationen sichergestellt, dass das Rückverfolgen zu einem realen Benutzer erschwert. Somit werden persönliche, private Informationen sicher aufbewahrt. [8, S.976]

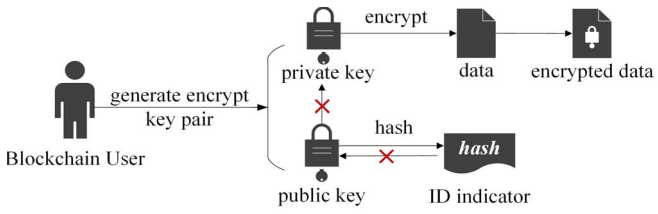


Abbildung 2.7.: Schutz der Privatsphäre in einer Blockchain [8, S.976]

In der Abbildung 2.7 ist zu erkennen, dass es nicht möglich ist, vom *public key* des Benutzers auf dessen *private key* zurückzuschießen. Somit ist es auch nicht möglich, auf seine verschlüsselten Daten zuzugreifen. Zusätzlich ist es ebenfalls unmöglich, aus dem erzeugten *Hash* den *public key* des Users zu errechnen. Diese Sicherheit ist allerdings nur dann gewährleistet, solange die aktuellen kryptografischen

Algorithmen nicht aufgebrochen werden. [8, S.976]

Auditierung wird durch die Validierung und Verkettung der einzelnen Blöcke gewährleistet. Jeder Benutzer kann auf Grund eines Zeitstempels und dem Link zum vorherigen Block die Echtheit überprüfen. So lässt sich die Blockchain bis zum *Genesis Block* zurückverfolgen. [3, S.357]

Im nächsten Schritt wird auf den einzelnen Block einer Blockchain eingegangen. Dieser setzt sich, wie bereits kurz besprochen, aus unterschiedlichen Bereichen zusammen. Wie die Abbildung 2.8 zeigt, kann zwischen dem Block *Header* und dem Block *Body* unterschieden werden. Der *Header* beinhaltet folgende Felder:

- ***Block version*:** Hier ist die Version des Blocks abgespeichert.
- ***Parent block hash*:** Hier ist der 256-Bit *Hash* von dem vorherigen Block hinterlegt. Der *Genesis Block* hat hier keinen Wert.
- ***Merkle tree root hash*:** Hier ist in Form eines *Merkle tree* der *Hash* für die zu diesem Block dazugehörigen Transaktionen gespeichert. Damit wird sichergestellt, dass dieser Block nicht verändert werden kann.
- ***Timestamp*:** Ist der Zeitstempel, von jenem Zeitpunkt in der der Block erstellt wird.
- ***nBits*:** Steht für einen hash in kompakter Form.
- ***Nonce*:** Dies ist der *Nonce*, der benutzt wurde, um diesen Block zu erstellen. Er startet normalerweise bei 0 und erhöht sich bei jeder *Hash* Kalkulation.

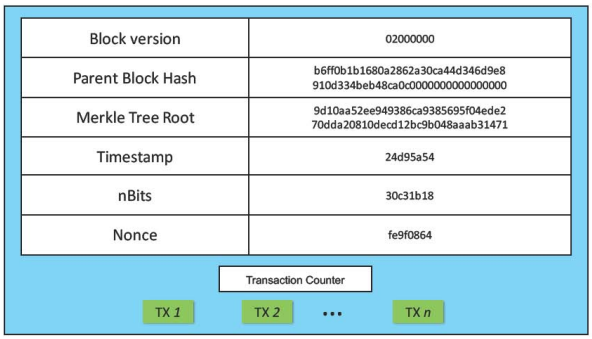


Abbildung 2.8.: Struktur eines Blockes [3, S.356]

Der *Body* beinhaltet die einzelnen Transaktionen sowie die Anzahl der Transaktionen. Diese Transaktionen werden durch *Mining* von Blöcken erstellt. Mehr Informationen dazu sind unter 2.1.5 ersichtlich.

Blockchain Type			Lesen	Schreiben	Bestätigen	Beispiele
öffentliche		*Public permissionless*	Offen für alle	Alle	Alle	Bitcoin, Ethereum
öffentliche		*Public permissioned*	Offen für alle	Berechtigte Teilnehmer	Alle oder ein Teil der berechtigten Teilnehmer	Sovrin
private		Konsortium	Auf einen berechtigten Teil eingeschränkt	Berechtigte Teilnehmer	Alle oder ein Teil der berechtigten Teilnehmer	Mehrere Banken betreiben eine geteilte *Ledger*
private		*Private permissioned ('enterprise')*	Vollständig privat oder eingeschränkt auf berechtigte Nodes	Nur Berechtigte	Nur Berechtigte	Interne Banken-*Ledger*, geteilt innerhalb der Firma und Niederlassungen

Tabelle 2.1.: Unterschied zwischen *permissionless* und *permissioned* Blockchain [22, S.37]

Aufgrund der zahlreichen Anwendungsarten muss bei einer Blockchain auch bei der Architektur unterschieden werden. Bei der Architektur bestehen zwei verschiedene Arten, eine *permissonless* und eine *permissioned* Blockchain. Diese Unterscheidung wirkt sich auch auf das Rechte-Management einer Blockchain aus. In der Tabelle 2.1 werden die Unterschiede dargestellt.
Dabei wird zwischen öffentliche bzw. *permissonless* Blockchain im Internet und private bzw. *permissioned* Blockchain-Netzwerke im Intranet unterschieden. [9, S.15] Ein öffentlicher Blockchain kann von jeder Person ohne Zugriffsbeschränkungen gelesen und beschrieben werden. Somit darf jeder Teilnehmer, sofern das entsprechende Konsensverfahren eingehalten wird, Blöcke hinzufügen. [22, S.11] Dies wird z.B. bei Krypto-Währungen eingesetzt. Hingegen kann eine private, geschlossene Blockchain nur von berechtigten Personen oder *Nodes* gelesen und beschrieben werden. Diese Art wird von Banken bevorzugt. [22, S.13] Der dabei zusätzliche Sicherheitsmechanismus verhindert die Möglichkeit der Manipulation. In Tabelle 2.1 wird der Begriff „Bestätigen“ als Synonym für *Mining* verwendet und bezeichnet jemanden der berechtigt ist, Blöcke hinzufügen. [22, S.11] Aus der Tabelle 2.1 kann bereits abgelesen werden, dass es unterschiedliche Berechtigungsstruktur gibt. Mehr dazu unter 2.1.6.

2.1.3. Klassifizierung einer Blockchain

Zusätzlich zu den Anwendungsarten einer Blockchain kann diese noch in unterschiedlichen Kategorien eingeteilt werden. Wie in der Tabelle 2.2 ersichtlich, wird zwischen einer öffentlichen Blockchain, einer Blockchain innerhalb einer Vereinigung und einer privaten Blockchain auf Basis diverser Eigenschaften unterschieden.

Eigenschaften	**öffentliche Blockchain**	**Konsortium-Blockchain**	**private Blockchain**
Konsensfindung	Alle Miner	ausgewählte Nodes	eine Organisation
Leserecht	öffentlich	kann eingeschränkt werden	kann eingeschränkt werden
fälschungssicher	nahezu unmöglich zu fälschen	nicht vollständig erfüllt	nicht vollständig erfüllt
leistungsfähig	Niedrig	Hoch	Hoch
zentralisiert	Nein	teilweise	Ja
Konsensverfahren	*Permissionless*	*Permissioned*	*Permissioned*

Tabelle 2.2.: Klassifizierung einer Blockchain [3, S.358]

Da die öffentliche Blockchain für alle verwendbar ist, verfügt sie über großes Interesse. Hingegen wird eine Konsortium-Blockchain oft für Business-Anwendungen, wie z.B. Hyperledger, verwendet. Private Blockchains werden in Firmen zur Effizienzsteigerung und Auditierung eingesetzt. [3, S.358]

2.1.4. Nodes und ihr Aufgabengebiet

In diesem Abschnitt werden der Unterschied und die Aufgabe der *Nodes* erläutert. Ein *Node* ist ein individuelles System wie z.B. ein Computer, der einen Bestandteil des Blockchain-Netzwerkes darstellt. *Node* bedeutet Knotenpunkt. Sie werden zum Teil von Benutzern betrieben. Der *Node* wird als Regulator bzw. als Controller in die Blockchain implementiert. Dabei gibt es verschiedene *Nodes* mit unterschiedlichen Aufgaben. [1, S.23] Der *Full Node* validiert jede Transaktion in der Blockchain auf seine Richtigkeit. Die Überprüfung erfolgt über ein in der Blockchain implementiertes Konsensverfahren. Mehr dazu im

nächsten Kapitel 2.1.5. Liefert die Validierung des Konsensverfahren ein negatives Ergebnis retour, wird der *Full Node* die Änderung in der Blockchain verwerfen und nicht in die Blockchain inkludieren. Ein *Full Node* besitzt eine vollständige Kopie der Blockchain. Nur ein Konsens führt dazu, dass der neue Block in die Blockchain angehängt wird.

Der *SPV Node* oder auch *Light Node* bzw. *Lightweight Node* genannt, beinhaltet kein vollständiges Abbild der Blockchain und überprüft die Blöcke nur anhand der *Header* auf ihre Richtigkeit. Dabei wird der *Merkle Tree* überprüft. Die Verifizierung erfolgt über *simplified payment verification* (SPV). Dadurch wird weniger Rechenleistung und Speicher benötigt. [13, S.2087] [22, S.55]

In Tabelle 2.2 ist ersichtlich, dass eine Konsortium-Blockchain ein *permissioned* Konsensverfahren hat. Diese Art von Blockchain besitzt einen Eigentümer. Ein näherer Blick auf die *Nodes* zeigt, dass diese indirekt durch den Eigentümer durch Hinzufügen von Neuen und Entfernen von unerwünschten *Nodes*, direkt einen Einfluss auf die Blöcke haben. In weiterer Folge können auch Blöcke ausgetauscht und dadurch indirekt die Gültigkeit andere Blöcke gelöscht werden. [11, S.34]

2.1.5. Konsensverfahren

Das Konsensverfahren ist dafür verantwortlich, die Übereinstimmung der Daten zu kontrollieren. Durch ein Model werden Blöcke erfolgreich validiert und automatisch in die Blockchain hinzugefügt. Da nicht alle Verfahren überall einsetzbar sind, werden unterschiedliche Arten implementiert. In der Tabelle 2.3 ist eine Übersicht der meist verwendeten Algorithmen dargestellt.

Eigenschaften	**PoW**	**PoS**	**PBFT**	**DPOS**	**Ripple**	**Tendermint**
Identitäts-Management	Offen	Offen	*Permissioned*	Offen	Offen	*Permissioned*
Energie sparend	Nein	Teilweise	Ja	Teilweise	Ja	Ja
Toleriertes Limit	<25% Rechenleistung	<51% Anteile	<33.3% fehlerhafte Repliken	<51% Prüfer	<20% fehlerhafte *Nodes* in UNL	<33.3% byzantinisch Stimmengewalt
Beispiele	Bitcoin	Peercoin	Hyperledger Fabric	Bitshares	Ripple	Tendermint

Tabelle 2.3.: Vergleich von Konsensverfahren [3, S.362]

Bei allen bereits erwähnten Vor- und Nachteilen einer Blockchain darf der wichtige Grundsatz des CAP-Theorems nicht vergessen werden. Die Abkürzung CAP steht für *Consistency* (Konsistenz), *Availability* (Verfügbarkeit) und *Partition Tolerance* (Ausfalltoleranz). CAP sagt aus, dass in einem verteilten System die drei Punkte Konsistenz, Verfügbarkeit und Ausfalltoleranz nicht gleichzeitig erfüllt werden können. Eine Blockchain stellt die Ausfalltoleranz durch die große Anzahl an Knoten sicher, und die Verfügbarkeit wird durch die Vielzahl der Instanzen sichergestellt. Um die Konsistenz zu gewährleisten, dienen Ansätze, wie z.B. *proof of work* und *proof of stake*. Steht die Konsistenz an oberster Stelle, muss theoretisch auf die uneingeschränkte Verfügbarkeit oder aber auf Ausfalltoleranz verzichtet werden. [24, S.4]

Beim *proof of work* (PoW) wird das Hinzufügen von Blöcken und Transaktionen in die Blockchain dadurch erschwert, dass ein kryptografisches Puzzle gelöst werden muss. Dieser Vorgang wird als *Mining* von Blöcken bezeichnet. Erst dann ist es möglich, den Block zu verankern und als valide Blöcke anzuerkennen. [12, S.4] Dieser Vorgang ist bei PoW sehr kostenintensiv, da sehr viel Energie zum Lösen der kryptografischen Aufgabe benötigt wird. Dabei muss durch oftmaliges Probieren ein digitaler Hash-Wert für den Block Header gefunden werden, welcher kleiner als der Zielwert ist. Der gefundene *Nonce*, eine Zufallszahl, wird zur Validierung an einen *Node* geschickt. [11, S.19]
Dem gegenüber steht das *proof of stake* PoS Verfahren, welches eine energiesparende Alternative zu PoW darstellt. Allerdings kann hier bei einem Anteil >51% das *Mining* von Blöcken beeinflusst werden. Das Sprichwort „Reiche werden noch reicher“ kommt dabei zur Geltung. Somit wird davon ausgegangen, dass Personen mit mehr Besitz, die den Besitz von Blöcken nachweisen, das Netzwerk weniger attackieren als Personen mit weniger Anteilen. PoS benötigt Personen, die den Besitz von Blöcken nachweisen. [3, S.360] Des Weiteren gibt es eine Erweiterung von PoS, die *delegated proof of stake* (DPOS) Methode. Diese ist der PoS sehr ähnlich. Hier werden jedoch bestimmte Prüfer in einem Auswahlverfahren für die Validierung bestimmt.
Bei einem weiteren Verfahren, dem *practical byzantine fault tolerance* (PBFT) Verfahren, dient das „byzantinische Generäle Problem“ als Grundlage. Dieses beruht nicht auf einer wahren Begebenheit, sondern ist ein Gedankenexperiment, das einen Sachverhalt darstellen soll. Generäle planen einen Angriff auf eine Stadt. Um erfolgreich zu sein, müssen sie sicherstellen, dass alle Truppen gleichzeitig angreifen. Der Befehl zum Angriff wird über Boten verteilt. Allerdings muss gewährleistet werden, dass kein Verräter unter ihnen weilt und einen falschen Befehl weitergibt. Dabei geht es um Vertrauen und Konsensverfahren, also wie ohne eine zentrale Kontrollstelle sichergestellt werden kann, dass niemand bösartige Meldungen/Transaktionen durchführt. [13, S.2111] Die Blockchain nutzt dafür den kostspieligen *proof of work* als Lösung. PBFT ist im Gegensatz energiesparend. Dies wurde durch eine Art von

Reihenfolge der *Nodes* geschaffen, wobei es immer einen primären *Node* gibt.
Zuletzt wird noch das *Ripple* Verfahren betrachtet. *Ripple* beruht, um einen Konsens zu finden, auf ausgewählte vertraute Autoritäten. Bei einer anstehenden Transkation wird eine Anfrage an eine *Unique Node List* (UNL) gestellt. Wird unter allen *Nodes* eine Vereinbarung mit mehr als 80% Übereinstimmung gefunden, wird die Transaktion in die Blockchain hinzugefügt. [13, S.2111] [11, S.361]

Abschließend ist zu erwähnen, dass ein gutes Konsensverfahren energiesparend, einfach zum Einsetzen und sicher sein muss. Allerdings haben die derzeitigen Methoden noch Mängel und sind z.B. sehr zeitaufwändig. Hier kann in neuen Verfahren eine Verbesserung erreicht werden. [3, S.362]

Sehr häufig ist der Begriff *smart contract* im Zusammenhang mit einer Blockchain zu hören. *Smart contract* wird als computerunterstützte Technologie verwendet, um verbreitete vertragliche Abmachung zu erfüllen. Zumeist ist das eine Sammlung aus *Code*, d.h. entwickelte Software und Daten, die auf Funktionen referenziert. Sie werden automatisiert von Computer ausgeführt und mit den vertraglichen Vereinbarungen abgeglichen. Die Werte, die berechnet werden müssen, werden als Parameter dem *smart contract* übermittelt, der dann ein Ergebnis retour liefert. Dabei muss es sich nicht immer um transaktionsrelevante Funktionen handeln. Sie können auch einfach eine Kalkulation durchführen, Informationen speichern oder eine Zufallszahl retour liefern. [11, S.V] *Smart contracts* werden dem gesamten Blockchain-Netzwerk zur Verfügung gestellt und liefern bei derselben Eingabe immer das gleiche Ergebnis retour. [9, S.13] Die Implementierung der *Smart contracts* hat die Blockchain um viele Möglichkeiten erweitert und damit zum Durchbruch beigetragen. [11, S.32]

2.1.6. Berechtigungsstruktur einer Blockchain

Nachdem im letzten Kapitel die Architektur der Blockchain besprochen wurde, wird hier die Berechtigungsstruktur erläutert. Die Blockchain unterscheidet sich deutlich von einem herkömmlichen Datenbank-System. Diese bieten die Möglichkeit, Daten in der Datenbank zu erstellen, sie zu lesen, zu verändern und Daten zu löschen. Dafür wird das Akronym „CRUD“ verwendet und steht für folgende vier grundlegenden Operationen:

- ***Create*:** Daten anlegen oder erstellen.
- ***Read*:** Einträge lesen.
- ***Update*:** Bereits hinzugefügte Daten aktualisieren.

- ***Delete*:** Bestehende Datensätze löschen.

Die Blockchain unterscheidet sich hier insofern, dass nur „CR“ möglich sind.

- ***Create*:** Daten erstellen oder schreiben.
- ***Read*:** Datensätze lesen.

Dadurch können Datensätze nur erstellt und gelesen werden. Die Blockchain wurde wie im vorigen Kapitel bereits mehrfach erwähnt, dafür generiert, Daten sicher und veränderungsresistent aufzubewahren. D.h. Daten können nur angehängt und gelesen werden. Alte Blöcke können durch neue ersetzt werden, jedoch bleiben diese weiterhin in der Blockchain erhalten. Es gibt keinen Prozess, um sie zu löschen. Die neuen Blöcke ergänzen die alten Einträge um neue Inhalte. Dabei ist die Historie immer rückverfolgbar. [11, S.44]

2.2. Zugriffskontrollen

Nachdem am Ende des letzten Kapitels 2.1 die Berechtigungsstrukturen behandelt wurden, werden in diesem Abschnitt die Zugriffskontrolle im Allgemeinen betrachtet. Diese kommen in jeder Blockchain in unterschiedlichen Ausprägungen, wie z.B. als *permissioned* Blockchain, zur Anwendung. Im nachfolgenden Kapitel 2.3 wird auf die kryptografische Zugriffskontrolle näher eingegangen.

Generell bietet die Zugriffskontrolle einem Benutzer oder einem Prozess die Möglichkeit, Rechte, wie Schreiben, Lesen oder Ändern von Datenobjekten an. Datenobjekte können Dateien, Datenbanken oder ähnliche Objekte sein. Typischerweise inkludiert sie Authentifikation und Autorisierung.
Bei der Authentifikation ist ausschlaggebend, dass ein System die vorgegebene Identität eines Benutzers verifizieren kann. Ein Passwort ist hier z.B. eine standardisierte Methode.
Dem gegenüber steht die Autorisierung. Hier wird festgestellt, ob der angegeben Benutzer auf einen Inhalt zugreifen darf und dafür über die nötigen Rechte verfügt. Diese können anhand von Regeln unterschiedlich ausfallen. Eine Berechtigung erlaubt ihm z.B. eine Datei zu lesen oder zu ändern. Einer Regel können wiederum unterschiedliche Berechtigungen zugewiesen werden. Ebenso können einem Benutzer mehrere Rollen zugeteilt werden.
Bei der Zugriffskontrolle geht es immer um die Vertraulichkeit und Integrität der Daten. Vertraulichkeit garantiert, dass nur von autorisierten Personen auf Daten zugegriffen werden können. Hingegen stellt die Integrität sicher, dass Daten unverändert und von böswilligen Zugriffen geschützt sind. Daher ist ein

Security System zuverlässig, wenn es die Verfügbarkeit, Glaubwürdigkeit und Sicherheit gewährleistet. Verfügbarkeit bedeutet, dass ein Benutzer auf Daten mit den entsprechenden Berechtigungen zugreifen darf. Glaubwürdigkeit stellt fest, dass Daten unverändert und im Originalzustand vorliegen. Die Sicherheit durch Berechtigungen hingegen dafür Sorge trägt, dass ein Benutzer die Daten nicht beschädigen kann. [25, S.12-13]

Um die hier erwähnten Methoden und Prozesse zu bewerkstelligen, gibt es unterschiedliche Zugriffskontrollmodelle. Dabei ist zu berücksichtigen, dass diese Modelle bereits sehr lange bestehen und dennoch nicht an Bedeutung verlieren.

Das einfachste Modell ist das *Discretionary Access Control* (DAC) System. DAC wird individuell auf jedes einzelne Objekt vergeben und daher nicht von einer zentralen Stelle gesteuert. DAC ist sehr einfach zu implementieren und muss deshalb auch keine Regeln befolgen. [25, S.13]

Ein weiteres Modell der Zugriffskontrolle ist das *Mandatory Access Control* (MAC) Modell. Dieses hat eine regelbasierte Strategie und ist in den Grundzügen das am weitest verbreitete Zugriffskontrollsystem. Im Gegensatz zum DAC werden hier die Zugriffe zentral verwaltet. Dies bietet eine Sicherheit, die bei DAC fehlt. Benutzer, die sich in ein System anmelden, verfügen bereits über die Zugriffsrechte auf Dateien. [25, S.15] Ein Referenzmonitor, welcher eine logische Einheit in IT-Systemen darstellt, ist für die Zugriffsrechte zuständig. Dieser entscheidet anhand von Regeln, welche Rechte ein Benutzer oder ein Prozess auf Objekte besitzt.

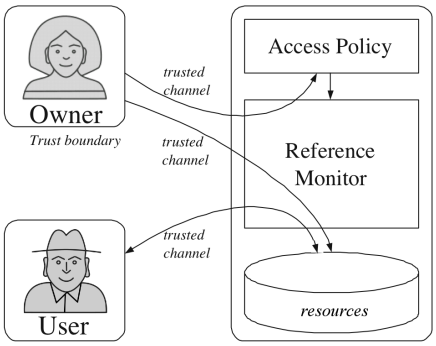


Abbildung 2.9.: Referenzmonitor für Zugriffskontrollen [26, S.2]

Wie in Abbildung 2.9 ersichtlich, vergibt der *Owner* (Besitzer) der Daten Rechte auf Ressourcen, um anschließend dem User (Benutzer) dementsprechende Berechtigung auf Informationen bereitzustellen. [26, S.2]

Des Weiteren gibt es ein *role-based access control* (RBAC) System, dass sich zwischen dem DAC und dem MAC ansiedelt. Diese Zugriffskontrolle soll das zu stark dezentrale DAC und das MAC, welches

wieder zu starr wirkt, erweitern. RBAC beruht auf der Basis von Rollen. Einem Benutzer werden nicht direkt Rechte auf Dateien erteilt, sondern ihm wird eine Rolle zugeteilt und diese Rolle besitzt das Recht auf Dateien zuzugreifen. Eine Rolle kann z.B. mit einer Jobposition oder mit einer Berechtigungsstufe verknüpft sein, mit der jedes Mitglied dieser Rolle Dateien lesen kann. Der Vorteil bei dieser Variante ist, dass eine Person mehrere Rollen und dadurch unterschiedliche Berechtigungen besitzen kann. Viele Benutzer können hier Mitglied in einer Rolle sein. Jedoch kann es dabei zum Problem kommen, dass Rollen gegenseitig mit den zugewiesenen Rechten im Konflikt stehen, so dass Berechtigungen, die zuvor erteilt wurden, aufgehoben werden. Diese können zu Schwachstellen führen, womit das RBAC Modell bei Änderungen von Rechten einer strengen Kontrolle unterliegen muss. [25, S.16]
Das *Multilevel Access Control* (MLS) ist eine weiter Methode um Zugriffe zu steuern und setzt auf das MAC Modell auf. Dabei werden Sicherheitsregeln genau beschrieben und festgelegt, was mit welchem Mechanismus wie zu schützen ist. Diese Regel legt fest, wie ein definiertes Ziel mit welchem festgelegten Profil zu schützen ist. Das bekannteste Modell von MLS ist das *Bell-LaPadula* Sicherheitsmodel (BLP).

streng geheim
geheim
vertraulich
öffentlich

Tabelle 2.4.: Klassifizierung Bell-LaPadula Sicherheitsmodel [25, S.18]

Das BLP System setzt auf Klassifizierung von Dateien in unterschiedlichen Kategorien nach ihrer Sensitivität auf. Dies ist in Tabelle 2.4 ersichtlich. Dabei wird sichergestellt, dass Personen mindestens die gleiche Stufe wie das Objekt besitzen müssen, um es lesen zu können. Das bedeutet, dass eine Person mit der Berechtigung zu „streng geheim“, darf Dokumente mit der Klassifizierung „geheim“ lesen, allerdings nicht umgekehrt. Dieses System kommt unter anderem im Militärbereich zum Einsatz. [25, S.18]
Es gibt noch einige weitere Modelle wie z.B. *Biba Integrity*, *Chinese Wall* und *Clark-Wilson Modell*. Besonders von Bedeutung ist das *Cryptographic Access Control* (CAC) Modell, [25, S.19] welches in Kapitel 2.3.4 erläutert wird.

2.3. Kryptografie

Dieses Kapitel behandelt die Kryptografie, die einen der wesentlichsten Bestandteile und die Basis für eine sichere Blockchain darstellt. Sie ist die Wissenschaft zur Verschlüsselung und Entschlüsselung von Informationen und hat zum Zweck, diese gegenüber Dritten geheim zu halten. Des Weiteren dient sie zur Bewertung der Sicherheit von kryptografischen Verfahren gegen unbefugte Zugriffe. Die Kryptografie wurde so entworfen, dass sie in einer nicht vertrauensvollen Umgebung, welche das Internet darstellt, volle Sicherheit bereitstellt. Deshalb kann sie auch außerhalb von eigenen Sicherungsgrenzen eingesetzt werden. Sie verfolgt folgende Hauptziele:

- **Vertraulichkeit:** Die Geheimhaltung von Informationen ist der offensichtlichste und bekannteste Anwendungsfall eines kryptografischen Verfahrens. Nur Personen mit einer Berechtigung, einem Schlüssel zum Entsperren, können Daten oder eine Nachricht und dessen Inhalt lesen.
- **Integrität:** Mit der Integrität wird sichergestellt, dass die Nachricht unversehrt bleibt. Daten werden mit einer Signatur versehen, die gewährleistet, dass die Daten unverändert sind.
- **Authentizität:** Die Identität des Absenders einer Nachricht muss für den Empfänger eindeutig identifizierbar sein.
- **Gültigkeit:** Die Gültigkeit wird durch Verwendung von Schlüssel bestätigt.
- **Nichtabstreitbarkeit/Verbindlichkeit:** Hier soll gewährleistet werden, dass ein Absender einer Nachricht nachweislich und rückwirkend überprüfbar bleibt und seine Urheberschaft nicht bestreiten kann.

In der Blockchain werden kryptografische *Hash* Funktionen [11, S.7], asymmetrische Kryptografie, die digitale Signatur zur Speicherung der Transkationen und Funktionen zur Verschleierung von Adressen verwendet. Diese Methoden machen eine Blockchain gegen Veränderungen von Blöcken oder Fälschung von Transaktionen robust. [11, S.7]
Das asymmetrische Verfahren beruht auf dem Prinzip von öffentlichen und privaten Schlüsseln, hier als *public keys* und *private keys* bezeichnet, und dient als Basis für die meisten kryptografischen Methoden. Die digitale Signatur, die ebenfalls auf asymmetrische Schlüssel zurückgreift, wird zur Überprüfung auf Echtheit als Unterzeichnung verwendet. Die kryptografische *Hash* Funktion wird beim *Mining* zum Lösen eines kryptografischen Puzzles eingesetzt. Dieses ist besonders rechenintensiv und dient zur Konsensfindung. [11, S.IV]

In weiter Folge werden die einzelnen Methoden und Verfahren genauer betrachtet.

2.3.1. Symmetrische und asymmetrische Verschlüsselung

Um die Sicherheit in einem Netzwerk zur Verfügung zu stellen, wurden unterschiedliche Methoden entworfen. Dazu zählt das symmetrische und asymmetrische Verfahren. Diese Verfahren werden für verschiedene Anwendungsfälle eingesetzt.

Die symmetrische Verschlüsselung beruht z.B. auf dem Prinzip von *Data Encryption Standard* (DES), welches nur eine Schlüssellänge von 56 Bit verwendet und als veraltet gilt. Aufgrund der kurzen Schlüssellänge ist DES nicht sicher. Dem gegenüber steht der Nachfolger *Advanced Encryption Standard* (AES), der von Joan Daemen und Vincent Rijmen (Rijndael) entworfen wurde. Diese Methode ist eine Blockverschlüsselung und verwendet zum Verschlüsseln und Entschlüsseln ein und denselben Schlüssel mit deutlich längeren Schlüssel als DES. Diese liegt bei 128, 192 oder 256 Bit. Der Nachteil bei AES liegt darin, dass der Schlüssel über einen weiteren sicheren Kanal ausgetauscht werden muss oder es besteht bereits eine vertrauensvolle Beziehung mit einem sicheren Kanal, um den *pre-shared* Key auszutauschen. Die Schlüssellängen sind deutlich kürzer als bei der asymmetrischen Verschlüsselung. Die Funktionsweise beruht auf dem Prinzip von mehrmaligen Byte-Ersetzungen (Substitutionen) mittels Ersetzungstabelle (S-Box), Verwürfelungen (Permutationen) von Reihen sowie Spalten und linearen Transformationen. [27, S.506] Das AES Verfahren ist trotz ihrer geringen komplexen Implementierung sehr sicher und gilt als höchst effizient.

Zur asymmetrischen Verschlüsselung gehört unter anderem das RSA-Verfahren, welches von Rivest, Shamir und Adleman entwickelt wurde. Dabei kommt ein Schlüsselpaar mit unterschiedlichen Schlüssel zum Einsatz. Der *private* Key dient zum Verschlüsseln der Daten. Dieser wird geheim beim Benutzer aufbewahrt. Der *public* Key übernimmt hingegen die Aufgabe der Entschlüsselung der Daten, um Nachrichten zu dekodieren. Dabei werden immer die entgegengesetzten Schlüssel zum Verschlüsseln und Entschlüsseln herangezogen. Alternativ kann der *public* Key auch zum Verschlüsseln und der *private* Key zum Entschlüsseln eingesetzt werden. Der Unterschied ergibt sich auf der einen Seite durch das Verschlüsseln von Daten und auf der anderen Seite durch das Signieren von Daten. [11, S.11] Da kein Rückschluss vom *public* Key zum *private* Key möglich ist, kann der öffentliche Schlüssel frei verteilt werden. Abgeleitet aus beiden Schlüssel lässt sich erkennen, dass der *public* Key die Befähigung hat, Daten und Information zu lesen, also eine Leseberechtigung besitzt. Der *private* Key verfügt dagegen über die Möglichkeit, Daten zu schreiben und hat daher eine Schreibberechtigung. [28, S.2] Die Sicherheit bei RSA basiert auf dem mathematischen Prinzip von Zerlegung großer Zahlen in ihre Primfaktoren.

Vereinfacht dargestellt wird beim Verschlüsseln die Nachricht mit dem *public* Key potenziert und mit Modulo durch eine zufällig gewählte Primzahl geteilt. Wird dieser Vorgang mit dem *private* Key nochmals ausgeführt, wird der unverschlüsselte Text zurückgeliefert. [27, S.508] Aktuelle Systeme setzen derzeit auf Schlüssel mit einer Mindestlänge von 2048 Bit. Das asymmetrische Verfahren dient nicht nur zur Kodierung von Daten, sondern wird auch für digitale Signaturen verwendet. Die RSA Methode wird ebenfalls als sicher betrachtet, sofern Schlüssellängen von mehr als 2048 Bit benutzt werden. Allerdings ist die asymmetrische Kryptografie deutlich langsamer als die symmetrische Verschlüsselung und Entschlüsselung. [28, S.3]
Da RSA mehrere Einsatzgebiete abdeckt und AES sehr sicher ist, wird ein kombinierter Einsatz empfohlen. Dabei übernimmt AES den Teil der Verschlüsselung von großen Datenmengen und RSA wird für Schlüsselmanagement und digitaler Signatur verwendet.
In der Blockchain ist die digitale Signatur ein zentraler Bestandteil, um die Sicherheit der Daten zu gewährleisten. Sie stellt sicher, dass gespeicherte Daten einer Geheimhaltung unterliegen und befolgt die Authentizität der Kryptografie. Daher wird die digitale Signatur im anschließenden Kapitel 2.3.2 behandelt.
Ein weiterer wichtiger Teil zur Gewährleistung der Richtigkeit der Daten in einer Blockchain stellt die kryptografische *Hash* Funktion dar. [11, S.7] Sie wird im Kapitel 2.3.3 erläutert.

2.3.2. Digitale Signatur

Die digitale Signatur basiert auf dem Prinzip der asymmetrischen Kryptografie und wird in einem nicht vertrauenswürdigen System zur Sicherstellung der Integrität der Daten eingesetzt. Sie stellt sicher, dass eine Person einen Inhalt bestätigt und die Daten seit der Erstellung unverändert bleiben. Anhand der Abbildung 2.10 wird das Prinzip näher erklärt. Der Prozess besteht aus zwei unterschiedlichen Phasen, die *signing* (Signatur) und die *verification* Phase (Verifikation). Jeder Benutzer einer Blockchain besitzt einen *public* und einen *private* Key. In der ersten Phase wird der *private* Key von Alice dafür verwendet, ihre Transaktion digital zu signieren. Dabei erstellt sie einen *Hash*, einen eindeutigen Fingerabdruck, aus dem Inhalt ihrer Daten und verschlüsselt diesen. Gemeinsam mit den originalen Daten und dem erstellten verschlüsselten *Hash*, schickt sie diese an Bob weiter. Diese Transaktion wird im Blockchain-Netzwerk öffentlich publiziert. Dadurch können die Transaktionen mit dem *public* Key von Alice, der öffentlich zugänglich ist, verifiziert werden. Grundvoraussetzung dafür ist, dass jeder seinen *public* Key publiziert und alle öffentlichen Schlüssel im gesamten Netzwerk gelesen werden können. [3, S.356]
In der zweiten Phase kann Bob den von Alice erstellten *Hash* mit dem *public* Key von Alice entschlüs-

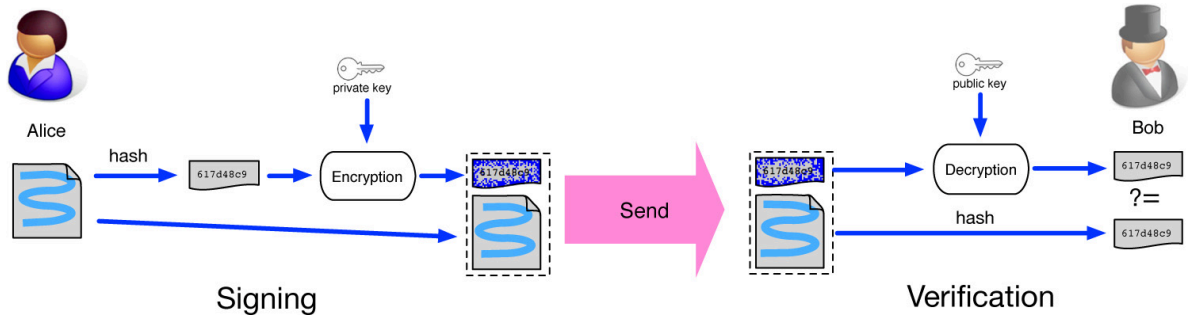


Abbildung 2.10.: Verifikation einer digitalen Signatur [3, S.356]

seln. Mittels Vergleiches des *Hash* der originalen Daten und des Entschlüsseln des *Hash* kann verifiziert werden, ob sie übereinstimmen. Ist dies der Fall, sind die Daten unverändert übertragen worden. Typischerweise wird für die digitale Signatur der (ECDSA) verwendet. [11, S.11] Dies steht für *elliptic curve digital signature algorithm*. Dieser Vorgang wird oft als *Public-key* Kryptografie bezeichnet. [11, S.49] Um eine Signatur für eine Transaktion zu erzeugen, wird bei ECDSA der *private* Key zusammen mit einem Zufallswert benutzt. Der Zufallswert muss für jede Signatur zufällig gewählt und nicht vorhersagbar sein. [12, S.4]

Zusammenfassend wird die digitale Signatur zum Schutz für Daten vor beabsichtigter oder durch Zufall passierter Änderungen verwendet. Die Vertraulichkeit und Integrität der Daten sind auch dann gewährleistet und wirksam, wenn sie in einem nicht vertrauensvollen Netzwerk abgelegt werden. [28, S.3] Des Weiteren wird durch den Einsatz von *public* und *private* Keys eine authentische Bestätigung des Eigentums von Daten sichergestellt. [4, S.1]

2.3.3. *Hash* Funktion

Eine wichtige Komponente der Blockchain ist die *Hash*-Funktion, die in vielen Operationen verwendet wird. [11, S.7] *Hash* wird zumeist als Kurzform von *Hash*-Wert eingesetzt. Ein *Hash*-Wert ist ein eindeutiger Fingerabdruck eines Dokuments, von Daten oder von Informationen. Dabei wird eine mathematische oder kryptografische Funktion zur Berechnung des *Hash* benutzt. Ändert sich diese nur minimal, z.B. in der kleinsten Form von einem Bit, stimmt der errechnete *Hash* nicht mehr mit dem *Hash* des ursprünglichen Dokuments überein. Des Weiteren ist der Hashwert unidirektional. Dies bedeutet, dass nicht vom *Hash*-Wert auf die Daten retour gerechnet werden kann, sondern nur von Daten auf den *Hash*, um ihn z.B. zu vergleichen. Wird ein *Hash* in einer Blockchain abgelegt, kann dieser nicht mehr verändert werden. Somit lässt sich die Echtheit von Informationen feststellen. Auf diese Art und Wei-

se kann nachgewiesen werden, dass Daten und Informationen zu einem bestimmten Zeitpunkt in einer bestimmten Version vorgelegen haben. [9, S.20] Der *Hash* selber ist eine berechnete, relativ eindeutige Zahl und wird oft als *Digest* bezeichnet. Eine besondere kryptografische *Hash*-Funktion, die sehr häufig bei Blockchains in Verwendung ist, ist der *Secure Hash Algorithm* (SHA). Dieser hat eine Ausgabelänge von 256 Bits und ist kurz auch als SHA-256 bekannt. 256 Bits entsprechen 32 Bytes (1 Byte = 8 Bits).

Nachfolgend sind die wichtigsten Sicherheitsmerkmale der kryptografischen *Hash*-Funktion aufgelistet:

1. **Widerstandsfähigkeit gegen Erstes-Urbild-Angriff *-pre-image attack*:** Nachdem die *Hash*-Funktion unidirektional ist, geht es bei dieser Angriffsmethode darum, zu einem bestehenden *Hash*-Wert (digest), das dazugehörige Urbild x, also den Klartext zu finden, wobei: hash(x) = digest ist. Dies stellt trotz rechenintensiver Leistung eine unlösbare Aufgabe dar. Die Wahrscheinlichkeit eine Übereinstimmung zu finden, liegt bei einer *Hash*-Länge von n bei 2^n Versuchen. [11, S.7]

2. **Widerstandsfähigkeit gegen Zweites-Urbild-Angriff *-second-pre-image attack*:** Wie der Name zweites Urbild bereits aussagt, wird hier versucht, zu einem *Hash* ein zweites Urbild zu finden, das genau denselben *Hash*-Wert aufweist. Beispielsweise muss das erste Urbild x mit dem hash(x) dem hash(y) gleich sein und y als Urbild muss erst noch gefunden werden. *Hash* Funktionen sind so entworfen, dass diese Aufgabe unlösbar bleibt. Somit lässt sich nur ein Suchen eines Urbilds mit einem Vergleich des *Hash*-Werts bewerkstelligen. Dies erscheint jedoch unmöglich. [11, S.7]

3. **Widerstandsfähigkeit gegen Kollisionsangriff *-collision attack*:** Ein Urbild x mit einem hash(x) und ein zweites Urbild y mit dem hash(y), wobei hash(x) = hash(y) entspricht, erscheint fast unmöglich. Die Wahrscheinlichkeit beträgt bei einer *Hash*-Länge von n $2^{n/2}$ Versuche. [11, S.7]

Derzeit gibt es 2^{256} gültige *Hash*-Werte. Um sich die Größe der Zahl besser vorstellen zu können, wird diese hier in voller Länge dargestellt:
115.792.089.237.316.195.423.570.985.008.687.907.853.269.984.665.640.564.039.457.584.007.913.129.639.936,00
Da diese Zahl fast unendlich erscheint, ist die Wahrscheinlichkeit, dass zwei Urbilder denselben *Hash*-Wert besitzen, sehr gering.

Der *Hash* wird mit 64 Zeichen als hexadezimale Zeichenkette dargestellt. Um ihn schneller berechnen zu können, ist dieser Algorithmus in den meisten Computern bereits als Hardware implementiert. [11, S.7] In der Tabelle 2.5 sind Beispiele für SHA-256 *Digest* als 64 Buchstaben lange Zeichenkette abgebildet.

Hello World!	7f83b1657ff1fc53b92dc18148a1d65dfc2d4b1fa3d677284addd200126d9069
Blockchain	625da44e4eaf58d61cf048d168aa6f5e492dea166d8bb54ec06c30de07db57e1

Tabelle 2.5.: Beispiele für Eingaben und ihre dazugehörigen SHA-256 *Hashes*

Neben der kryptografischen *Hash*-Funktion SHA-256, die als Standard in der Blockchain gilt, gibt es weiter Funktionen, wie z.B. RIPEMD-160 und Keccak, der als Gewinner zur Erstellung des SHA-3 hervor ging. RIPEMD-160 *Hash* steht für *RACE Integrity Primitives Evaluation Message Digest* und fand durch den kurzen *Hash*-Wert von 160 Bit den Einsatz bei der Adressierung von Blöcken, ohne auf die Sicherheit verzichten zu müssen.

Das Konsensverfahren *proof of work* benutzt ebenfalls die *Hash*-Funktion, um mittels eines kryptografischen *Nonce* einen *Hash*-Wert zu erstellen. Dabei handelt es sich um eine zufällig gewählte Zahl, die nur einmalig verwendet wird. Dieser *Nonce* wird, um unterschiedliche Hashwerte zu erhalten, mit den Daten als Produkt verwendet:

$$hash(data + nonce) = digest \tag{2.1}$$

Wie in der Formel 2.2 dargestellt, müssen *Hash*-Werte gefunden werden, die mit „000000“ beginnen. Diese sind notwendig, um einen Konsens zu finden. [11, S.9, S.20]

$$hash(„Blockchain“ + Nonce) = Hashwert \tag{2.2}$$

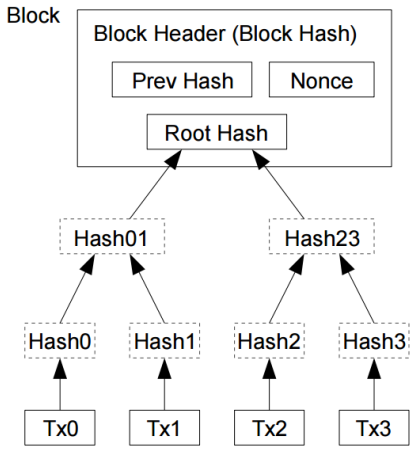


Abbildung 2.11.: Darstellung *Merkle* Baum [12, S.2]

Wie bereits in Kapitel 2.1.2 erwähnt, wird ein weiteres Signaturverfahren, der *Merkle tree*, (Merkle

Baum) verwendet. Entwickelt wurde dieser von Ralph Merkle, einem Pionier der kryptografischen Systeme. Er dient zur Verifikation der einzelnen Transaktionen in einem Block. Wie in der Darstellung 2.11 ersichtlich, wird für jede Transaktion *Tx0- Tx3* in einem Block ein *Hash*-Wert *Hash0- Hash3* generiert. Dieser beruht wieder auf dem Prinzip, dass er unidirektional ist. Anschließend wird für die erstellten *Hashes* paarweise ein *Hash*, *Hash01* und *Hash23* erzeugt. Den daraus resultierenden *Hash* nennt man *Root Hash*. Er repräsentiert die gesamte Struktur der Transaktionen als *Hash*.
Dieser *Hash* wird in dem Blockchain Block *Header* geschrieben. Da dieser unveränderbar ist, lässt sich so jede einzelne Transaktion dahingehend überprüfen, ob sie in einem speziellen Block vorhanden ist. [13, S.2102] Soll die Transkation von *Tx2* überprüft werden, wird lediglich der *Hash01* und *Hash3* benötigt. Mit *Tx2* und *Hash3* wird der rechte Zweig überprüft. Anschließend kann mit *Hash23* und *Hash01* der *Root Hash* verifiziert werden. [12, S.2]

2.3.4. Kryptografische Zugriffskontrolle

Wie bereits unter 2.2 angeführt, gibt es in einem verteilten System eine weitere Art der Zugriffskontrolle mit großer Bedeutung. Dabei handelt es sich um die *Cryptographic Access Control* (CAC). [25, S.19] Diese kommt unter anderem bei globalen Verbünden von Informationstechnologie (IT-Systemen) zum Einsatz und übernimmt die Aufgabe der Zugriffskontrolle, jedoch exklusive auf Kryptografie aufgebaut. Dadurch können die Vertraulichkeit und Integrität der Daten sichergestellt werden. Zusätzlich kann die CAC auch in nicht vertrauensvoller Umgebung, die das *public* Blockchain-Netzwerk darstellt, eingesetzt werden, und die Daten sind trotzdem sicher aufbewahrt. [28, S.1] Das CAC benötigt gegenüber anderen Zugriffskontroll-Modellen keinen Referenzmonitor. Daten werden direkt verschlüsselt in IT-Systemen abgelegt. Somit behandelt das System und der Datenspeicher nur verschlüsselte Informationen. Um zwischen Leseberechtigung und Schreibberechtigung zu unterscheiden, verwendet CAC einen asymmetrischen Algorithmus.
Daten werden mit dem *private key*, der auch *encryption key* genannt wird, verschlüsselt. Für die Entschlüsselung der Daten wird der *public key*, auch als *decryption key* bezeichnet, verwendet. Somit kann selektiv *read-access* (Leseberechtigung) und *write-access* (Schreibberechtigung) auf Daten vergeben werden, indem der *encryption key*, der *decryption key* oder beide weitergegeben werden. [28, S.2]
Bis jetzt wurde angenommen, dass ein asymmetrisches Verfahren schnell genug für Verschlüsselung und Entschlüsselung ist. Jedoch reicht die Geschwindigkeit nicht aus, um Daten in einem System zeitgerecht verschlüsselt abzuspeichern. Daher wird eine Kombination aus der symmetrischen und der asymmetrischen Kryptografie verwendet. Dabei werden Daten mit dem symmetrischen Verfahren verschlüsselt

sowie entschlüsselt und die asymmetrische Methode für Verifikation und digitaler Signatur eingesetzt. Somit wird die Integrität der Daten ebenfalls gewährleistet. Daraus ergibt sich folgender Unterschied zwischen Schreibvorgang und Lesezugriff:

- **Lesezugriff:** Clients benötigen zum Lesen der Informationen den symmetrischen Schlüssel, um Daten zu entschlüsseln und zusätzlich den *decryption key*, um die Signatur zu verifizieren, welche die Integrität der Daten sicherstellt.
- **Schreibzugriff:** Hingegen wird zum Schreiben der Daten der symmetrische Schlüssel zum Verschlüsseln und der *encryption key* zur Erstellung der Signatur benutzt.

Aufgrund der Implementierung von unterschiedlichen Operationen und der Grundlage, dass alle Daten immer verschlüsselt gespeichert werden, ist das System darauf ausgelegt, in einem öffentlichen unsicheren Netzwerk zu funktionieren. Somit können Zugriffskontrollen ebenfalls in einer dezentralen Umgebung sichergestellt werden. Dadurch kann bei der kryptografischen Zugriffskontrolle in einem dezentralen Netzwerk vollständig auf einen Referenzmonitor, der eine zentrale Komponente darstellt, verzichtet werden. Zugriffskontrollen werden direkt am Client durchgeführt. [28, S.7]

2.4. Sicherheit von Daten in Cloud-Speicher

Dieses Kapitel behandelt die Sicherheit von Daten in einem Cloud-Speicher sowie in mobilen Geräten. Diese Methoden können die Sicherheit der Blockchain weiter erhöhen. Des Weiteren wird *Trusted Computing* als möglicher Prozess für sichere Verarbeitung bei Zugriffen auf Daten eingesetzt.

2.4.1. Zugriffskontrolle für mobile Geräte

Die große Beliebtheit von *Cloud Computing* und *Smartphones* bedarf einer zusätzlichen Sicherheit, um persönliche Informationen erfolgreich zu schützen. Wie in [29] [30] von Piller et al. besprochen, wurde eine neue kryptografische Zugriffskontrolle entworfen, um mobile Geräte zu sichern. Bei diesem Verfahren wurden die auf mobilen Geräten eingeschränkte Verarbeitungsgeschwindigkeit, Speichervolumen und Datensicherung berücksichtigt. Des Weiteren kann auf Grund der Dezentralität von mobilen Geräten oder von Cloud-Speicher, nicht auf die herkömmliche Zugriffskontrolle zurückgegriffen werden, um Daten ausreichend zu schützen. Die neue Methode „Cryptographic Access Control Solution for Smart Phones“ (CASSP) beachtet die Handhabung der dezentralen Verarbeitung und beinhaltet ein umfangreiches Konzept und ein Schlüsselmanagement-System. Sie passiert auf Basis einer kryptografischen

Zugriffskontrolle und dem Prinzip der Bunimov-Methode [31] zur Optimierung der modularen Multiplikation bei sehr großen Zahlen, dem *Rabin-Kryptosystem*. [32] Für die Sicherung der kryptografischen Schlüssel dient das Shamir-Secret-Sharing [33] Schema für (k, n). Zusätzlich ist diese Methode ressourcenschonend, da eine Neuverschlüsselung von Daten bei Änderung von Zugriffsrechten nicht notwendig ist. Dies stellt bei mobilen Geräten eine große Herausforderung dar. [30, S.5] Zusätzlich besticht das neue Verfahren bei der Verarbeitungsgeschwindigkeit beim Schlüsselmanagement und bei der Sicherung der geheimen kryptografischen Schlüssel. CASSP verhindert nicht das Löschen oder Zerstören der Daten gesamtheitlich. Das muss vom jeweiligen mobilen Gerät oder Cloud-Anbieter sichergestellt werden. Das System schützt jedoch den Inhalt der Daten mittels kryptografischer Methode. Diese beruht auf dem Prinzip von geheimen Primzahlen, *p* und *q* und dem Wert $n = p \cdot q$. Die Zahl n und eine zufällig gewählte Zahl z dienen als öffentlicher Schlüssel (z, n).
Durch Quadrieren mittels $z_{i-1} = {z_1}^2 \pmod n$ wenn, $i > 2$, kann ein Schlüssel retour gerechnet werden. Dabei sind die geheimen Teile p und q nicht erfoderlich.
Mittels modularer Quadratwurzel von $z_i \pmod n$, wobei $n = p \cdot q$ ist, können neue Schlüssel z_{i+1} berechnet werden. CASSP stellt dabei sicher, dass niemand unberechtigt auf Daten zugreifen oder sie verändern kann. Die Neuverschlüsslung von großen Datenmengen stellt jedoch auf mobilen Geräten eine Herausforderung dar. [30, S.7]
Diese kryptografischen Methoden und die dazugehörigen Verfahren werden in der empirischen Untersuchung auf die Anwendbarkeit für eine Blockchain im Detail betrachtet.

2.4.2. Einsatz von *Trusted Computing*

Um ein Cloud-Service zu benutzen, stellt das Vertrauen in die Cloud derzeit das größte Problem der Konsumenten dar. Zumeist wird nicht eine reine Cloud-Lösung, sondern eine Hybridvariante mit unterschiedlichen Systemen, die in der Cloud und on-premis benutzt werden, verwendet. Dadurch wird es kompliziert, die Sicherheit in der Cloud zu gewährleisten. Um die sensiblen Daten bei *Cloud Computing* abzusichern, wird zusätzlich zur kryptografischen Zugriffskontrolle eine *Trusted Computing Platform* (TCP) integriert. Diese stellt sicher, dass nur autorisierte Zugriffe auf die Infrastruktur und die Daten möglich sind. Dabei darf der Komfort der Cloud nicht auf Kosten der Sicherheit abhandenkommen. Das *Trusted Platform Module* (TPM) stellt diese Möglichkeit bereit. Die Integration der Sicherheitsmechanismen wird daher in die *Core*-Infrastruktur eingefügt, statt diese als zusätzliche Anwendung zu implementieren. Dies hat zum Vorteil, dass es nicht einfach anzugreifen ist und das System von den sensiblen Daten getrennt betrieben wird. [34, S.942]
Das *Trusted Computing*-System riegelt die Teile des Computers, die sensible Daten verarbeiten, in einem

sicheren Bereich mittels Kryptografie ab. Greift ein Programm auf diese Daten zu, vergibt das System einen *decryption key* an Anwendungen, die als vertrauenswürdig eingestuft sind. Das Kriterium, welche Funktion vertrauenswürdig ist, folgend wird definiert: „Eine vertrauensvolle Komponente, Operation oder Prozess bei dem in allen Lagen der Operation das Verhalten so vorhersagbar ist, dass die Operation auch nicht, z.B. durch einen Virus, beeinflusst werden kann.“ Die Funktion, der das Betriebssystem des Computers immer vertraut, wird *Roots of Trust* genannt. [34, S.943] Dieser Prozess wird von den meisten Herstellern als eigene Hardware in Computern implementiert.

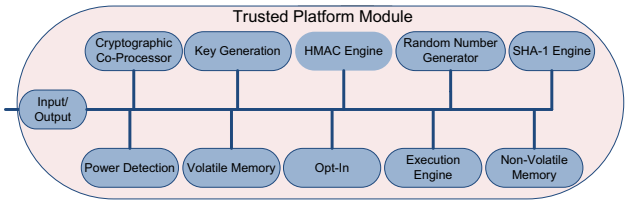


Abbildung 2.12.: *Trusted Platform Module*-Architektur [35, S.887]

Wie in Abbildung 2.12 ersichtlich, werden diese mit Funktionen wie Zufallsgenerator, *SHA hashing*, asymmetrische sowie symmetrische Verschlüsselung und Entschlüsselung, Signatur und Verifikation mittels 2048 Bit RSA und asymmetrische Schlüsselerzeugung ausgestattet. [35, S.887]

Da bei *Cloud Computing* zumeist eine große Anzahl an Benutzern mitwirkt, benötigt *Trusted Computing* eine Möglichkeit, den Zugriff zu steuern. Dies wird, wie bereits in Kapitel 2.2 besprochen, mittels *role-based access control* durchgeführt. Der Zugriff wird für Benutzer mittels definierter Kriterien in Klassen oder Gruppen eingeteilt. Dabei wird einem User mittels persönlicher ID ein *secret key* passend zu einer Rolle und einer zugeteilten Rolle Zugriff gewährt. [36, S.13]
Ein weiterer wichtiger Mechanismus ist die Verschlüsselung von Daten. Diese werden so verschlüsselt, dass nur bestimmte Maschinen sie entschlüsseln können. Dies passiert nur, wenn die Daten in einer gewissen Gruppierung vorhanden sind. Dabei wir ein „master key“ für jede Maschine erstellt. Dieser wird verwendet, um „sub-key“ für eine weitere Gruppierung dieser Maschine zu erstellen. Somit muss die Maschine, um Daten entschlüsseln zu können, in der richtigen Gruppierung sein. Für die Kommunikation wird anschließend der „sub-key“ eingesetzt, der einen „session-key“ erzeugt. Der„session-key“ gilt nur, solange die Maschinen in derselben Gruppierung arbeiten. Dadurch, dass diese zufällig generierten Schlüssel mit der Hardware erzeugt werden, gelten sie gegen Angriffe besonders sicher und stellen eine sichere Kommunikation bereit. Will ein Benutzer oder ein Programm auf Daten zugreifen, muss zuerst die Authentifikation durchgeführt werden. Danach können mit dem sicher im TPM abgelegt Schlüssel Daten verschlüsselt werden. Diese Funktionen können in einer Hybridvariante sicher eingesetzt werden.

[36, S.14]

Des Weiteren kann die *Trusted Computing* Plattform für den Einsatz von *Europay-Visa-Mastercard* (EMV) Kartenzahlung benutzt werden. Bei der EMV Chip-Technologie wird die Authentifikation der Karte, zur Sicherung von Transkationen bei Kreditkarten und Bankomatkarten verwendet. Zur Erhöhung der Sicherheit wird vom Kunden zusätzlich eine PIN-Abfrage für die Transaktion benötigt. Dieses Verfahren wird durch *Static Data Authentication* (SDA) und *Dynamic Data Authentication* (DDA) Karten oder eine Kombination aus beiden Verfahren, dem *Combined Dynamic Data Authentication* (CDA), unterstützt. SDA verwendet symmetrische Schlüssel, die aber nicht sicher vor Fälschung von Karten sind. Das DDA Verfahren hingegen verwendet asymmetrische Schlüssel und ist somit sicher gegen Klonen von Karten. Dabei wird auf die Schlüsselverteilung durch Zertifikatsketten zurückgegriffen. EMV stellt dabei einen sehr hohen Sicherheitsstandard dar. [37, S.1]

2.5. Europäische Datenschutz-Grundverordnung

Um die Europäische Datenschutz-Grundverordnung (DSGVO), in Englisch *general data protection regulation* (GDPR) genannt, im Zusammenhang mit Blockchain besser zu verstehen, behandelt dieses Kapitel die wichtigsten Faktoren und fasst die für den Einsatz einer Blockchain wesentlichen Themen zusammen.

2.5.1. Grundsätze

In der heutigen Zeit ist es selbstverständlich, dass Daten im Internet abgelegt und veröffentlich werden. Die wenigsten Leute haben Einfluss darauf, was mit ihren Daten passiert, wie diese abgespeichert oder ob sie an Dritte weitergegeben werden. Die DSGVO regelt die Verarbeitung von personenbezogen Daten. Sie soll Transparenz und die Kontrolle über die eigenen Daten erleichtern. Bei personenbezogen Daten handelt es sich um Daten von natürlichen Personen. Im Speziellen können das Kreditkartendaten, eine Sozialversicherungsnummer und E-Mail-Adressen sowie eine Wohnadresse oder ein Geburtsdatum sein. [38, S.5] Zusätzlich regelt sie die Verarbeitung von sensiblen Daten. Dazu zählen z.B. genetische und biometrische Daten zur eindeutigen Identifizierung einer natürlichen Person, wie Fingerabdruck und Iris sowie Gesundheitsdaten und religiöse Ansichten.

In der DSGVO wird festgelegt, welche Bedingungen erfüllt werden müssen, um Daten zu verarbeiten:

- **Einwilligung:** Betroffene Personen haben die Einwilligung auf Verarbeitung der personenbezogenen Daten zu einem bestimmten Zweck gegeben.

- **Vertrag:** Die Datenverarbeitung ist für die Erfüllung eines Vertrages zwischen einer Partei und der betroffenen Person erforderlich.
- **Gesetz:** Die Verarbeitung der Daten ist zur Erfüllung von gesetzlichen Verpflichtungen notwendig.
- **Berechtigtes Interesse:** Bei diesem Begriff handelt es sich um einen unbestimmten Rechtsbegriff. Die Verarbeitung der Daten muss individuell ermittelt werden. Die Anmeldung für einen Newsletter kann z.B. ein berechtigtes Interesse sein, sofern der Schutz von personenbezogenen Daten überwiegt. [5, Art. 6]

Die Verarbeitung der Daten darf nur aus einem bestimmten Zweck erfolgen und muss auch für diesen Zweck auf ein Minimum eingeschränkt werden.

2.5.2. Recht der betroffenen Personen

Wie in Artikel 13 der DSGVO [5] beschrieben, hat jede betroffene Person das „Recht, die Löschung" der Daten bei den Verantwortlichen zu beantragen. Dies wird häufig „Recht auf Vergessenwerden" genannt. Die Verantwortlichen sind verpflichtet, die Daten unverzüglich zu löschen, sofern eine der folgenden Gründe vorliegt:

- „Die personenbezogenen Daten sind für die Zwecke, für die sie erhoben oder auf sonstige Weise verarbeitet wurden, nicht mehr notwendig." [5, Art. 17]
- Die betroffene Person widerruft die Einwilligung, damit entfällt der Grund diese Daten zu verarbeiten. [5, Art. 17]
- Die zweckgebundene Notwendigkeit, die Daten aufzuheben, fällt weg. [5, Art. 17]
- „Die personenbezogenen Daten wurden unrechtmäßig verarbeitet." [5, Art. 17]
- Hingegen kann eine Löschung untersagt werden, sofern aus rechtlicher Verpflichtung notwendig ist, diese aufzubewahren. [5, Art. 17]

Des Weiteren hat jede Person das „Recht auf Berichtigung" der Daten sofern:

- diese gegen das Recht, dem der Verantwortliche unterliegt, verstößt; [5, Art. 16]
- die Daten hinsichtlich des Zwecks, für die sie erhoben wurden, nicht mehr benötigt werden; [5, Art. 16]
- sie unrichtige personenbezogene Daten beinhalten; [5, Art. 16]

- die Daten unvollständig sind. [5, Art. 16]

Nachfolgend werden die Informationspflicht und das Auskunftsrecht behandelt.

2.5.3. Informationspflicht und Auskunftsrecht

Bei der Verarbeitung von Daten sind die Verantwortlichen verpflichtet, den betroffenen Personen Informationen über ihre Daten zu gewähren. Dabei müssen sie unter anderem Auskunft über folgende Informationen geben: [5, Art. 13] [5, Art. 15]

- den Verarbeitungszweck; [5, Art. 15]
- „die Kategorien personenbezogener Daten, die verarbeitet werden;“ [5, Art. 15]
- „falls möglich die geplante Dauer, für die die personenbezogenen Daten gespeichert werden oder, falls dies nicht möglich ist, die Kriterien für die Festlegung dieser Dauer.“ [5, Art. 15]

2.5.4. Verantwortlicher und Auftragsverarbeiter

Bei der DSGVO wird bei der Verarbeitung von Daten zwischen einer betroffenen Person, einem Verantwortlichen und einem Auftragsverarbeiter unterschieden.

Der Verantwortliche trägt dafür Sorge, dass die Verarbeitung der Daten betroffener Personen gemäß der Verordnung erfolgt. Um dies sicherzustellen, verwenden sie technische oder organisatorische Maßnahmen. Zur Verarbeitung von Daten zählt z.B. das Erfassen, die Speicherung, die Veränderung oder Anpassung, sowie das Abfragen, die Verwendung und die Löschung von personenbezogenen Daten. [5, Art. 24]

Der Auftragsverarbeiter verarbeitet die Daten nur im Auftrag eines Verantwortlichen. Darunter fallen z.B. externe Dienstleister, wie Cloud-Anbieter oder IT-Dienstleister. [5, Art. 28]

3. Empirisches Verfahren

Die Erkenntnisse aus der theoretischen Analyse bilden die Grundlage für den empirischen Teil dieser Diplomarbeit. Im folgenden Kapitel wird auf das Ziel der empirischen Untersuchung und das methodische Vorgehen eingegangen. Darüberhinaus wird ein Einblick in das Vergaberecht auf Zugriffe in einer Blockchain und das Schlüsselmanagement auf Basis von Kryptografie sowie die neuartigen Konsensverfahren mittels kryptografischem Zugriffskontrollsystem und die Implementierung von kryptografischen Zugriffskontrollsystemen in Blockchains gewährt. Abschließend wird unter Kapitel 3.8 die Einhaltung und Erfüllung der DSGVO überprüft.

3.1. Ziel der empirischen Untersuchung

Die empirische Untersuchung dient zur Beantwortung der Forschungsfragen aus Kapitel 1.4. Dabei werden unter Berücksichtigung der aktuellen Technologien neue Konsensverfahren entwickelt, die den Einsatz von kryptografischer Zugriffskontrolle für Blockchains ermöglichen. Diese bedienen sich ausschließlich kryptografischer Methoden und benötigen dadurch keine zusätzlichen dritten Beteiligten, wie z.B. ein Betriebssystem, ein Datenbanksystem oder Anwendungen für den Leseschutz und Schreibschutz von Daten. In dieser Untersuchung wird vollständig auf die herkömmliche zentrale Zugriffskontrolle verzichtet, da sie nicht der Philosophie einer Blockchain entspricht. Bei der empirischen Untersuchung wird die zugrundeliegende Technologie und Funktionalität der Blockchain berücksichtigt. Zusätzlich werden, wie in den Forschungsfragen definiert, die Vorgaben der DSGVO erfüllt. Dies soll dazu beitragen, die *Blockchain for Data Security* bestmöglich für alle Anwendungsbereiche auszustatten.
Da Cloud Speicher und mobile Geräte wie eine Blockchain dezentral funktionieren, werden die unter Kapitel 2.4.1 beschriebenen Verfahren als Basis für das Schlüsselmanagement und für die neuen Konsensverfahren gewählt. Dabei wird überprüft, ob sie für eine Blockchain geeignet sind. Des Weiteren ist es notwendig neue Nodes zu schaffen, welche die Richtlinien der DSGVO erfüllen. Diese sollen ebenfalls die Philosophie der Blockchain verfolgen und keine zentrale Stelle bilden. In der empirischen Untersuchung werden diese Methoden dahingehend überprüft, ob ein vollständiger Einsatz bei Blockchains möglich ist. Das daraus entstandene Framework stellt die neue Basis der *Blockchain for Data*

Security dar.

3.2. Methodisches Vorgehen

Auf Grund der fehlenden Technologien für sichere Blockchains und der gewonnenen Erkenntnis aus der Literaturanalyse wird das in Kapitel 2.4.1 beschriebene Verfahren herangezogen und dessen Anwendbarkeit in einer Blockchain für das Schlüsselmanagement sowie für Leseschutz und Schreibschutz überprüft. Dabei wird mittels qualitativer Analyse betrachtet, ob und in welcher Art und Weise das Verfahren für eine Blockchain anwendbar ist. Die beschriebenen Theorien werden anhand der in dieser Arbeit durchgeführten Forschung überprüft.

3.3. Anforderung an *Blockchain for Data Security*

Um Anforderungen für die *Blockchain for Data Security* zu definieren, ist es notwendig, die internationalen Standards im Bereich der Informationssicherheit zu erfüllen. Diese sind unter anderem im „IT-Grundschutz vom Bundesamt für Sicherheit in der Informationstechnik“ (BSI), der ISO/IEC 27001-Norm, der *National Institute of Standards and Technology* (NIST) und *Federal Information Processing Standards* (FIPS) festgelegt. Das Ziel der Informationssicherheit ist, die Vertraulichkeit, die Verfügbarkeit und die Integrität der Daten sicherzustellen. Somit muss auch die Cyber-Security als Schlüsselelement in eine Blockchain implementiert werden.
Für die Anforderungen der *Blockchain for Data Security*, als Basis der IKT-Landschaften, wird die Beschaffungsplattform it-sicher.kaufen, welche in dem Sicherheitsforschungs-Förderprogramm KIRAS mit der Fachhochschule St. Pölten und dem Bundesministerium für Verkehr, Innovation und Technologie (BMVIT) entwickelt wurde, herangezogen. Die Plattform unterstützt dabei, IT-sicherheitsrelevante Fragen gegen Angriffe und Bedrohungen aus dem Cyber-Raum zu beantworten. Die dort definierten Schwerpunkte bilden die Grundlage für das in dieser Arbeit erstellte Blockchain-System. Nachfolgend sind die Anforderungen für die *Blockchain for Data Security* angeführt:

- **Zugriffskontrolle:** Um zu gewährleisten, dass nur berechtigte Benutzer Zugriff zum Blockchain-System haben, ist eine Feststellung der Identität eines Benutzers notwendig. Dabei muss die Dezentralität eine Blockchain berücksichtigt werden und es darf keine zentrale Autorität zur Verifizierung zum Einsatz kommen. Zur Authentifizierung des Benutzers werden kryptografischer Schlüssel verwendet. Sie dürfen nur für die in ihrer Rolle notwendigen Informationen autorisiert sein. Dies kann durch unterschiedliche Schlüssel erzielt werden. Dadurch ist die Informationsweitergabe nur für Berechtigte sichergestellt.

- **Verschlüsselung:** In der *Blockchain for Data Security* müssen alle Daten verschlüsselt abgelegt werden. Wie unter Kapitel 2.3.1 bereits erwähnt, muss ein bewährtes asymmetrisches oder symmetrisches Verfahren eingesetzt werden. Dadurch kann ein sicheres Aufbewahren der Daten gewährleistet werden. Der Einsatz einer digitalen Signatur, wie in Kapitel 2.3.2 beschrieben, ist dabei unverzichtbar. Nur so kann überprüft werden, ob Daten unverändert und von einer bestimmten Person stammen. Die Verschlüsselungsschlüssel und Entschlüsselungsschlüssel sind so zu wählen, dass sie der aktuellen IT-Sicherheit entsprechen.

- **Schlüsselmanagement:** Das Schlüsselmanagement darf nicht von einer zentralen Stelle verwaltet werden. Schlüsselteile werden auf viele Beteiligte n aufgeteilt, somit ist eine böswillige Veränderung von Schlüsseln ausgeschlossen. Zusätzlich muss ein Schwellenwert k definiert sein. Der Schwellenwert selbst wiederum definiert, dass mindestens diese Anzahl an Teilschlüsselbesitzer im Stande ist, diesen wiederherzustellen oder einen neuen Schlüssel auszustellen. Eine weitere Anforderung ist das Widerrufen von alten Schlüsseln. Das Schlüsselmanagement wird ausführlich unter Kapitel 3.5 behandelt.

- **Protokollierung:** Um in einem IT-System für Audits alle Vorgänge rückwirkend zu untersuchen, müssen sie innerhalb der Blockchain protokolliert werden. Dabei sind Log-Daten vor unerlaubten Zugriffen und Änderungen zu schützen. Kritische Informationen, wie z.B. geheime Schlüssel, dürfen dabei nicht protokolliert werden.

- **Datenschutz:** Der Schutz Benutzerdaten und die Einhaltung der Privatsphäre hat oberste Priorität. Um dies in einer Blockchain zu gewährleisten, müssen alle Daten in einer verschlüsselten Form vorliegen. Nur so ist der Einsatz von kryptografischen Methoden für den Schutz der Daten möglich. Daher sind alle Daten zu verschlüsseln und nur berechtigte Personen dürfen auf diese zugreifen. Somit kann sichergestellt werden, dass beim Löschen von Schlüssel ein Zugriff auf Daten nicht mehr möglich ist. Zusätzlich muss die Verarbeitung der Daten zweckgebunden sein.

- **Auditierung:** Die Auditierung der Daten ist bereits durch die Philosophie der Blockchain erfüllt. Daten können nicht gelöscht werden. Somit ist sichergestellt, dass alle Informationen für ein Audit fälschungssicher vorhanden sind.

- ***Code review*:** Um sicherzustellen, dass die Verarbeitung der Daten und die Funktionen in einer Blockchain korrekt verarbeitet werden, ist es notwendig, einen *Code review* durchzuführen. Damit kann gewährleistet werden, dass Fehler und *Bugs* gefunden werden. Dabei ist der *Software Development Life Cycle* einzuhalten. Dieser beinhaltet zunächst das Analysieren und Erfassen aller

Anforderungen. Danach werden Konzepte erstellt und diese in der Implementierung umgesetzt. Abschließend sind Tests durchzuführen und die Blockchain zum Einsatz zu bringen. Eingesetzte *smart contracts* sind ebenfalls einem *Code review* zu unterziehen.

Diese definierten Mindestanforderungen können in der „Europäischen Agentur für Netz- und Informationssicherheit“ (ENISA) als Rahmenwerk für die *Blockchain for Data Security* festgehalten werden.

3.4. Vergaberecht auf Zugriffe in einer Blockchain

In der Literaturanalyse wurde auf die Schwierigkeit der Vergabe von Zugriffen in einer Blockchain hingewiesen. Auf Grund der Dezentralität ist es problematisch, Berechtigungen auf verteilten Systemen zu verwalten. Im nachfolgenden Kapitel wird auf die Basis für eine kryptografische Zugriffskontrolle durch Definieren neuartiger *Nodes* eingegangen. Diese sind Teil der Zugriffsteuerung und regulieren den Zugriff zu kritischen und wertvollen Informationen, wie Transaktionen oder jeglicher Form, welche in einer Blockchain gespeichert sind. Zuerst werden im Kapitel 3.4.1 Definitionen für *Nodes* beschrieben, anschließend wird unter Kapitel 3.4.2 auf die Festlegung der *Nodes* eingegangen.

3.4.1. Definition neuer Nodes

Wie in Kapitel 2.1.4 bereits besprochen, gibt es unterschiedliche *Nodes* in einer Blockchain. Sie erfüllen mit unterschiedlichen Konsensverfahren jene Anforderungen, um Blöcke in eine Blockchain durch *Mining* hinzuzufügen. Allerdings ist die derzeitige Blockchain nicht in der Lage, die DSGVO zu befolgen. Es besteht hier keine Möglichkeit, der Informationspflicht nachzukommen oder Kunden Auskunft zu geben. Des Weiteren kann das Löschen von Daten derzeit nicht durchgeführt werden. Das Löschen von Daten in einer Blockchain wird im herkömmlichen Sinn aus heutiger Sicht nie möglich sein. Jedoch muss ein Prozess geschaffen werden, der von einem *Node* durchgeführt und eingehalten werden kann. Dieser wird in den nachfolgenden Definitionen erläutert. Die Definition neuer *Nodes* und die Erweiterung ihrer Funktionen ist notwendig, um den Einsatz kryptografischer Zugriffskontrollsysteme zu ermöglichen.

Nachfolgend sind Handlungsempfehlungen für neuartige *Nodes* mit ihren zusätzlichen Aufgaben definiert:

- ***Definition 1*: *Nodes*** müssen weiterhin unabhängig von einer dritten Partei sein, um unbeeinflusst ihre Aufgabe, das *Mining* von Blöcken, durchführen zu können. Dadurch werden Transaktionen auf ihre Richtigkeit geprüft. Sie können in den unterschiedlichen Typen wie *public*, Konsortium

und *private* Blockchain zum Einsatz kommen. Ähnlich wie in einer Konsortium-Blockchain werden sie durch spezielle Kriterien bestimmt oder gewählt.
Die Bestimmung der *Nodes* wird anschließend beschrieben.

- ***Definition 2*:** *Nodes* müssen zusätzlich für die regulatorischen Aufgabe bzw. als Controller zur Erfüllung der DSGVO bereitstehen. Somit können sie das „Recht auf Auskunft“ übernehmen und damit die Informationspflicht laut der Datenschutz-Grundverordnung erfüllen. Das kann mittels Automatisierung durch den Einsatz von *smart contracts* erreicht werden.

- ***Definition 3*:** *Nodes* werden durch die Anwendung von kryptografischen Zugriffskontrollen mit öffentlichem und geheimem Schlüssel zum Verschlüsseln, Entschlüsseln und Signieren der Daten verwendet. Sie nutzen diese Möglichkeiten, um Daten auf ihre Richtigkeit zu überprüfen. Beim *Mining* kommt ein Konsensverfahren, welches im Kapitel 3.6 beschrieben wird, zum Einsatz.

- ***Definition 4*:** Hinzu kommt das Verteilen von kryptografischen Schlüsseln als erweiterte Aufgabe. Die Erstellung der Schlüssel wird separat durchgeführt und beschränkt sich nicht auf einzelne *Nodes*. Somit wird gewährleistet, dass nur beteiligte *Nodes* einen neuen Schlüssel erstellen können. Das Schlüsselmanagement wird unter Kapitel 3.5 behandelt.

- ***Definition 5*:** Da in einer Blockchain das Löschen von Daten im herkömmlichen Sinn nicht möglich ist, muss ein *Node* diese Aufgabe mit anderen Mitteln übernehmen. In einer *Blockchain of Data Security* sind bereits alle Daten mit einem geheimen Schlüssel (z,n) verschlüsselt. Dabei ist z eine Zufallszahl und n das Produkt aus p und q, welche Primzahlen darstellen. Ein *Node* hat, wie in der Definition 2. beschrieben, die Aufgabe der Informationspflicht. Zusätzlich müssen sie bei einer Anfrage auf Löschung von Daten alle Beteiligten informieren, dass dieser geheime Schlüssel zu löschen ist. Somit kann sichergestellt werden, dass niemand auf diese Daten Zugriff hat und somit die DSGVO erfüllt wird. Im Bedarfsfall sind zusätzlich neue geheime Schlüssel zu generieren. Dieses Verfahren wird unter Kapitel 3.6.1 näher erklärt.

Unter der Voraussetzung, dass die *Nodes* die oben angeführten Definitionen erfüllen, werden sie per Definition als *Service Nodes* bezeichnet. Sie gewährleisten den Einsatz der Blockchain in unterschiedlichen Anwendungsbereichen. Dadurch muss auch die Wahl der *Nodes* anhand des Einsatzgebietes einer Blockchain unterschiedlich durchgeführt werden, ohne dabei die Philosophie einer dezentralen Blockchain zu vernachlässigen. Durch den Einsatz einer kryptografischen Zugriffskontrolle wird generell keine Computer-Power benötigt, da die Kryptografie trotz ihrer Komplexität einfach berechnet werden kann.
Durch ihre neue Aufgabe kommen der *Service Nodes* eine große Bedeutung und auch viel Entscheidungsfreiheit zu. Dieses Vertrauen an *Service Nodes* darf nicht von ihnen missbraucht werden. Sollte

dies der Fall sein, verlieren sie den Auftrag zum *Mining* von Blöcken und werden ersetzt. Sie besitzen die Datenhoheit über das Schlüsselmanagement und das Vergaberecht der Schlüsselberechtigungen. Wenn sie nicht mehr den Definitionen entsprechen, können sie von unabhängigen Gremien oder Konsortien neu gewählt werden. Somit ist ihre Rolle nur eine befristet übertragene Aufgabe. Weist ein *Service Node* ein Fehlverhalten auf, kann ihm der Schlüssel entzogen werden. Die daraus resultierende Rolle der *Service Nodes* entspricht dem dezentralen Verwalten der Daten in einer Blockchain.
Im nächsten Kapitel wird die Festlegung der *Service Nodes* besprochen.

3.4.2. Auswahl von Service Nodes

Nachdem im letzten Kapitel *Service Nodes* definiert wurden, wird hier auf deren Auswahl eingegangen. Dabei werden die unterschiedlichen Typen wie *public*, Konsortium und *private* Blockchain berücksichtigt. Durch ein Auswahlkriterium werden *Service Nodes* definiert und damit Berechtigungen gesteuert. Somit benötigt jede Art einer Blockchain ihr eigenes Kriterium. Dies können z.B. Regulatoren, Konsortien und E-Government sowie Behörden, Bankenaufsicht aber auch Kunden-Lieferanten-Beziehungen sein. Wie in der Darstellung 3.1 ersichtlich, werden die *Service Nodes* von außerhalb der Blockchain gesteuert. Die steuernde Einheit wird hier als Auditoren bezeichnet.

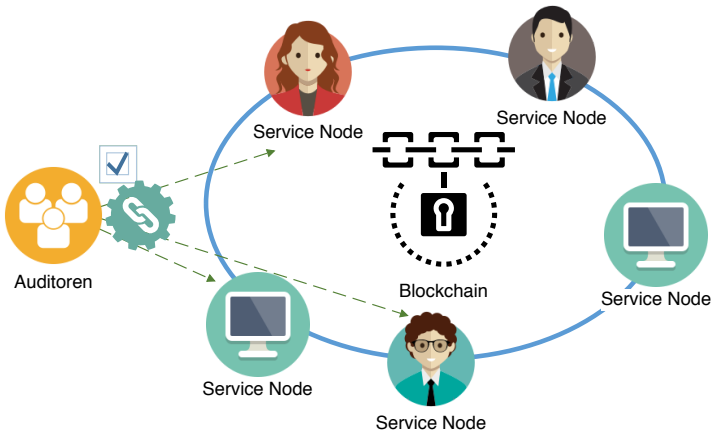


Abbildung 3.1.: Eigene Darstellung, Übersicht *Service Nodes*

Die Auditoren wählen anhand von Kriterien für das Einsatzgebiet der Blockchain die gewünschten *Service Nodes* aus und berechtigten sie, neue Schlüssel zu erstellen oder Schlüssel zu widerrufen. Die verliehene Rolle als *Service Node* kann jederzeit widerrufen werden. Sie können dadurch ihrer herkömmlichen Aufgabe als *Node* nachgehen und Blöcke eintragen, eingetragene Blöcke auf ihre Richtigkeit validieren

oder ablehnen.
In der Tabelle 3.1 sind Beispiele mit bestimmten Einsatzgebieten und Kriterien angeführt, anhand derer die Auditoren spezielle *Service Nodes* auswählen.

Art der Blockchain	**Kriterium**	**Beschreibung**
Gesundheitswesen	Ärzteverzeichnis	In einem Ärzteverzeichnis obliegen die eingetragenen Ärzte bereits einem Verfahren, das bestätigt, dass diese vertrauensvoll sind. Somit sind ausgewählte Ärzte die *Service Nodes*. Sobald ein Patient seine E-Card beim Arzt vorzeigt, bekommt dieser die Berechtigung Patientendaten zu lesen und zu schreiben.
Finanzbereich	Bankenaufsicht	Wird eine Blockchain für den Finanzbereich verwendet, müssen die zuständigen Bankenaufsichten die Aufgabe der *Service Nodes* übernehmen.
Notenvergabe	Lehrbeauftragte	Kommt eine Blockchain im Bereich Notenvergabe zum Einsatz, können Lehrbeauftragte, die in einem vertraglichen Verhältnis mit einer Fakultät stehen, als vertrauenswürdig eingestuft werden. Sie übernehmen die Aufgabe der *Service Nodes*.
Lieferketten	Lieferanten	In einer *Supply-Chain*-Blockchain stehen Kunden und Lieferanten in einer Beziehung. Somit werden hier Service Nodes anhand von Lieferanten gewählt.
Dateiablage	Besitzer einerDatei	Ein *Service Node* muss nicht zwingendermaßen eine Hierarchie darstellen. Betrachtet man eine Blockchain, die Daten und Informationen beinhaltet, kann dieser auch aus einer einzelnen Person bestehen. Somit fungiert der Besitzer der Datei als *Service Node* seiner Daten.
Daten	Inhaber von Informationen	Ein Inhaber von Daten aller Arten kann ebenfalls ein *Service Node* sein. Hier besteht eine flache Hierarchie. Somit bestimmt der Dateninhaber, wer seine Informationen lesen darf.

Tabelle 3.1.: Eigene Tabelle, Übersicht Auswahl Service Nodes

Alle *Service Nodes* derselben Kategorie können somit Blöcke eintragen und bestätigen. Diese Beispiele lassen sich auf alle Einsatzgebiete einer Blockchain ausweiten und anwenden. Auch wenn *Service Nodes* durch Kriterien ausgewählt werden, stellen sie normale Teilnehmer im Blockchain-Netzwerk dar, welche besondere Berechtigungen erhalten. Somit werden *Service Nodes* ausgewählt, die mittels Schlüsselmanagement berechtigt werden, Blöcke in einer Blockchain einzufügen. Dabei muss sichergestellt werden, dass nicht einzelne *Nodes* neue Schlüssel erzeugen können. Die *Service Nodes* verteilen die Schlüssel zum Lesen der Daten.

3.5. Schlüsselmanagement auf Basis von Kryptografie

Im letzten Kapitel wurden die *Service Nodes* festgelegt. Diese sollen die Verteilung der Berechtigung, also die Verteilung der kryptografischen Schlüssel und die Erzeugung der Schlüssel, übernehmen. Wie in Kapitel 2.4.1 bereits bei mobilen Geräten im Einsatz, wird hier überprüft, wie das *Shamir Secret Sharing* Verfahren für Blockchain als Schlüsselmanagement angewendet werden kann. Das entspricht einem Verteilen des Basisschlüssels auf viele Beteiligte. Damit wird sichergestellt, dass nur ein Konsortium an *Service Nodes* den Basisschlüssel zur Erzeugung neuer Schlüssel zusammensetzen kann. Des Weiteren wird das besprochene Verfahren CASSP 3.5.2 von Piller et al. – auf Basis vom *Rabin-Kryptosystem* zur Schlüsselerzeugung – auf ihren Einsatz in einer Blockchain überprüft.

3.5.1. Sichere Aufbewahrung des Basisschlüssels

Die Basis für die Kryptografie stellt ein Schlüssel (z_i, n) dar. Dieser ist notwendig, um einen Geheimtext mittels geheimen Schlüssel zu einem Klartext umzuwandeln. Dieser Schlüssel muss vor unberechtigten Zugriffen geschützt werden. Er stellt sich aus einer Zufallszahl z_i und aus n, dem Produkt aus p und q, welche Primzahlen sind, zusammen. Dadurch können mit p und q neue Schlüssel (z_{i+1}, n) aus (z_i, n) erstellt werden. Mit z_1, p und q lassen sich neue Schlüsselgenerationen generieren.
Somit ist es von großer Bedeutung, den Basisschlüssel d, bestehend aus (p, q, z_1), geheim aufzubewahren. Dabei gilt es als unsicher, den Schlüssel an nur einem einzigen Ort abzuspeichern:

- Alle Daten einer Blockchain können mit dem geheimen Schlüssel geschrieben werden und die Blockchain wird dadurch angreifbar.
- Des Weiteren kann bei Verlust des Basisschlüssels kein neuer Schlüssel ausgestellt werden.
- Bei Verlust ist das Lesen von Daten in der Blockchain nicht mehr möglich.

Da die zuvor gewählten *Service Nodes* vertrauenswürdig sind, wird mittels *Shamir Secret Sharing* der Basisschlüssel auf diese aufgeteilt. Somit gibt es keine zentrale Autorität, die die Schlüssel verwaltet. Um die Erstellung der Schlüssel sicher zu gestalten, kann das durch *smart contracts* in einem *Trusted Platform Module* erfolgen. Anschließend wird das Geheimnis des Basisschlüssels d, bestehend aus (p, q, z_1), in einem Hardware-Token abgelegt. Dadurch ist sichergestellt, dass niemand unberechtigt darauf zugreifen kann. Nähere Informationen sind in Kapitel 3.7 ersichtlich.

Dabei wird der Basisschlüssel d in n Teile, n entspricht der Anzahl der eingesetzten *Service Nodes* $d_1, d_2, d_3, \ldots, d_n$, aufgeteilt. Zusätzlich wird ein Schwellenwert k definiert. Dieser stellt sicher, dass die Anzahl von (k) beteiligten *Service Nodes* benötigt wird, um den Schlüssel zusammenzustellen. Dies ist ein zusätzlicher Sicherheitsfaktor, damit $(k-1)$ Teilnehmer den Schlüssel nicht herstellen können.
Gehen dabei Schlüsselteile $n < k$ verloren, kann dieser trotzdem folgendermaßen wiederhergestellt werden:

Zuerst müssen $k-1$ zufällig gewählte Zahlen $a_1, a_2, a_3, \ldots, a_{k-1}$ aus einem Galois Feld $\mathbb{F}$ der Größe p ausgewählt werden, wobei folgendes zutreffen muss:

$$p \in \mathbb{P} : p > S, p > n, n \geq k, k > 0 \tag{3.1}$$

Das a_0 entspricht dabei dem geheimen Schlüssel S. Alle anderen zufällig gewählten Zahlen entsprechen $a_i < p$, p ist dabei eine Primzahl. Anschließend wird mittels Funktion 3.2 das Polynom erstellt:

$$f(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + a_3 \cdot x^3 + \cdots + a_{k-1} \cdot x^{k-1} \tag{3.2}$$

Im nächsten Schritt 3.3 werden die n Teile erstellt:

$$d_{x-1} = (x, f(x) \mod p, x = 1, 2, \ldots, n) \tag{3.3}$$

Jeder Teil ist ein Punkt d auf dem zuvor erstelltem Polynom. Dabei verbirgt sich hinter a_0 immer noch der geheime Schlüssel. Die daraus erstellten Punkte $d_1, d_2, d_3, \ldots, d_n$, werden auf die Anzahl n *Service Nodes* der Blockchain aufgeteilt.

Um den geheimen Schlüssel zu extrahieren, muss mindestens die Anzahl k an *Service Nodes* die folgende Lagrangesche Interpolationsformel 3.4 anwenden:

$$
\begin{aligned}
l_0 &= \left(\frac{x-x_1}{x_0-x_1}\cdot\frac{x-x_2}{x_0-x_2}\cdot\ldots\cdot\frac{x-x_{k_1}}{x_0-x_{k1_1}}\right) \pmod p \\
l_1 &= \left(\frac{x-x_0}{x_1-x_0}\cdot\frac{x-x_2}{x_1-x_2}\cdot\ldots\cdot\frac{x-x_{k_1}}{x_1-x_{k1_1}}\right) \pmod p \\
&\vdots \\
l_{k-1} &= \left(\frac{x-x_0}{x_{k-1}-x_0}\cdot\ldots\cdot\frac{x-x_{k-2}}{x_{k-1}-x_{k1-2}}\right) \pmod p
\end{aligned}
\tag{3.4}
$$

Daraus ergibt sich folgende Funktion 3.5:

$$
f(x) = \sum_{j=0}^{k-1} y_i \cdot l_j(x) \tag{3.5}
$$

Unter der Voraussetzung, dass alle Funktionen erfüllt sind, können *Service Nodes* in einer Blockchain geheime Schlüssel anhand von *Shamir Secret Sharing* aufteilen und somit eine sichere Schlüsselaufteilung gewährleisten. Des Weiteren ist sichergestellt, dass durch die Anwendung dieses Verfahren nur ausgewählte *Service Nodes*, mit denen der Schlüssel geteilt wird, diesen gemeinsam wiederherstellen können. Scheidet ein *Service Node* aus, kann der Schlüssel trotzdem wiederhergestellt werden.

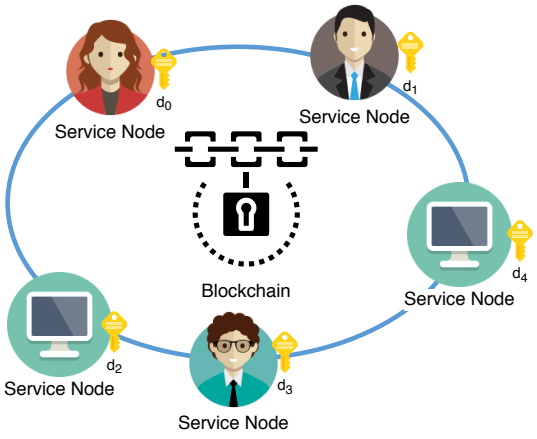


Abbildung 3.2.: Eigene Darstellung, Schlüsselaufteilung *Service Nodes*

In der Abbildung 3.2 ist ersichtlich, wie die einzelnen Teile des Basisschlüssels $d_1, d_2, d_3, \ldots, d_n$, welche Punkte auf dem Polynom darstellen, dezentral über alle *Service Nodes* verteilt sind. Gemeinsam mit den Teilen des Basisschlüssels kann dieser wiederhergestellt und somit eine neue Schlüsselversion (z_{i+1}, n) generiert werden. Im Anhang unter A.1 ist ein Bespiel zur Schlüsselaufteilung vollständig zur

Erläuterung berechnet.

Aus diesem Beispiel resultierend, ist der Einsatz des *Shamir Secret Sharing* Verfahren zum Aufteilen von kryptografischen Schlüsseln, zur Verschlüsselung von Daten in einer Blockchain möglich. Dabei gilt:

- k *Service Nodes* können den Basisschlüssel d wiederherstellen, um neue kryptografische Schlüssel (z_{i+1}, n) zu erstellen.
- $k-1$ *Service Nodes* und damit d_i Teile sind nicht ausreichend, um den Basisschlüssel zu rekonstruieren. Somit können sie keine neuen kryptografischen Schlüssel erzeugen.

Der Basisschlüssel d, bestehend aus (p, q, z_1), dient gleichzeitig den *Service Nodes* zur Signatur dieses Schlüssels, als privater Schlüssel. Dieser wird dazu verwendet, den Basisschlüssel selbst zu signieren. Dabei ist der Schlüssel (z_1, n) zugleich als öffentlicher Schlüssel tätig. Damit kann der Schlüssel der *Service Nodes* auf Echtheit verifiziert werden.

3.5.2. Schlüsselerzeugung

Nachdem das letzte Kapitel die Sicherung des geheimen Basisschlüssel behandelt hat, wird in diesem Abschnitt das Erzeugen von Schlüssel auf den Einsatz in einer Blockchain mittels *Rabin-Kryptosystem* überprüft. Dabei werden wieder die Zufallszahlen z_i und n, welche ein Produkt aus den Primzahlen p und q sind, verwendet. Dieses Verfahren baut auf dem Prinzip des Quadrierens auf. Mit Hilfe von n wird ein älterer Schlüssel z_{i-1} aus dem aktuellen Schlüssel z_i berechnet. Dabei sind die geheimen Teile des Schlüssels p und q nicht notwendig. Des Weiteren werden durch das Wurzelziehen, wobei hier p und q benötigt werden, neue Schlüssel z_{i+1} erstellt.
Das *Rabin-Kryptosystem* ist nur unter der Voraussetzung anwendbar, wenn gilt:

$$p \equiv q \equiv 3 \pmod 4 \tag{3.6}$$

Dabei sind p und q die geheimen Teile, die nur zur Erzeugung neuer kryptografischer Schlüssel (z_{i+1}, n), für $i > 1$, benötigt werden. Sie sind, wie unter Kapitel 3.5.1 erklärt, auf die unterschiedlichen *Service Nodes* als Teile $d_1, d_2, d_3, \ldots, d_n$ aufgeteilt.
Zur Erstellung neuer Schlüssel (z_{i+1}, n) muss eine modulare Quadratwurzel von $z_i \mod n$ mittels For-

mel 3.7 und mit Hilfe vom *Chinesischen Restsatz* 3.8 erfolgen:

$$\begin{aligned} a_1 &= \sqrt{z} = z^{\frac{p+1}{4}} \mod p \\ a_2 &= \sqrt{z} = p - a_1 \mod p \\ b_1 &= \sqrt{z} = z^{\frac{q+1}{4}} \mod q \\ b_2 &= \sqrt{z} = q - b_1 \mod q \end{aligned} \tag{3.7}$$

$$z_{i+1} = \sqrt{z_i} \equiv b \cdot s \cdot p + a \cdot t \cdot q \pmod{n} \tag{3.8}$$

Die fehlenden Werte s und t werden unter Zuhilfenahme des *erweiterten euklidischen Algorithmus* 3.9 aus den Werten p und q berechnet.

$$\mathrm{ggT}(p, q) = s \cdot p + t \cdot q \tag{3.9}$$

Das Ergebnis aus dem *Chinesischem Restsatz* ergibt, durch die vier unterschiedlichen a und b (a_1, b_1), (a_1, b_2), (a_2, b_1) und (a_2, b_2), vier weitere Ergebnisse aus der modularen Quadratwurzel. Aus diesen vier Resultaten muss der Schlüssel z_{i+1} ausgewählt werden, welcher einen *quadratischen Rest* $\pmod{n}$ ergibt. Dabei wird unter Anwendung des *Eulerschen Kriteriums* 3.10 jedes Ergebnis gewählt, das diese Bedingung erfüllt:

$$\begin{aligned} z^{\frac{p-1}{2}} \mod p = 1 \\ z^{\frac{q-1}{2}} \mod q = 1 \end{aligned} \tag{3.10}$$

Werden die oben angeführten Kriterien alle erfüllt, ergibt sich daraus ein neuer Schlüssel (z_{i+1}, n).

Wie in der Abbildung 3.3 dargestellt, wird dieser neue Schlüssel z_{i+1} auf alle *Service Nodes* aufgeteilt und dazu verwendet, Daten in der Blockchain zu verschlüsseln. Dabei besitzen alle *Service Nodes* denselben Schlüssel. Zusätzlich benötigen alle Teilnehmer ebenfalls diesen Schlüssel, damit sie die Möglichkeit haben, Daten zu lesen. Sie haben dadurch nicht automatisch ein Leserecht auf alle Blöcke. Diese sind zusätzlich mit einem eigenen Schlüssel zum Lesen der Daten versehen.

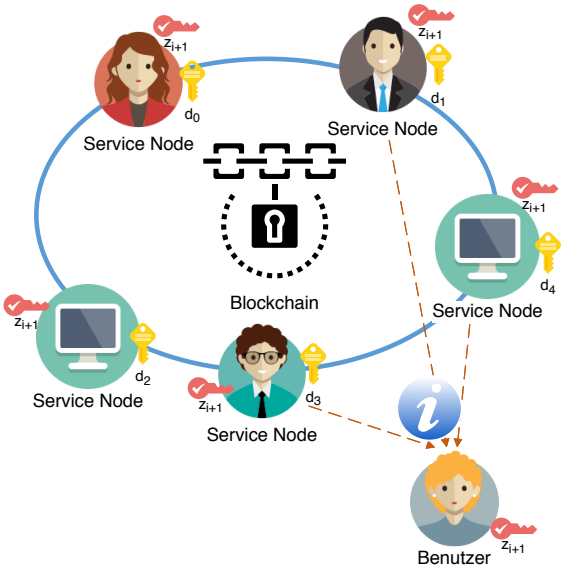


Abbildung 3.3.: Eigene Darstellung, Schlüsselverteilung *Service Nodes*

Tritt ein neuer *Service Node* ein, bekommt er dadurch nur den aktuellen Schlüssel (z_i, n). Somit fehlen ihm alle alten Schlüssel zum Entschlüsseln der Daten. Durch den Einsatz dieses Verfahren kann jedoch auf ältere Schlüssel (z_{i-1}, n) retour gerechnet werden. Dabei sind die geheimen Zahlen p und q nicht notwendig. Durch Quadrieren 3.11 wird zu einem vorherigen Schlüssel retour gerechnet:

$$z_{i-1} = {z_i}^2 \pmod{n} \text{ wenn, } i > 2 \tag{3.11}$$

Beim Ausscheiden eines *Service Node* ist dieser im Besitz des aktuellen Schlüssels (z_{i+1}, n). Da er bereits alle Informationen gelesen hat, ist es nicht notwendig, ihm den Schlüssel zu entziehen. Es wird allerdings eine neue Schlüsselversion (z_{i+2}, n) erstellt und nicht mit dem eliminierten *Service Node* geteilt. Alle anderen *Service Nodes* erhalten diesen neuen Schlüssel. Somit kann dieser neu verschlüsselte Blöcke in der Blockchain lesen.

Die Übertragung der Schlüssel (z_{i+1}, n) muss über einen sicheren Kanal erfolgen. Dies wird über das symmetrische Verfahren AES sichergestellt. Dabei muss für den erstmaligen Verbindungsaufbau, der einen unsicheren Kanal darstellt, ein Geheimnis vereinbart werden. Dies kann mittels Schlüsselaustausch über *Diffie-Hellman* (DH) stattfinden. Dabei vereinbaren zwei Teilnehmer über den unsicheren Kanal eine große Primzahl p und eine natürliche Zahl g. Werden diese abgefangen, kann der sichere Kanal trotzdem nicht abgehört werden. Danach wählt jeder der beiden Teilnehme für sich eine geheime Zahl a

und b aus $a, b \in \mathbb{N} : \{1, 2, \ldots, p-1\}$ und berechnet 3.12 seinen öffentlichen Schlüssel:

$$\begin{aligned} \alpha &= g^a \mod p \\ \\ \beta &= g^b \mod p \end{aligned} \tag{3.12}$$

Anschließend werden α und β ausgetauscht und jeweils der gemeinsame geheime Schlüssel k bzw. k' berechnet. 3.13:

$$\begin{aligned} k &= \beta^a \mod p \\ \\ k' &= \alpha^b \mod p \end{aligned} \tag{3.13}$$

Dadurch lässt sich das symmetrische Verfahren, trotz des gleichen Schlüssels, zum Verschlüsseln und Entschlüsseln mit dem asymmetrischen Verfahren zum Schlüsselaustausch optimal in ein Hybridverfahren kombinieren. Dieses stellt ein sehr effizientes Verfahren beim Einsatz in einer Blockchain dar. Damit lassen sich mit Hilfe von symmetrischer Übertragung Daten sicher weiterentwickeln. Eine andere Möglichkeit ergibt sich aus der Verwendung von *Diffie-Hellman* mit elliptischen Kurven, bekannt als *Elliptic Curve Diffie-Hellman* (ECDH), für den Austausch auf dem zuvor unsicheren Kanal. Dieses Verfahren wählt einen Punkt auf einer elliptischen Kurve aus. Damit ist es robust gegen Angriffe.
Zusammenfassend lässt sich die CASSP-Methode mit dem *Rabin-Kryptosystem*-Verfahren durch Aufteilen der Schlüssel auf unterschiedliche *Service Nodes* und Verteilung des symmetrischen Schlüssels mittels *Diffie-Hellman* sicher in einer Blockchain anwenden.

3.5.3. Verschlüsselung von Blöcken

Des Weiteren muss sichergestellt werden, dass jeder Blockchain Block durch Leseschutz zusätzlich geschützt wird. Das wird durch das Verschlüsseln der Blöcke in der Blockchain gewährleistet. Dabei muss jeder *Service Node*, sowohl aus einer flachen Hierarchie aber auch als *Service Node* aus einem Konsortium, Daten verschlüsseln können. Das *Rabin-Kryptosystem*-Verfahren kann als asymmetrische Methode angewendet werden. Zusätzlich besitzt danach jeder *Service Node* seinen eigenen privaten Schlüssel $(z_a^d, z_b^d, \ldots, z_n^d)$, bestehend aus dem eigenen geheimen p und q und einem öffentlichen Schlüssel $(z_a^e, z_b^e, z_c^e, \ldots, z_n^e)$ aus dem Produkt von p und q. Damit alle Teilnehmer Daten in der Blockchain lesen können, besitzen diese ebenfalls einen öffentlichen und einen privaten Schlüssel.

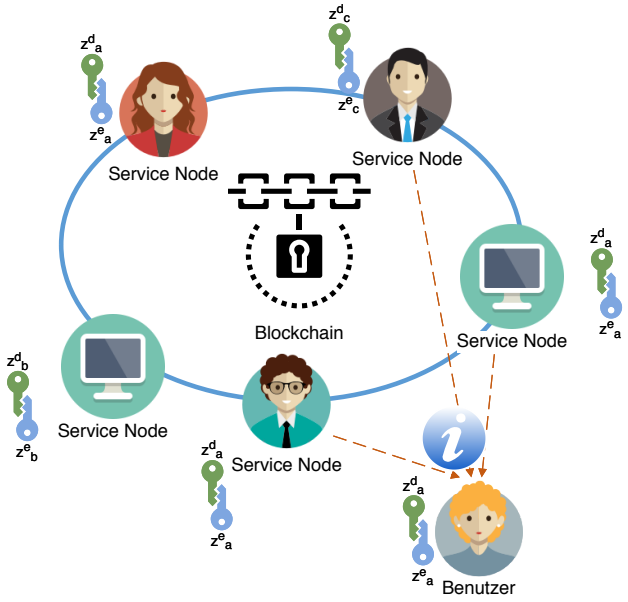


Abbildung 3.4.: Eigene Darstellung, Verteilung privater und öffentlicher Schlüssel

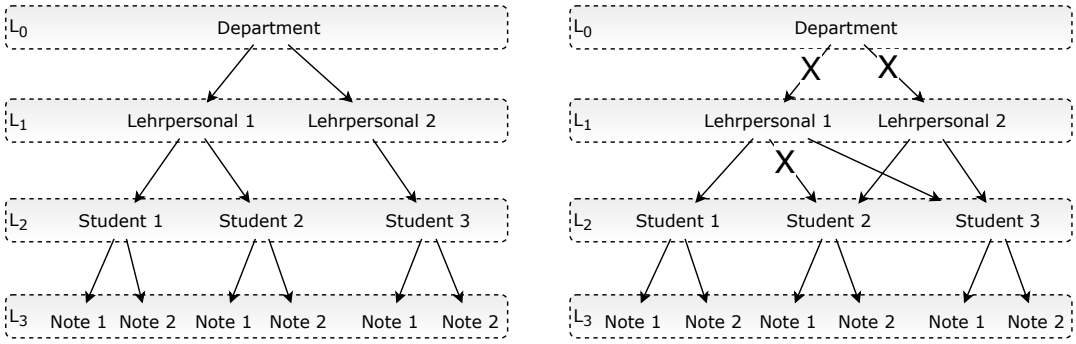
Wie in der Abbildung 3.6 erkennbar, sind somit öffentliche und private Schlüssel auf die gesamten Teilnehmer der Blockchain verteilt. Diese werden durch die kryptografische Zugriffskontrollen dazu verwendet gesicherte Blöcke zu lesen. Zusätzlich werden von den *Service Nodes* die privaten und öffentlichen Schlüssel zum Signieren von Blöcken verwendet.

Für die Verschlüsselung der einzelnen Blöcke können auch kryptografische Verfahren, wie z.B. *Pretty Good Privacy* (PGP) [39] oder RSA, zum Einsatz kommen. Dabei müssen alle Blöcke mit einem eigenen Schlüssel verschlüsselt werden. Nur so kann gewährleistet werden, dass mittels Löschen der Schlüssel die DSGVO eingehalten wird.

Ist eine hierarchische kryptografische Zugriffskontrolle notwendig, kann das mit dem bereits bestehenden Verfahren von Mackinnon et al. [40] oder Akl und Taylor [41] umgesetzt werden. Dieses Verfahren setzt auf die RSA-Verschlüsselung auf. Dabei können Zugriffskontrollen mittels einer Organisationsstruktur oder aber auch individueller Zugriffe, wie in Abbildung 3.5 dargestellt, abgebildet werden. Bei diesem Verfahren werden Schlüssel entsprechen der Hierarchie $L_0 > L_1 > L_2 > L_3$ aufgebaut. Das Schema kontrolliert Daten anhand einer organisatorischen Hierarchie, welche teilweise geordnet sein kann. Es erlaubt Benutzern aus der Organisation, auf unterschiedliche Level aus dem eigenen kryptografischen Schlüssel, Schlüssel aus der darunterliegenden Hierarchie abzuleiten. Das Level L_0 ist, wie in (a) veranschaulicht, auf alle Daten mit dem eigenen Entschlüsselungsschlüssel berechtigt. Allerdings darf ein Level L_1 nur auf die darunterliegende Struktur zugreifen, hat aber kein Recht für das darüberliegende Level L_0. Hingegen wird in (b) die Steuerung der Zugriffe individuell umgesetzt.

Die erzeugten Schlüssel sind nicht nur voneinander abhängig, sondern werden immer mit der indivi-

duellen darüberliegenden Hierarchie erstellt. Eine hierarchische kryptografische Zugriffskontrolle kann ebenfalls den Level L_0 aus dem Basisschlüssel d ableiten.



(a) Zugriffe in einer Organisationsstruktur

(b) Individuelle Zugriffssteuerung

Abbildung 3.5.: Eigene Darstellung, Zugriffssteuerung in einer Hierarchie, in Anlehnung an [40, S.798ff]

Abschließend steht fest, dass die Schlüsselaufbewahrung mittels *Shamir Secret Sharing* Methode und die Schlüsselerzeugung durch die CASSP-Methode, unter Anwendung des *Rabin-Kryptosystem*, in einer Blockchain anwendbar ist. Dadurch ist gewährleistet, dass einzelne *Service Nodes* nicht fälschlicherweise Daten in eine Blockchain hinzufügen können. Der Nachteil einer Neuverschlüsselung von großen Datenmengen, welcher sich bei CASSP im Zusammenhang mit mobilen Geräten ergibt, kommt in der Blockchain nicht zu tragen. Bei der Blockchain ist die Neuverschlüsselung von Daten nicht notwendig, da diese unveränderbar und deshalb vor Manipulationen geschützt sind. Durch den Einsatz von asymmetrischer Kryptografie lassen sich einzelnen Blöcke vor unberechtigten Teilnehmern schützen. Des Weiteren kann eine hierarchische Zugriffskontrolle mittels Mackinnon et al. für die Blockchain angewendet werden. Die Vielzahl der Schlüssel macht die Implementierung einer Schlüsselverwaltung notwendig. Diese wird unter Kapitel 3.7 näher betrachtet. Im nächsten Kapitel werden die angeführten Methoden in neue Konsensverfahren eingebunden.

3.6. Neuartige Konsensverfahren mittels kryptografischem Zugriffskontrollsystem

Wie bereits erwähnt, gibt es bereits einige Algorithmen für Konsensverfahren. Diese sind allerdings entweder nicht energiesparend, sehr zeitaufwändig oder nicht für ein kryptografisches Zugriffskontrollsystem geeignet. Unter Schlüsselmanagement auf Basis von Kryptografie wurde die Voraussetzung dafür geschaffen, dass *Shamir Secret Sharing* und *Rabin-Kryptosystem* für die Blockchain einsetzbar sind.

Auf dieser Grundlage werden die neuen Konsensverfahren *proof of cryptography* und *proof of signature* entwickelt. Sie können durch den Einsatz von *Service Nodes* angewendet werden. Diese Verfahren sind auf Grund des Einsatzes von kryptografischen Methoden sehr schnell und effizient. Im Gegensatz zu herkömmlichen Zugriffskontrollsystemen benötigen die neu entwickelten Methoden keinen zentralen Referenzmonitor. Sie gewährleisten rasch einen Konsens über alle beteiligten *Service Nodes*, um Daten sicher in der Blockchain zu speichern. Da das Konsensverfahren selbst keine Auswahl auf Richtigkeit treffen kann, ist sicherzustellen, dass anhand von Kriterien oder Regeln die Entscheidungen getroffen werden. Die Implementierung neuer Modelle muss die aktuelle Philosophie der Blockchain weiterverfolgen.
Im Anschluss ist die Ausarbeitung von neuen Lösungskonzepten für Konsensverfahren dargestellt.

3.6.1. Konsensverfahren - *proof of cryptography*

In einem dezentralen Netzwerk ist es wesentlich, einen Konsens oder eine Vereinbarung zwischen Beteiligten über die Gültigkeit von Transaktionen zu finden. Dabei ist es nicht notwendig, dass diese sich gegenseitig kennen. Dieses Konzept trägt dazu bei, sehr ökonomisch und wirkungsvoll, den Konsens zwischen *Service Nodes* zum Validieren von Daten in einer Blockchain zu erlangen. Dabei wird die Technik von *Shamir Secret Sharing* zur Aufteilung des Basisschlüssels und CASSP zur Ausstellung neuer Schlüssel zum Verschlüsseln von Daten benutzt.
Nachfolgend ist das Konzept für das Konsensverfahren *proof of cryptography* definiert:

- ***Definition 1*:** Dieses Konsensverfahren verwendet zur Schlüsselerstellung nur zwei geheime Primzahlen p, q und eine Zufallszahl z_1. Damit können neue Schlüsselgenerationen (z_{1+1}, n) erstellt werden. Alle Daten in der Blockchain werden mit diesem Schlüssel z_1 mit dem symmetrischen Verschlüsselungsverfahren AES verschlüsselt und entschlüsselt. Die Übertragung der Schlüssel erfolgt dabei durch ein Diffie-Hellman (DH) Schlüsselaustauschverfahren.

- ***Definition 2*:** Der Basisschlüssel d wird auf alle gewählten *Service Nodes* aufgeteilt. Damit ist ein Betrug zum Erstellen neuer Schlüssel durch einzelne *Service Nodes* ausgeschlossen. Wird ein *Service Node* ausgegliedert, kann der Basisschlüssel auf Grund eines definierten Schwellwertes zur Wiederherstellung neuer Schlüssel trotzdem zusammengestellt werden. Somit ist eine Unabhängigkeit von einzelnen *Service Nodes* gegeben. Um Daten neu zu verschlüsseln, können Schlüssel daher jederzeit neu ausgestellt werden.

- ***Definition 3*:** Auf Grund des Besitzes des aktuellen Schlüssels bei *Service Nodes*, welche als vertrauensvolle *Service Nodes* ausgewählt wurden, kann ein Konsens der Blöcke jederzeit durch die

Service Nodes und deren Prüfung des kryptografischen Schlüssels auf Validität geprüft und gewährleistet werden. Durch Rückrechnen alter Schlüssel z_{1-1} beim CASSP-Verfahren ist jederzeit die Validierung älter Blöcke möglich.

- ***Definition 4*:** Veränderungen bei den *Service Nodes* durch Ausscheiden oder Hinzukommen von Neuen wirken sich direkt auf das Konsensverfahren aus. Dadurch müssen gemeinsam neue Schlüssel z_{1+1} erstellt werden. Sie sind für alle neuen Verschlüsselungen zu verwenden. Dadurch wird verhindert, dass ehemalige *Service Nodes* auf neu verschlüsselte Daten zugreifen können. Alte, bereits eingesehene Daten können allerdings weiterhin gelesen werden, da diese Information bereits zuvor geteilt wurde.

- ***Definition 5*:** Werden Blöcke von anderen Teilnehmern außerhalb des *Service Nodes* Kreises eingetragen, finden diese niemals einen Konsens und enden als ungültige Ketten in der Blockchain. Dadurch wird sichergestellt, dass nur vertrauenswürdige *Service Nodes* Blöcke zur Blockchain hinzufügen können. Somit kann eine böswillige Übernahme der Blockchain verhindert werden.

- ***Definition 6*:** Darüber hinaus kann zur symmetrischen AES-Verschlüsselung jeder einzelne Block zusätzlich durch eine asymmetrische Verschlüsselung geschützt werden. Damit lassen sich ergänzend Leserechte für einzelne Blöcke verwirklichen.

- ***Definition 7*:** Der Einsatz des Konsensverfahren funktioniert nur unter der Voraussetzung, dass die unter Kapitel 3.4.1 definierten *Service Nodes* diese Aufgabe übernehmen. Sie sind dafür verantwortlich, die Schlüssel $(z_1, z_2, \ldots, z_n)$ für das Blockchain-Netzwerk zu verteilen.

Die oben angeführten Definitionen müssen erfüllt sein, damit das Konsensmodell *proof of cryptography* angewendet werden kann. Durch die Verkettung und Referenzierung des Hash-Werts vom vorhergehenden in den nachfolgenden Block-Header ist sichergestellt, dass ein Block, sobald er als gültig angesehen wird, nicht mehr verändert werden kann. Des Weiteren gilt dieses Modell als sehr effizient und benötigt keine große Rechenleistung. Dabei geht es nicht darum, welche Kette in der Blockchain am längsten ist, sondern nur darum, dass durch Konsortien gewählte *Service Nodes* Blöcke in der Blockchain bestätigen. Das hier veröffentlichte Konsensverfahren baut auf die symmetrische Kryptografie auf und besticht durch das sehr rasche AES-Verschlüsselungsverfahren.
Im nächsten Kapitel wird ein weiteres Konzept für ein Konsensverfahren dargestellt, welches auf die Grundlage der digitalen Signatur aufsetzt.

3.6.2. Konsensverfahren - *proof of signature*

Das hier vorgestellte Verfahren *proof of signature*, ist ein kryptografisches Zugriffskontrollsystem, welches auf asymmetrischer Verschlüsselung basiert. Dazu sind Schlüssel zum Verschlüsseln und Entschlüsseln notwendig. Die Idee liegt darin, dass alle Beteiligten ihre Zustimmung für einen Konsens geben müssen. Diese Aufgabe übernehmen die hier eingesetzten *Service Nodes*. Damit kann die Echtheit der Transaktionen sichergestellt werden.
Anhand eines Beispiels wird diese Funktion verdeutlicht: wird eine Blockchain als Bahnticket betrachtet, so ist dieses Ticket von einem Startpunkt zu einem Fahrtziel ausgestellt. Die Strecke beinhaltet viele Verkehrsverbünde. Somit wird das Ticket und damit eingeschlossen der Block sowie die Transaktion nur dann gültig, wenn alle Verkehrsverbünde zustimmen, dass die Strecke und das Ticket ident sind.

Anschließend ist das Konzept für die Methode *proof of signature* beschrieben:

- ***Definition 1*:** Dieses Verfahren baut auf Basis der digitalen Signatur auf. Ein Block oder eine Transaktion in der Blockchain wird nur dann gültig, wenn jeder *Service Node* seine Zustimmung durch Signieren des Blockes erteilt hat. Die Signatur stellt dabei die Verbindung zu der Identität der einzelnen *Service Nodes* dar.

- ***Definition 2*:** Ein *Service Node* (SN_1) benutzt zur Erzeugung seines privaten Schlüssels zwei geheime Primzahlen p, q und eine Zufallszahl z_1. Zusätzlich erhält dieser einen öffentlichen Schlüssel. Jeder hinzugefügte Block in der Blockchain muss auf seine Validität geprüft und anschließend mit dem privaten Schlüssel des SN_1 signiert werden. Somit ist der Block gültig. Ein anderer *Service Node* (SN_2) kann mit dem öffentlichen Schlüssel von SN_1 die Signatur überprüfen und diesen ebenfalls signieren.

- ***Definition 3*:** Ein definierter Schwellwert unterstützt die Entstehung eines Konsenses. Dieser kann, je nach Art der Anwendung der Blockchain, individuell gewählt werden. Mindestens die als Schwellwert definierte Anzahl an *Service Nodes* muss den Block signieren, damit dieser gültig ist. Der Konsens muss in einem fairen Auftreten erfüllt werden und darf nicht von Eigeninteresse sein.

- ***Definition 4*:** Blöcke können zusätzlich zu der digitalen Signatur mit einem asymmetrischen Verfahren verschlüsselt werden. Damit können Leserechte in den Blöcken dazu beitragen, die Sicherheit zu erhöhen und nur Berechtigten Zugriff zu gewähren.

- ***Definition 5*:** Fügen andere Teilnehmer außerhalb des *Service Nodes*-Kreises Blöcke in die Blockchain hinzu, erlangen diese keine Gültigkeit. Sie werden keinen Konsens zwischen den qualifi-

zierten *Service Nodes* erhalten. Somit bleiben sie zwar gespeichert, enden aber als unvollständige Blöcke.

- ***Definition 6*:** Nur unter der Annahme, dass die unter Kapitel 3.4.1 definierten *Service Nodes* diese Aufgabe übernehmen, wird diese Methode erfolgreich zum Einsatz kommen. Sie gewährleistet die Echtheit der Blöcke in der Blockchain.

Werden diese Definitionen erfüllt, kann das *proof of signature* Verfahren erfolgreich in einer Blockchain angewendet werden. Dieses Modell ist gegenüber anderen Modellen, wie z.B. dem *proof of work*, sehr effizient und vergeudet keine unnötige Rechenleistung und Zeit, damit Blöcke in eine Blockchain hinzugefügt werden können. Obwohl der Prozess sehr einfach ist, gilt er als sicher und benötigt keine zentrale Verwaltung oder Steuerung zum Finden eines Konsenses. Die vertrauensvoll ausgewählten *Service Nodes* können sehr ökonomisch Transaktionen durch die Überprüfung der digitalen Signatur in eine Blockchain verankern.

3.7. Implementierung von kryptografischen Zugriffskontrollsystemen in Blockchains

Durch den Einsatz neuer *Service Nodes* und neuer Konsensverfahren, wie in Kapitel 3.4 und 3.6 beandelt, ist es notwendig, diese Methoden durch eine sichere Implementierung in der Blockchain zu unterstützen. Wie in der Darstellung 3.6 erkennbar, haben *Service Nodes* durch ihre neue Funktion eine große Anzahl an Schlüssel zu verwalten. Werden diese geheimen Schlüssel allerdings in einer unsicheren Umgebung, wie z.B. einem Computer gespeichert, entsteht ein großes Sicherheitsrisiko für die Blockchain. Die Daten in der Blockchain sind dann nicht sicher aufbewahrt und können durch unsachgemäße Handhabung kompromittiert oder falsche Blöcke hinzugefügt und bestätigt werden. Deshalb muss die Schlüsselverwaltung in einer sicheren Umgebung stattfinden.
Wie in Kapitel 2.4.2 behandelt, stellt ein *Trusted Platform Module* (TPM) diese Sicherheit zur Verfügung. Dieses fügt einem Computer zur Unterstützung einen kleinen Mikrocontroller hinzu, welcher die Aufgabe der *Trusted Computing* Technologie übernimmt. Somit wird die Verarbeitung von sensiblen Daten in einem abgetrennten Bereich mittels Kryptografie durchgeführt. Das System stellt die Integrität des Computers durch Prüfen auf Veränderung sicher.
Zusätzlich muss die Sicherheit durch den Einsatz von Security-Token und Chipkarten, wie z.B. USB-Sticks oder Smartcards, erhöht werden. Sie stellen trotz ihrer sehr geringen Kosten eine hochsichere Lösung für die Speicherung von persönlichen Daten, wie geheime kryptografische Schlüssel, dar. Sie

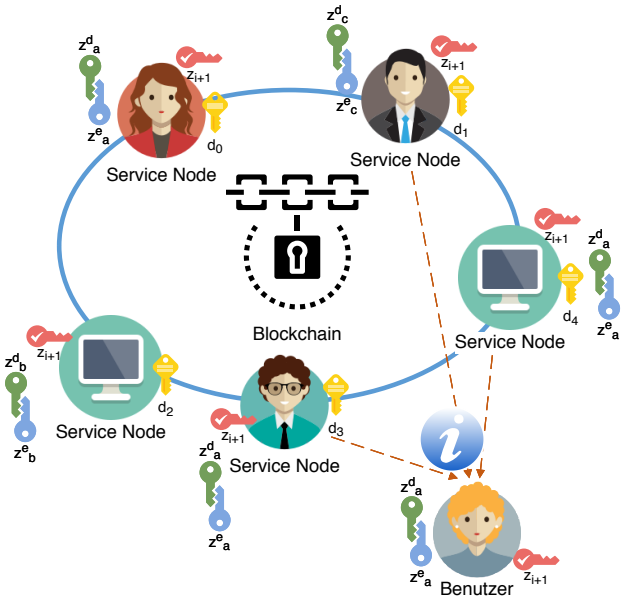


Abbildung 3.6.: Eigene Darstellung, Übersicht Schlüsselverteilung

können für die Identifikation und Authentifikation von Benutzern herangezogen werden. Sie stellen bereits heute bei elektronischen Reisepässen, Bankomatkarten und USB-Sticks sowie SIM-Karten in Mobiltelefonen und bei der österreichischen Handy-Signatur einen sehr hohen Sicherheitsstandard zur Verfügung. Die Hardware besitzt eine Schlüsselverteilung durch Zertifikatsketten, welche absolut geheim aufbewahrt werden.

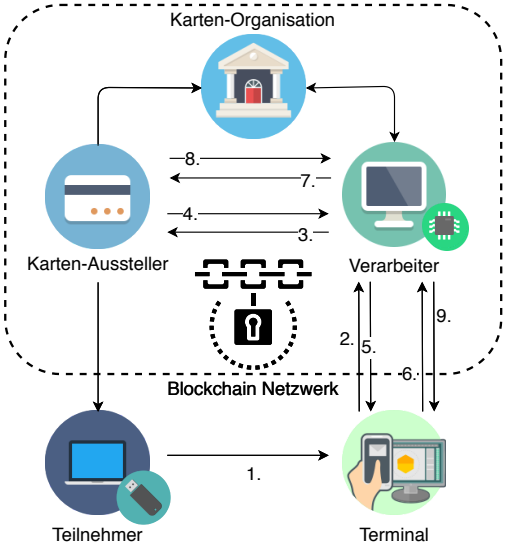


Abbildung 3.7.: Eigene Darstellung, Prozess zur Authentifikation

Wie unter Kapitel 2.4.2 besprochen, ist das EMV-System hochsicher. Deshalb soll diese Eigenschaft ebenfalls in das Blockchain-Netzwerk implementiert werden. Dabei kann ein Hardware-Token, z.B. als

Smartcard oder ein USB-Stick, verwendet werden. In der Abbildung 3.7 ist der Ablauf zur Authentifikation für Findung eines Konsenses durch *Service Nodes* mittels *proof of cryptography*, *proof of signature* oder aber für Teilnehmer zum Zugreifen auf Blöcke dargestellt. Dabei besitzt der Teilnehmer einen Hardware-Token mit einem vom Karten-Austeller signierten Zertifikat. Dieses wurde wiederum durch eine *Certificate Autority* von der Karten-Organisation ausgestellt. Dadurch kann der private Schlüssel auf dem Token sicher abgelegt und mit dem öffentlichen Zertifikat im Terminal auf Validität gegeneinander geprüft werden. Diese Methode entspricht dem DDA-Verfahren aus Kapitel 2.4.2.
Anschließend wird dieser Prozess näher erläutert:

Schritt 1: Zuerst wird die Smartcard in das Terminal bzw. der USB-Stick in den Computer gesteckt. Dadurch wird eine Anforderung für den Zugriff erzeugt.

Schritte 2-5: Nachdem die Hardware für den Zugriff gesteckt wurde, wird sofort eine Anfrage für eine Autorisierung vom Terminal an den Verarbeiter weitergeleitet. Dieser sendet eine Anfrage über die Karten-Organisation an den Karten-Aussteller zur Überprüfung der Ermächtigung weiter. Dabei ist der Karten-Aussteller jene Organisation, die Smartcards, Token oder USB-Sticks mit der entsprechenden Berechtigung zur Verfügung stellt. Die Karten-Aussteller können in diesem Fall eine Organisation darstellen, die für die Blockchains Hardware-Token zum Zugreifen auf die Blockchain erteilen. Damit bekommen *Service Nodes* die Berechtigung, Blöcke zu erstellen. Ist die Autorisierung erfolgreich, wird dies dem Teilnehmer retour gemeldet.

Schritte 6-9: Sind die Schritte 2-5 erfolgreich, können anschließend autorisierte *Service Nodes* Blöcke in die Blockchain hinzufügen oder mit den zuvor erstellen Konsensverfahren durch den Verarbeiter bestätigen. Diese können z.B. *smart contracts* in der Blockchain darstellen. Somit können alle Teilnehmer der Blockchain durch ihren eigenen Hardware-Token authentifiziert und autorisiert werden. Damit werden auch Leseberechtigungen umgesetzt. Versuchen nicht-berechtigte Teilnehmer diese Aktion durchzuführen, scheitern sie bereits bei den Schritten 2-5.

Der gesamte Vorgang wird im *Trusted Computing*-Umfeld durchgeführt. Um die Sicherheit bei diesem Vorgang zu erhöhen, müssen die jeweiligen Teilnehmer ein zusätzliches Authentifizierungsmerkmal, z.B. einen PIN, als zweiten Faktor eingeben. So kann die maximale Sicherheit erreicht werden. Somit kann jeder Blockchain-Betreiber seine eigenen Blockchain-Karten oder Blockchain-USB-Sticks ausstellen und Teilnehmer berechtigen. Diese Hardware dient als fälschungssicherer elektronischer Schlüssel für die jeweilige Blockchain.

Wie in der Abbildung 3.8 ersichtlich, sind die zuvor definierten *Service Nodes* für ihre Kategorie verantwortlich. Dabei können unterschiedliche Anwendungsarten, wie z.B. Lieferungen, Verträge oder auch einfache Informationen, in einer Blockchain-Kette eingesetzt werden. Die jeweiligen *Service Nodes* sind daher für den Konsens ihrer Blöcke verantwortlich.

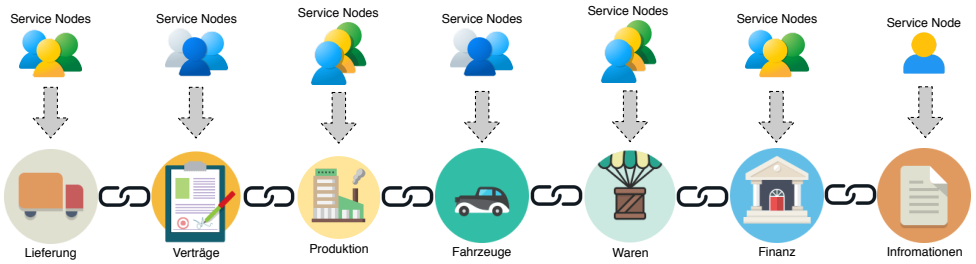


Abbildung 3.8.: Eigene Darstellung, Übersicht Blockchain

Um die entwickelten Konsensverfahren optimal einzusetzen, ist es nicht zwingenden notwendig, dass eine Blockchain linear wächst. Die Konsensverfahren *proof of cryptography* und *proof of signature* beruhen nicht auf dem Prinzip, dass die längste Kette die Gültige ist. Alleine der Konsens jedes Blockes reicht aus, dass eine Blockchain eine beliebige Form, wie z.B. eine Baumstruktur oder eine strahlenförmige Anordnung, darstellen kann. Dadurch können einzelne Ketten mit vielen Blöcken Einträge für einen Teilnehmer darstellen.

3.8. Datenschutz-Grundverordnung bei Blockchain for Data Security

Wie bereits erwähnt, sind in einer Blockchain keine Mechanismen zur Löschung von Daten und Blöcken vorhanden. Sie sind unveränderbar in einer Blockchain gespeichert. Zur Erfüllung der Datenschutz-Grundverordnung bei einer Blockchain ist es erforderlich, diese Anforderung auf einem neuartigen Weg umzusetzen. Eine Anonymisierung der Daten kann in einer Blockchain das Löschen von Daten nicht ersetzen. Durch den Einsatz von neuen *Service Nodes* kann die Anfrage zur Löschung von Daten, welche eindeutig einer bestimmten Person zuordenbar sind, umgesetzt werden.
Der Zweck der Datenverarbeitung muss dabei aufrecht bleiben. Ist ein rechtmäßiger Verarbeitungszweck vorhanden, wie z.B. ein Vertrag, eine Einwilligung oder auf Grund von berechtigtem Interesse, können die Daten aufbewahrt bleiben. Ist dieser jedoch nicht länger erfüllt, müssen die personenbezogen Daten unkenntlich gemacht werden, so dass kein Rückschluss auf Personen möglich ist. Ohne die Philosophie der Blockchain zu verändern, ist das einfache Löschen von Blöcken nicht möglich. Da alle Blöcke mit

einem *Hash*-Wert miteinander verkettet sind, verliert die Blockchain beim Löschen eines Blockes ihre Gültigkeit.

Um den Wünschen der Teilnehmer gerecht zu werden, übernehmen die neu definierten *Service Nodes* die Aufgabe der Verantwortlichen in der DSGVO. Zusätzlich kommen sie der Auskunftspflicht und dem Recht der Teilnehmer nach. Somit kann ein Teilnehmer über *smart contracts* Anfragen an die *Service Nodes* stellen, um Auskunft über Informationen zu erhalten. Wird dabei das „Recht auf Löschung“ in Anspruch genommen, führt dies ein *Service Node* durch. Dabei wird auf die Implementierung der Schlüssel für jeden einzelnen Block zurückgegriffen. Dadurch ist es möglich, durch Löschen des Schlüssels des jeweiligen Blockes ungewünschte Zugriffe zu vermeiden. Damit die Verkettung rückverfolgbar bleibt, werden die Blöcke nicht vollständig verschlüsselt. Bereits anonymisierte Daten im Block-*Header* bleiben im Klartext gespeichert. Somit lässt sich der *Hash* der benachbarten Blöcke und der *Merkle-Tree* weiterhin auf Korrektheit überprüfen.

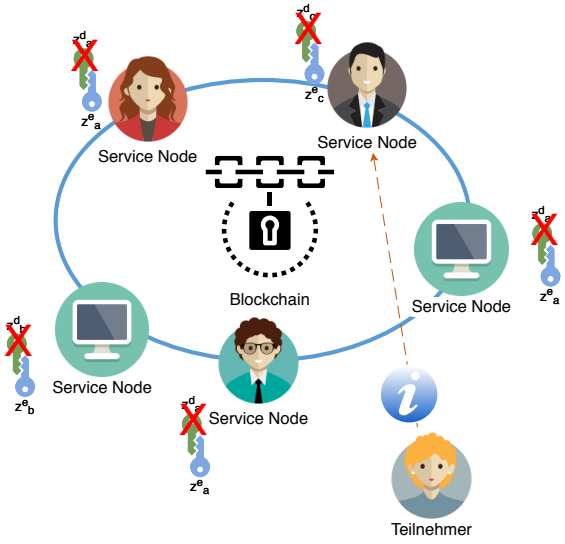


Abbildung 3.9.: Eigene Darstellung, Löschen von Schlüssel

Wie in der Darstellung 3.9 ersichtlich, ist die Aufgabe des *Service Node*, alle berechtigten Teilnehmer inklusive aller *Service Nodes* zu informieren, dass der Schlüssel für den angefragten Block zu löschen ist. Sie müssen dieser Aufforderung nachkommen. Durch diesen Vorgang kann das Löschen von Daten in einer Blockchain sichergestellt werden. Niemand hat danach die Möglichkeit, auf diesen Block Zugriff zu erlangen. Dies verhält sich wie das Löschen von Daten. Dabei reicht die Information aus, dass alle Beteiligten den Schlüssel löschen müssen. Um menschliche Fehler zu vermeiden, kann dieser Vorgang mittels *smart contract* automatisiert durchgeführt werden.

Sollte jemand dieser Aufforderung nicht nachkommen, hat der Teilnehmer das Recht, auf eine Beschwerde bei der Datenschutzbehörde. Die danach ausgeschriebene Geldbußen kann bis zu EUR 20 Millionen

oder im Fall eines Unternehmens bis zu 4% des gesamten Jahresumsatzes eines Geschäftsjahres betragen. Somit ist die Haftungsfrage bei Verstößen gegen die DSGVO geklärt. Tritt für eine Person ein Schaden auf, haftet der Verantwortliche der Daten, welcher dem *Service Node* entspricht. Hingegen ist der Verantwortliche nicht verpflichtet, im öffentlichen Gesundheitswesen und im öffentlichem Interesse liegenden Archivzwecken oder Forschungszwecken dem „Recht auf Löschung“ nachzukommen.

4. Fazit und Ausblick

Die empirischen Untersuchungen haben gezeigt, dass die Implementierung von kryptografischen Zugriffskontrollsystemen in einer Blockchain möglich ist. Dadurch kann die Sicherheit der Daten in einer Blockchain gesteigert werden, wodurch die Einsatzmöglichkeiten auf viele Anwendungsgebiete ausgeweitet werden können.

In Anlehnung an die herkömmlichen *Nodes* in einer Blockchain wurden ***Service Nodes*** entwickelt. Diese übernehmen, unter der Berücksichtigung von zwei neuen Konsensverfahren, die bisherige Funktion der *Nodes*, das *Mining* von Blöcken. Der Unterschied liegt darin, dass ***Service Nodes*** durch ein Konsortium aus den Teilnehmern gewählt werden. Zu ihren Aufgaben zählen neue kryptografische Schlüssel zu erstellen und neu zu berechnen sowie die Richtlinien der DSGVO zu erfüllen.
Durch den Einsatz der vorgestellten Verfahren *Shamir Secret Sharing* und *Cryptographic Access Control Solution for Smart Phones*, können Informationen in der Blockchain durch die symmetrische Kryptografie vor ungewünschten Zugriffen gesichert werden. Diese Verfahren werden in dem neuen Konsensverfahren ***proof of cryptography*** dazu benutzt, dass die Blöcke beim Hinzufügen in die Blockchain durch ***Service Nodes*** auf Echtheit überprüft werden. Nur sie haben das Recht, Blöcke hinzuzufügen und neue Schlüssel zu generieren.
Das zweite entwickelte Konsensverfahren ***proof of signature*** baut auf dem asymmetrischen kryptografischen Verfahren auf. Hier werden durch ***Service Nodes*** die Echtheit der hinzugefügten Blöcke überprüft und mit einer digitalen Signatur bestätigt. Diese Signatur kann von allen Teilnehmern auf Echtheit verifiziert werden. Zusätzlich lassen sich mit der asymmetrischen Kryptografie Leseberechtigungen umsetzen. Dadurch sind Blöcke vor ungewünschten Zugriffen geschützt.
Um das Vertrauen der Teilnehmer gegenüber der Blockchain weiter zu erhöhen, wird empfohlen, dass entweder eines der beiden Konsensverfahren oder beide in Kombination angewendet werden. Dadurch wird die Sicherheit der Daten in der Blockchain gesteigert. Die Verwendung von kryptografischen Verfahren für die Zugriffskontrolle in Blockchains ist sehr ökonomisch und benötigt gegenüber anderen Verfahren wenig Rechenleistung, wodurch sie sehr effizient ist. Des Weiteren entfällt durch die Implementierung von kryptografischen Zugriffskontrollsystemen die zentrale Steuerung der Rechte. Dies fügt

sich in die Philosophie der verteilten Blockchains optimal ein. Die Verwaltung der großen Anzahl geheimer Schlüssel für die kryptografischen Methoden wird durch den Einsatz von *Smartcards* für die entsprechende Blockchain gewährleistet. Somit sind diese sicher vor ungewünschten Zugriffen aufbewahrt.

Das „Recht auf Löschen“ in der DSGVO wird dadurch erfüllt, dass bei Anfragen der Teilnehmer ***Service Nodes*** bestehende Schlüssel löschen. Dabei werden die anderen Teilnehmer dazu aufgefordert, ihre dazugehörigen Schlüssel zu entfernen, um den Zugriff auf den Block zu unterbinden. Damit wird die Unveränderbarkeit der Blockchain umgangen und stellt keinen Widerspruch zur Datenlöschung dar. Durch ***Service Nodes*** wird ebenfalls die Informationspflicht und das Recht auf Auskunft der DSGVO eingehalten.
Die Blockchain der Zukunft, auch als ***Blockchain for Data Security*** bezeichnet, muss den Einsatz der neu definierten ***Service Nodes*** in Kombination mit den entwickelten Konsensverfahren ***proof of cryptography*** und ***proof of signature*** unterstützen. Diese neuartige Blockchain soll alle bereits bestehenden Blockchains ablösen. Durch die hierbei entstandene Sicherheit trägt die ***Blockchain for Data Security*** zur weiteren Verbreitung dieser Technologie bei. Die daraus resultierenden Vorteile sind die Erhöhung der Widerstandsfähigkeit gegen Datenverlust, die Integrität der Daten durch neuentwickelte Konsensverfahren und die Transparenz durch bereits bestehende etablierte kryptografische Verfahren sowie das gewonnene Vertrauen durch die hohe Sicherheit der Daten und die Erfüllung der DSGVO. Nebenbei können durch die effektiven Konsensverfahren die Effizienz und Geschwindigkeit zum Finden eines Konsenses gesteigert werden.

In weiterer Folge können die Ergebnisse aus dieser Arbeit als Basis für die Implementierung in eine Software verwendet werden. Durch die Programmierung der Methoden können die hier entwickelten Konsensverfahren auf ihren Einsatz und die praktische Tauglichkeit in der Blockchain überprüft werden. Dadurch kann sichergestellt werden, dass diese Verfahren den Praxistest in der Blockchain bestehen. Die Blockchain kann in Kombination mit *smart contracts* durch Automatisierung die menschliche Fehlerquelle weiter verringern.

Zusätzlich ist die Haftungsfrage in einer Blockchain bei der Berücksichtigung der DSGVO ungeklärt. Es bedarf einer weiteren intensiven Behandlung, ob für falsch programmierte Software die Software-Entwickler, das Konsortium oder die Betreiber der Blockchain haften.

Blockchain-Entwickler müssen sich darüber im klaren sein, dass eine fehlerhafte Implementierung der Software ihrerseits zum Datenverlust oder Datendiebstahl führt. Dadurch kann das Vertrauen der Teilnehmer in Hinsicht auf die Sicherheit der Blockchain verloren gehen und die Verwendung von Blockchains hinterfragt werden.

A. Berechnung

A.1. Shamir Secret Sharing

Anzahl der *Service Nodes*: $n = 5$

Schwellwert zur Wiederherstellung des Schlüssels: $k = 3$

Der geheimer Schlüssel enspricht dem Text $S = 16$.

$S_{hex} = 0x3136_{16}$

$S_{dec} = 12598_{10}$

Unter Berücksichtigung der Formel 3.1 die Primzahl $P = 12611$.

Um das Polynom 3.2 zu erstellen, wurde folgende Zufallszahlen gewählt:

$a_0 = S = 12598 \equiv$ geheimer Schlüssel.

$a_1 = 1324$

$a_2 = 7654$

$a_3 = 9765$

$a_4 = 10347$

$a_5 = 4594$

Somit gilt für 3.2:

$f(x) = 12598 + 1324 \cdot x + 7654 \cdot x^2$

Daraus werden folgende Teilschlüssel erzeugt:

$D_0 = (1, f(1) \mod p) = (1, 21576)$

$D_1 = (2, f(2) \mod p) = (2, 45862)$

$D_2 = (3, f(3) \mod p) = (3, 85456)$

$D_3 = (4, f(4) \mod p) = (4, 140358)$

$D_4 = (5, f(5) \mod p) = (5, 210568)$

Diese Schlüssel werden auf alle *Service Nodes* aufgeteilt.

Muss ein Basisschlüssel wieder zusammengestellt werden, kann das mit k *Service Nodes* anhand von 3.4 durchgeführt werden. Dabei werden drei zufällig gewählte *Service Nodes* ausgesucht:

$D_0 \equiv (x_0, y_0) = (1, 21576), D_1 \equiv (x_1, y_1) = (2, 45862), D_2 \equiv (x_2, y_2) = (4, 46786)$

$\frac{x-x_1}{x_0-x_1} \cdot \frac{x-x_2}{x_0-x_2} = \frac{x-2}{1-2} \cdot \frac{x-3}{1-3} = \frac{1}{2}(x-2)(x-3)$

$57536 - 91724 + 46786 = 12598$

Dabei entspricht 12598 dem geheimen Schlüssel.

Literaturverzeichnis

[1] B. Informationswirtschaft, "Blockchain #Banking," Bundesverband Informationswirtschaft, Tech. Rep., 2016, besucht am: 02.02.2019. [Online]. Verfügbar: https://www.bitkom.org/sites/default/files/file/import/161104-LF-Blockchain-final-2.pdf

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008, besucht am: 02.01.2019.

[3] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "Blockchain challenges and opportunities: a survey," *IJWGS*, vol. 14, pp. 352–375, 2018.

[4] A. Nayak and K. Dutta, "Blockchain: The perfect data protection tool," in *2017 International Conference on Intelligent Computing and Control (I2C2)*, June 2017, pp. 1–3, besucht am:.

[5] C. of European Union, "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation),," *Official Journal of the European Union*, vol. vol. L119, May 2016, besucht am: 02.01.2019. [Online]. Verfügbar: https://eur-lex.europa.eu/eli/reg/2016/679/oj

[6] D. Schmelz, G. Fischer, P. Niemeier, L. Zhu, and T. Grechenig, "Towards Using Public Blockchain in Information-Centric Networks: Challenges Imposed by the European Union's General Data Protection Regulation," in *2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)*, Aug 2018, pp. 223–228, besucht am: 02.01.2019.

[7] K. Panetta and M. J. Walker, "3 Trends Appear in the Gartner Hype Cycle for Emerging Technologies, 2016," Gartner, Inc., Tech. Rep., 2016, besucht am: 02.01.2019. [Online]. Verfügbar: https://www.gartner.com/smarterwithgartner/3-trends-appear-in-the-gartner-hype-cycle-for-emerging-technologies-2016

[8] F. Dai, Y. Shi, N. Meng, L. Wei, and Z. Ye, "From Bitcoin to cybersecurity: A comparative study of blockchain application and security issues," *Conference: Conference: 2017 4th International Conference on Systems and Informatics (ICSAI)*, pp. 975–979, 11 2017, besucht am: 23.04.1019.

[9] C. Welzel, K.-P. Eckert, F. Kirstein, and V. Jacumeit, "Mythos Blockchain: Herausforderung für den öffentlichen Sektor," Fraunhofer-Institut für Offene Kommunikationssysteme FOKUS, Wiesbaden, Tech. Rep., 2017, besucht am: 02.01.2019.

[10] N. Pohlmann, *BlockChain Idee, Konzepte, Mechanismen und Anwendungen*. Institut für Internet-Sicherheit – if(is) Westfälische Hochschule, Gelsenkirchen, 2017. [Online]. Verfügbar: https://norbert-pohlmann.com/app/uploads/2017/10/336-Blockchain-\T1\textendash-Idee-Konzepte-Mechanismen-und-Anwendungen-Prof.-Norbert-Pohlmann.pdf

[11] D. J. Yaga, P. M. Mell, N. Roby, and K. Scarfone, "Blockchain Technology Overview," Nov 2018, besucht am: 02.01.2019. [Online]. Verfügbar: https://www.nist.gov/publications/blockchain-technology-overview

[12] F. Gierschner, "Bitcoin and Beyond," *IEEE Communications*, 2016.

[13] F. Tschorsch and B. Scheuermann, "Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies," *IEEE Communications Surveys Tutorials*, vol. 18, no. 3, pp. 2084–2123, third-quarter 2016.

[14] L. Chen, S. Jordan, Y.-K. Liu, D. Moody, R. Peralta, R. Perlner, and D. Smith-Tone, "Report on Post-Quantum Cryptography," April 2016, besucht am: 02.01.2019. [Online]. Verfügbar: https://nvlpubs.nist.gov/nistpubs/ir/2016/NIST.IR.8105.pdf

[15] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-key Cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978. [Online]. Verfügbar: http://doi.acm.org/10.1145/359340.359342

[16] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage, "A Fistful of Bitcoins: Characterizing Payments Among Men with No Names," in *Proceedings of the 2013 Conference on Internet Measurement Conference*, ser. IMC '13. New York, NY, USA: ACM, 2013, pp. 127–140. [Online]. Verfügbar: http://doi.acm.org/10.1145/2504730.2504747

[17] A. Biryukov, D. Khovratovich, and I. Pustogarov, "Deanonymisation of clients in Bitcoin P2P network," *CoRR*, vol. abs/1405.7418, 2014, besucht am: 02.01.2019. [Online]. Verfügbar: http://arxiv.org/abs/1405.7418

[18] T. Lyons, L. Courcelas, and K. Timsit, *Blockchain and the GDPR*. The European Union Blockchain Observatory and Forum, Oct 2018, besucht am: 02.04.2019. [Online]. Verfügbar: https://www.eublockchainforum.eu/sites/default/files/reports/20181016_report_gdpr.pdf

[19] T. Kusber, S. Schwalm, C. Berghoff, and U. Korte, "Langfristige Beweiswerterhaltung und Datenschutz in der Blockchain," 09 2018.

[20] K. Panetta, "5 Trends Emerge in the Gartner Hype Cycle for Emerging Technologies, 2018," besucht am: 05.02.2019. [Online]. Verfügbar: https://www.gartner.com/smarterwithgartner/5-trends-emerge-in-gartner-hype-cycle-for-emerging-technologies-2018/

[21] S. Davidson, P. De Filippi, and J. Potts, "Economics of Blockchain," in *Public Choice Conference*, Fort Lauderdale, United States, May 2016, besucht am: 05.02.2019. [Online]. Verfügbar: https://hal.archives-ouvertes.fr/hal-01382002

[22] M. Rauchs and G. Hileman, *Global Cryptocurrency Benchmarking Study*, ser. Cambridge Centre for Alternative Finance Reports. Cambridge Centre for Alternative Finance, Cambridge Judge Business School, University of Cambridge, October 2017, no. 201704-gcbs. [Online]. Verfügbar: https://ideas.repec.org/b/jbs/altfin/201704-gcbs.html

[23] Anonymous, "How can Europe benefit from blockchain technologies?" Feb 2018, besucht am: 02.03.2019. [Online]. Verfügbar: https://ec.europa.eu/digital-single-market/en/news/how-can-europe-benefit-blockchain-technologies

[24] H. Graf, "Blockchain: Byzantinische Generäle und das CAP Theorem," Feb 2018, besucht am: 02.03.2019. [Online]. Verfügbar: https://blog.novatrend.ch/2018/01/22/blockchain-byzantinische-generaele-und-das-cap-theorem/

[25] A. V. D. M. Kayem, S. G. Akl, and P. Martin, *Adaptive Cryptographic Access Control (Advances in Information Security)*. Springer, 2010.

[26] S. D. C. di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, "A Data Outsourcing Architecture Combining Cryptography and Access Control," in *Proceedings of the 2007 ACM Workshop on Computer Security Architecture*, ser. CSAW '07. New York, NY, USA: ACM, 2007, pp. 63–69. [Online]. Verfügbar: http://doi.acm.org/10.1145/1314466.1314477

[27] A. A. Hasib and A. A. M. M. Haque, "A Comparative Study of the Performance and Security Issues of AES and RSA Cryptography," in *2008 Third International Conference on Convergence and Hybrid Information Technology*, vol. 2, Nov 2008, pp. 505–510.

[28] A. Harrington and C. Jensen, "Cryptographic Access Control in a Distributed File System," in *Proceedings of the Eighth ACM Symposium on Access Control Models and Technologies*,

ser. SACMAT '03. New York, NY, USA: ACM, 2003, pp. 158–165. [Online]. Verfügbar: http://doi.acm.org/10.1145/775412.775432

[29] E. Piller and F. M. de Rivas, "A New Decentralized Cryptographic Access Control Solution for Smart-phones," *Universal Journal of Communications and Network*, vol. 3, pp. 51–56, 2015.

[30] E. Piller and A. Westfeld, "Kryptografisches Zugriffskontrollsystem für mobile Endgeräte," in *syss-ec IT-Security und IT-Management*, ser. CCS '16, 2015, pp. 1–11.

[31] V. Bunimov and M. Schimmler, "Completely Redundant Modular Exponentiation by Operand Changing." in *In Proceedings of the 2005 International Conference on Computer Design*. Las Vegas, Nevada, USA: CDES 2005, 01 2005, pp. 224–232.

[32] G. Barthe, D. Pointcheval, and S. Zanella Béguelin, "Verified Security of Redundancy-free Encryption from Rabin and RSA," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, ser. CCS '12. New York, NY, USA: ACM, 2012, pp. 724–735. [Online]. Verfügbar: http://doi.acm.org/10.1145/2382196.2382272

[33] A. Shamir, "How to Share a Secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979. [Online]. Verfügbar: http://doi.acm.org/10.1145/359168.359176

[34] Z. Shen, L. Li, F. Yan, and X. Wu, "Cloud Computing System Based on Trusted Computing Platform," in *2010 International Conference on Intelligent Computation Technology and Automation*, vol. 1, May 2010, pp. 942–945.

[35] "TPM Main: Part 1 Design Principles, Trusted Computing Group (TCG) Specification 1.2," March 2011, besucht am: 23.04.2019. [Online]. Verfügbar: https://trustedcomputinggroup.org/wp-content/uploads/TPM-Main-Part-1-Design-Principles_v1.2_rev116_01032011.pdf

[36] Z. Shen and Q. Tong, "The security of cloud computing system enabled by trusted computing technology," in *2010 2nd International Conference on Signal Processing Systems*, vol. 2, July 2010, pp. V2–11–V2–15.

[37] O. Ogundele, P. Zavarsky, R. Ruhl, and D. Lindskog, "The implementation of a full EMV smartcard for a point-of-sale transaction," in *World Congress on Internet Security (WorldCIS-2012)*, June 2012, pp. 28–35.

[38] E. Rusek and T. Schweiger, "Praxisleitfaden zur Umsetzung der DSGVO," April 2018, besucht am: 02.04.2019. [Online]. Verfügbar: https://media.wko.at/epaper/Datenschutzgrundverordnung/Umsetzung/#0

[39] D. Atkins, W. Stallings, and P. Zimmermann, "PGP Message Exchange Formats," Internet Requests for Comments, RFC Editor, RFC 1991, August 1996, besucht am: 02.03.2019. [Online]. Verfügbar: http://www.rfc-editor.org/rfc/rfc1991.txt

[40] S. J. MACKINNON, P. D. TAYLOR, H. MEIJER, and S. G. AKL, "An Optimal Algorithm for Assigning Cryptographic Keys to Control Access in a Hierarchy," *IEEE Transactions on Computers*, vol. C-34, no. 9, pp. 797–802, Sep. 1985.

[41] S. G. Akl and P. D. Taylor, "Cryptographic Solution to a Problem of Access Control in a Hierarchy," *ACM Trans. Comput. Syst.*, vol. 1, no. 3, pp. 239–248, Aug. 1983. [Online]. Verfügbar: http://doi.acm.org/10.1145/357369.357372