Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Entwurf vom
23. September 2024

Bachelorarbeit

Privatsphärewahrendes Anreiz- und Betrugserkennungssystem im Datentreuhandmodell

vorgelegt von

Knut Hoffmeister
Matrikelnummer 7509085
Studiengang Software System Entwicklung

MIN-Fakultät
Fachbereich Informatik

eingereicht am 23. September 2024

Betreuer: Kevin Röbert

Inhaltsverzeichnis

Abstract

1 Einleitung

In der heutigen Zeit wird der fachgerechte Umgang mit Daten jeglicher Form, zunehmend wichtiger. Viele der großen Player wie Facebook oder Google, machen ihr Hauptgeschäft mit dem Verwenden von Nutzerdaten zu gewerblichen Zwecken, wie bspw. targeted Ads [Fac24] [Goo24]. Und obwohl das Misstrauen eines Nutzers, solch eines großes Unternehmens gegenüber berechtigt ist, benötigen diese die gesammelten Daten auch, um neue Technologien zu entwickeln. Jedoch hat der Nutzer, der diese Daten generiert, meist keine Einsicht darüber, wer seine Daten verwendet und wofür diese zum Einsatz kommen. Ein potentieller Lösungsansatz für dieses Problem, ist die Verwendung von Datentreuhändersystemen. In einem Datentreuhändersystem, kann ein Endnutzer der Daten generiert zum einen seine Daten verschlüsselt bei einem Treuhänder lagert. Zum anderen ist der Treuhänder ein Mediator zwischen dem Endnutzer und einem Unternehmen, welches an den Daten interessiert ist. Möchte nun ein Unternehmen die Daten für ihre Zwecke verwendet, so kann dieses bei dem Treuhänder Daten anfordern. Daraufhin kontaktiert der Treuhänder den Nutzer, von dem die Daten stammen und fragt diesen, nach seiner Zustimmung. So kann der Nutzer genau einsehen, wer seine Daten verwenden möchte und kann unerwünschte Benutzung unterbinden. Das Ziel liegt hierbei vor allem darauf den Nutzer und seine Privatsphäre so gut wie möglich zu schützen.

In dieser Arbeit sollen, um die Motivation zur Benutzung eines Datentreuhänder zu erhöhen, zwei Forschungsfragen beantwortet werden: Wie kann ein privatsphäreschützender Anreiz zur Benutzung eines Datentreuhändermodells geschaffen werden? Und: Wie kann dieser Anreiz gegen Missbrauch geschützt werden? Aktuell ist der einzige Anreiz zur Benutzung eines Datentreuhändermodells für den Nutzer, der Schutz der Privatsphäre durch die Kommunikation über den Treuhänder. Um einen weiteren Anreiz zu bieten, soll hier ein Ansatz vorgestellt werden, welcher den Datengebenden für seine zur Verfügung gestellten Daten, angemessen entlohnt. Dies geschieht durch eine Transaktion von dem Datennutzenden an den Datengebenden, die trotz des Austausches von Zahlungsinformationen, die Privatsphäre des Datengebenden und dessen Identität so weit wie möglich schützt. Hierfür könnte bspw. eine leicht erweiterte Version des GNU Taler Zahlungssystem verwendet werden 3.2, um den Zahlungsverkehr zwischen den Parteien zu ermöglichen. Um dieses System vor Missbrauch zu schützen, wird ein Reputationssystem eingesetzt, um zu verhindern, dass ein Datengebender wertlose Daten oder qualitativ niedrig wertige Daten, in großer Masse bereitstellen kann, um so das System auszunutzen. Es soll zum einen, gleich wie der Bezahlvorgang, die Identität des Datengebenden schützen. Gleichzeitig soll es den Datennutzenden, der die Bewertung ausstellt, davon abhalten, das System durch mehrfaches Bewerten auszunutzen. Auch Betrugsversuche des Datengebenden sind nicht zu vernachlässigen, wie bspw. die Neuanmeldung eines Nutzers mit schlechter Reputation, um diesen Wert zurückzusetzen.

Dafür wird in dieser Arbeit der aktuelle Stand der Forschung in dem vorliegenden Kontext evaluiert und geprüft was bereits anwendbar und wo noch Lücken zur gewünschten Verwendung bestehen. Mit den gesammelten Forschungsergebnissen werden daraufhin mehrere Konzepte präsentiert, um das gerade genannte Ziel in das Treuhändermodell zu integrieren. Diese Konzepte werden des weiteren in ein bestehendes Treuhändersystem eingebaut, um konkrete Vergleichsgrundlagen zu erhalten. Bei dem hierfür vorgesehenen System, handelt es sich um das Tresor-Projekt der Universität Hamburg [Ham22].

2 Hintergrund

Die zunehmende Digitalisierung unseres Alltags ist unabstreitbar [ZDF23]. Viele der täglichen Aktivitäten hängen stark mit ihr zusammen. Sei es den schnellsten Weg zu Arbeit finden mit Google Maps, das kontaktlose Bezahlen an der Kasse mit GooglePay oder Applepay, die Benutzung von Social Media zur Unterhalten oder der Onlinehandel über Anbieter wie Amazon. Sie alle liefern Komfort der durch die zunehmende Verwendung von Computern ermöglicht wird, welche im Hintergrund Unmengen an Daten für ihre Berechnungen verwenden. Diese Daten stammen meist von den Benutzern selbst. Bspw. die Standortdaten für die Berechnung potentieller Staus im Straßennetz [han17] oder die Unterhaltungsinteressen basierend auf Watchtime von bestimmten Social Media Inhalten.
Aufgrund des ständig wachsenden Markts für neue Digitaltechnologien ist auch die Nachfrage nach Daten über die letzten Jahr in Höhe gestiegen. Über das letzte Jahrzehnt haben Daten, Öl als wertvollste Ressource abgelöst [FK20]. Während 2008 die vier weltweit wertvollsten Unternehmen Ölkonzerne waren, waren 2018 bereits die 7 wertvollsten Unternehmen Internet- und Technologiefirmen.

2.1 Herkömmliche Datenkommunikation

Unter anbetracht des hohen Wertes von Daten sind viele Unternehmen verständlicherweise sehr zurückhaltend was den Austausch betrifft. Schließlich beutet eine eigene Sammlung von Daten ein potentielles Verkaufsgut. Laut einer durch die Bundesregierung aufgegriffenen Studie von Fedkenhauer et al. geben zwar viele der befragten Unternehmen an, Aktivitäten im Bereich des Datenaustausches zu betreiben, allerdings umfasst das in 83% der Fällen den Austausch von Daten mit Kundinnen und Kunden. 53% der Unternehmen teilen ihre Daten mit Lieferantinnen und Lieferanten. Ein noch kleinerer Anteil von 21% teilen ihre Daten mit Unternehmen aus der gleichen oder anderen Branchen und nur 15% teilen sie mit Wettbewerbern [Bun21].

Aus der kapitalgetriebenen Sicht eines Unternehmens besitzt das Teilen der eigenen Daten keinen direkten Nutzen. Da ein Unternehmen seine eigenen Gewinnmaximierung anstrebt, ist das Teilen von Daten eher ein Nachteil, da fremde Unternehmen mit den selbst gesammelten Daten ihre Produkte qualitativ erweitern können. Dadurch werden entweder andere Wettbewerben oder branchenfremde Unternehmen in ihrem Marktwert gefördert, was zu dem sinken des eigenen Marktwertes führt.
Diese protektive Herangehensweise kann allerdings auch der Gewinnmaximierung im Weg stehen. Im Fall von einem direkten Tausch an Daten können beide Parteien einen Profit aus der Interaktionen erwirtschaften. Die Bundesregierung selbst schreib in [Bun21] das kaum Datenkooperationen zwischen staatlichen und wirtschaftlichen Akteurinnen bestehen, obwohl die staatlich gesammelten Daten eine Grundlage für wirtschaftliche Innovation sein könnten. Im Gegenzug könnten die Daten von Unternehmen dem Staat bei der Sicherstellung seines Versorgungsauftrages, der Daseinsvororge und der Wahrung öffentlicher Schutzgüter helfen. Dies ist eine optimale Situation für die Verwendung eines Datentreuhänders.

2.2 Datentreuhänder

Ein Datentreuhänder ist ein neutraler vertrauenswürdiger Vermittler von Daten eines Datengebenden zu einem Datennutzenden. Er hat selbst kein kommerzielles Interesse an der Verwertung der Daten und agiert vergleichbar zu einem Notar strickt für den Datengeber. Seine Hauptaufgaben umfassen meist

die Kontrolle von Zugriffsrechten, das Kontrollieren von Einhaltung der Datenschutzrichtlinien, sowie das verschlüsseln oder anonymisieren von Datenbeständen. In speziellen Fällen kann ein Datentreuhänder auch die Auswertung von Daten vornehmen. [Bun21][Ric]

Da wie bereits angeführt viele Unternehmen die Weiterleitung ihrer Daten vermeiden, ist unter der Annahme eines etablierten Datentreuhänders ein deutliche größerer Datenbestand verfügbar. Bereits heute - vor einer großen Etablierung von Datentreuhänder - verspricht das Konzept einige gesellschaftliche Vorteile: [Ric]

1. Durch sie können Datenbestände besser vernetzt werden und Zusammenhänge hergestellt werden, welche zu Innovationen führen.

2. Der Wettbewerb unter Firmen wird gestärkt da mit besser zugänglichen Daten auch kleinere Unternehmen die kein Datenmonopol besitzen ihre Produkte aufwerten können.

3. Der individuelle Endnutzer erhält mehr Kontrolle und Transparenz über die Speicherung und Verwendung seiner Daten.

Allerdings ist das Verständnis eines Datentreuhänders nicht eindeutig. Jürgen Kühling beschreibt den Datentreuhänder als “ein schillerndes Wesen. Jeder kennt ihn, jeder setzt ganz eigene Hofnungen in ihn – und jeder stellt sich doch etwas anderes unter ihm vor“ [KB21]. Obwohl die Technologie eines Datentreuhänders bereits seit Jahren existiert und verwendet wird [Har18] ist es nicht gelungen eine konkrete allumfassende Definition für die Technologie zu finden.

2.2.1 Definition

Die allgemeingültigste Definition stammt aus der Rechtswissenschaft und bezieht sich auf die Treuhandschaft im Allgemeinen. *[Treuhandschaften sind ein] Rechtsverhältnis, bei dem eine natürliche oder juristische Person (Treugeber) einer zweiten Person (Treuhänder) ein Recht unter der Bedingung überträgt, von diesem Recht nicht zum eigenen Vorteil Gebrauch zu machen. [...] Gemeinsames Charakteristikum ist die Uneigennützigkeit und Vertrauenswürdigkeit bei der Wahrnehmung fremder Interessen bzw. die uneigennützige Ausübung von amtlichen Befugnissen"*[Pro13]. Diese Treuhandschaft wurde in der Vergangenheit verwendet, um bspw. Ländereien zu verwalten und im Namen einer lokalen Gemeinschaft Entscheidung zu treffen. [Har18]. In dem Fall eines Datentreuhänders bedeutet dies konkret, dass eine Datengebende Person beim bereitstellen ihrer Daten den Datentreuhänder dazu ermächtigt, über die Daten zu verfügen. Darunter fällt unter anderen auch die Weitergabe der Daten, solange die im Sinne des Datengebers ist.

2.2.2 Verschieden Modelle

Insgesamt lassen sich alle bis zum heutigen Zeitpunkt in Betrieb genommenen oder geplanten Datentreuhandsysteme wie folgt kategorisieren. Zum einen besteht die Einteilung in Customer to Business oder Business to Business Systeme und zum anderen die risikobasierte Einteilung nach Zentralen/Dezentraler Datenspeicherung und Freiwilliger/Verpflichtender Nutzung. (siehe 2.1)

Die Unterscheidung zwischen Customer to Business und Business to Business Datentreuhändern basiert ausschließlich auf den interagierenden Parteien. Im Falle einer B2B Interaktion kommunizieren zwei Unternehmen die vorhandenen Daten miteinander. In diesem Fall kommt es häufig vor, dass eines der Unternehmen durch eine staatliche Behörde dargestellt wird. Bei den gespeicherten Daten handelt es sich meist um personenbezogene Daten. Aufgrund dessen befassen sich B2B Datentreuhänder häufig mit der Pseudonymisierung und der Verwaltung der bereitgestellten Daten. Sie sind unter anderem im Gesundheitswesen viel vertreten. [Bla+20] C2B Systeme umfassen solche, bei denen einen Endnutzer die Daten generiert und diese an ein Unternehmen zur weiteren Benutzung freigibt. Ihre Aufgabe ist hauptsächlich die Unterstützung des Nutzers bei der gerechten Weiterverarbeitung seiner

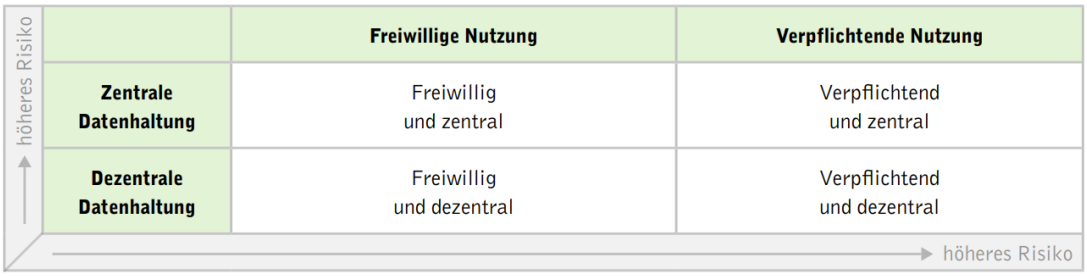

Abbildung 2.1: Risikobasierte Unterscheidung von Datentreuhandmodellen [BS21b]

Daten. [Bla+20] Hier sind dementsprechend die Pseudonymisierung der Daten sowie die Einhaltung von Datenschutzrechtlinien und einheitlicher Standarts die Hauptziele des Datentreuhänders.

Des weiteren lassen sich Datentreuhandsysteme anhand ihrer Datenspeicherung sowie Nutzung kategorisieren und Risikotechnisch bewerten. Die zentrale Speicherung der Daten bietet einige Vorteile für den Treuhänder. Sie ermöglicht es Daten vor zu verarbeiten, zu analysieren und Datenverarbeitende von dem direkten Zugang der Daten auszuschließen. Allerdings birgt die zentrale Datenspeicherung ein enormes Risiko der Datensicherheit, da hier ein single point of failure entsteht. Bei einem Angriff auf einen solchen Datentreuhänder fällt es einem Angreifer somit leichter eine große Menge an Daten zu stehlen. Bei einer dezentralen Speicherung werden die Daten durch den Treuhänder pseudonymisiert oder ganz anonymisiert und bei dem Datengeber gelagert. Aus diesem Grund bietet eine dezentrale Datenspeicherung mehr Sicherheit vor Diebstahl. Sie zieht allerdings auch eine eingeschränkte Verarbeitung und Analyse der Daten mit sich und erhöht die Komplexität der Verwaltung.

Es besteht eine weitere Unterteilung in Datentreuhandsysteme, dessen Benutzung freiwillig ist oder verpflichtend ist. Dabei fällt der größte Anteil an Systeme unter die freiwillige Benutzung. Es gibt jedoch auf Szenarien in denen die Verwendung einer Datentreuhand verpflichtend ist. Ein Beispiel hierfür wäre das Krebs- und Transplantationsregister aus dem medizinischen Bereich. Das Risiko steigt bei verpflichtenden Systemen, da sie meist einen wichtigen Bestandteil der Kommunikation ausmachen der nicht umgangen werden kann. Somit sind die potentiellen Schäden die bei einem Angriff entstehen können höher als in einem freiwilligen System.

2.3 Anwendungsfälle

Das mögliche Spektrum an Anwendungsfällen ist denkbar breit. In beinahe jedem Bereich der eine große Menge an Daten benötigt oder verwaltet ist die Verwendung eines Datentreuhänders angedacht. Beispiele dafür sind:

- **Patientendaten** Der Datentreuhänder sorgt für eine pseudonymisierung von Patientendaten zur Bereitstellung an Forschungseinrichtungen. Hierbei behält der Patient die Hoheheit über seine Daten und kannst selbst entscheiden mit wem seine Daten geteilt werden.
- **Autonomes Fahren** Beim autonomen Fahren werden enorm viele Daten generiert, welche seit 2017 per Gesetz gespeichert werden müssen [Bun]. Leider ist die Zugehörigkeit der Daten rechtlich weder dem Autohersteller noch dem Autoinhaber zuzuschreiben [Ric]. An dieser Stelle kann ein Datentreuhänder die Kommunikation erleichtern und exklusiven Zugang von bspw. Versicherungen oder Automobilkonzernen ausschließen.
- **E-Government** Durch die Verwendung eines Datentreuhänders können bei der behördlichen Verwaltung von Bürgerdaten große Fortschritte erzielt werden. Unter anderem müssen so notwendige Informationen aus anderen Registern für Verwaltungsvorgänge nicht mehr vom Bürger bereitgestellt werden. Der Bürger gibt lediglich seine Einwilligung zum Abruf der Daten zu Beginn

des Verwaltungsvorgangs. Auf diese Weise können Verwaltungsvorgänge erheblich effizienter Ablaufen.

- **KI-Datenpools** Eines der größten Probleme bei Entwicklung von KI-Software ist der Zugang zu einer ausreichend großen Menge an Trainingsdaten. Ein Datentreuhänder kann solche Daten die zur Verwendung für KI Training freigegeben wurden sammeln und pseudonymisiert an mehrere Interessierte verteilen. Dadurch entsteht ein einheitlicher Zugang zu Trainingsdaten der gleichzeitig nur freigegebene Daten beinhaltet und so rechtlichen Streit über das Copyright aus dem Weg geht.

- **Industrie** In der Industrie besteht eine hohe Abhängigkeit von Warenbewegungen, seien es in Lieferketten, Logistik oder Handel. Durch die Verwendung eines Datentreuhänders können diese Informationen pseudonymisiert an Warenempfänger weitergegeben werden. Vor allem in diesem Bereich besteht durch die Zusammentragung an Lieferinformationen in Kombination mit Algorithmen der Graphentheorie ein großes Innovationspotential.

- **PIMS** Personal Information Management Systeme befassen sich grundsätzlich mit der Wahrung von personenbezogenen Daten und bietet ihren Nutzern mehr Kontrolle über diese. Datentreuhänder sind für solche Systeme vor allem von Vorteil da sie im Umgang mit personenbezogenen Daten dem Nutzer wieder die Kontrollen über seine Daten zurückgeben.

[BS21a][BS21b][Bun]

Der erste Datentreuhänder entstand einer bereits im Jahre 2006 in England. Seit dem das Thema Datentreuhänder in den letzten Jahren immer mehr zum Trend wurde [Ric]. Die “UK Biobank“ ist eine biomedizinische Datenbank die sowohl medizinische Daten als auch biologische Proben von einer halben Millionen Teilnehmer aus Großbritanien speichert[Har18]. Die gespeicherten Daten sind pseudonymisiert und werden ausschließlich an Forscher im Feld der Medizin weitergegeben, was die UK Biobank zu einem Paradebeispiel für einen Patientendatentreuhänder macht.

2.4 Bestehende Anreize

Generell sei gesagt, dass die Verwendung eines Datentreuhänders keine direkten Nachteile mit sich bringt. Wenn Daten ohnehin geteilt werden oder werden müssen, so behält der Nutzer bei der Speicherung der Daten über einen Datentreuhänder mehr Kontrolle über die Verwendung seiner geteilten Daten verglichen mit dem direkten Teilen mit Unternehmen. Durch den Datentreuhänder wird ihm ermöglicht zu einem beliebigen Zeitpunkt das weitere Teilen seiner Informationen einzustellen. Vermutlich teilen deswegen Privatpersonen ihre Daten lieber mit Datentreuhändern als auf direktem Weg. [TRE24]

Allerdings gibt es nur wenige Vorteile die die freiwillige Benutzung eines Datentreuhänders reizvoll machen. Bei freiwilligen C2B Datentreuhändern, hängt die Entscheidung für oder gegen die Nutzung des Treuhänders beim Nutzer. Es ist also an ihm abzuwägen ob die Verwendung ausreichend Vorteile liefert. Im Fall von PIMS Treuhänder wird die Verwendung von manchen als erstrebenswert angesehen, da der Nutzer mehr Kontrolle über die Verbreitung von persönlichen Daten erhält und selbst entscheiden kann, mit wem diese geteilt werden. Die Erkenntnissen von Jai et al. [JK16] zeigen hingegen, dass vor allem jüngere Erwachsene weniger Wert auf den Schutz ihrer persönlichen Daten legen.

Allerdings gibt es keinen direkten Anreiz zur Verwendung einer solchen Software oder der Weitergabe der persönlichen Daten, da nur das Business von den gesammelten Daten einen Mehrwert hat. Der Nutzer der seine Daten freigibt erhält keine Kompensation in irgendeiner Form. Folglich kann es für einen Datentreuhänder dessen Verwendung freiwillig ist mühsam sein neue Nutzer zu gewinnen und die Technologie als solche auszubauen.

3 Anforderungen und verwandte Arbeiten

3.1 Anforderungen

Da eine Lösung zur Verwendung durch einen Datentreuhänder konzeptioniert ist, decken sich die Anforderungen an eine mögliche Lösung stark mit denen eines üblichen Datentreuhänders. Im Grunde kann zwischen funktionalen und nichtfunktionalen Anforderungen unterschieden werden.

3.1.1 Funktionale Anforderungen

1. Ein Bezahlsystem ermöglicht es einem Datengebenden Geld für seine Daten zu erhalten.
2. Die Präsenz eines Reputationswertes gibt den Datennutzenden vor Erwerb der Daten eine Einschätzung über deren Qualität.
3. Nach Abschluss der Transaktion kann ein Datennutzender den entsprechenden Datengebenden aufgrund der Qualität der übermittelten Daten bewerten.
4. Ein Datengebender muss eine Bewertung seiner Daten ermöglichen.
5. Ein Datennutzender kann pro Austausch nur genau eine Bewertung für einen Datengebenden abgeben. Mehrfache Bewertungen ist nur im Fall von mehrfachem Erwerb möglich.

3.1.2 Nicht funktionale Anforderungen

1. **Anonymität** Die Identität des Datengebenden darf durch den Austausch von Zahlungsmitteln oder durch dessen Reputation nicht nachverfolgbar sein.
2. **Zeitsensitivität** Der Bezahlvorgang muss in vernachlässigbarer Zeit geschehen.
3. **Skalierbarkeit** Die Rechenzeit des Systems soll bei linear steigender Menge an Datengebenden und Datennutzenden auch mit linearem Zeitaufwand zunehmen.
4. **Vertraulichkeit** Die kommunizierten Daten dürfen nicht durch unbefugte Dritte ausgelesen werden können.
5. **Integrität** Die kommunizierten Daten dürfen nicht unbemerkt durch unbefugte Dritte verändert werden.

3.2 GNU Taler

Das im Paper “Enabling Secure Web Payments with GNU Taler“ von J. Burdges et al. eingeführte Zahlungssystem GNU Taler, ist ein elektronisches Online-Zahlungssystem, das Datenschutz für Customer und Mechanismen zur steuerlichen Nachverfolgung für Merchants bietet [Bur+16]. Es verwendet einen Exchangeservice, um Münzen mithilfe von blinden Signaturen zwischen Nutzern und Händlern zu transferieren. Im Folgenden werden diese Münzen als Taler bezeichnet, um Verwirrung zu vermeiden. Das System basiert auf 4 übergeordneten Rollen, dessen Interaktion grob in Abbildung 3.1 skizziert ist. Der Customer möchte ein Gut oder eine Dienstleistung bei dem Merchant erwerben und bezahlt diesen dafür mit Talern, welcher er beim Exchange erworben hat. Der Merchant kann die erhaltenen Taler

wieder beim Exchange für herkömmliche Währungen eintauschen. Ein Auditor überprüft währenddessen die Liquidität des Exchange, um sicherzustellen, dass dieser auch bei Datenverlust von Taler noch in der Lage ist allen Beteiligten, Auszahlungen zu ermöglichen.

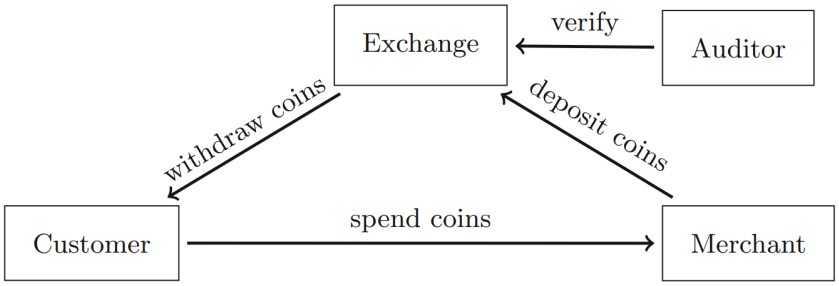

Abbildung 3.1: Grundlegender Ablauf des GNU Taler Systems [Bur+16]

Taler abheben. Damit ein Customer Geld auf sein Wallet laden kann, muss er sich zuerst bei seiner Bank anmelden. Sollte die Bank GNU Taler native unterstützen, so kann der Customer in einem Formular eine Summe auswählen, die er in Taler übertragen möchte und einen Exchangeservice, über welchen der Tausch abgewickelt wird. Nachdem der Customer die Transaktion bestätigt hat, wird die ausgewählte Summe transferiert, der Exchangeservice signiert die equivalente Summe an Talern und überträgt diese in das Wallet des Customers.

Taler ausgeben. Gehen wir von der Situation aus, dass eine Customer bei einem Merchant (hier ein Onlineshop), etwas erwerben möchte. Nach der Auswahl des Produktes und GNU Taler als Zahlungsmittel, erstellt der Merchant einen Zahlungsvertrag, der Details wie den Gesamtpreis, mögliche offene Umwandlungsgebühren und akzeptierte Exchangeservices beinhaltet und sendet diesen an das Wallet des Customers. Wenn der Customer daraufhin die Zahlung übermittelt, so leitet der Merchant die erhaltenen Taler direkt an den Exchange weiter. Wenn der Exchange den Eingang bestätigt, so kann der Merchant dem Customer die Transaktion bestätigen und der Kauf ist somit abgeschlossen.

In dem Treuhändermodell eignet sich GNU Taler aber nur teilweise als Zahlungssystem. Der Datengebende stellt dabei den Merchant da, der seine Daten als digitales Gut anbietet. Der Datennutzende stellt hier die Rolle des Customers dar und möchte diese Daten erwerben. Sollte der Datengebende der Anfrage des Datennutzende zustimmen, so würde er einen Zahlungsvertrag formulieren, um den Anspruch auf seine Vergütung zu formalisieren. Allerdings soll der Datengebende im Datentreuhändermodell so gut wie möglich vor Informationsgewinnung geschützt werden. Bei dem von GNU Taler vorgeschlagenen Prozess wird jedoch die Identität des Datengebenden nachverfolgbar, während der Datennutzenden anonym bleibt. Zusätzlich besteht eine direkte Kommunikation zwischen Datengebenden und Datennutzenden, was weitere schützenswert Information über die Identität des Datengebenden preis gibt.

3.3 Bitcoin

Die Idee hinter Bitcoin entstand 2008 durch eine Person oder Gruppe mit dem Namen Satoshi Nakamoto. Heute ist Bitcoin die mit Abstand weit verbreiteteste Onlinewährung weltweit [Sch24]. Sie basiert auf einem verteilten öffentlichen Register das alle Transaktionen für jeden pseudonymisiert einsehbar macht und speichert. Dieses Register ist auch als Blockchain bekannt.

Die Blockchain ist eine Kette an Blöcken, die jeweils den Hash des vorherigen Blocks und neue Transaktionen speichert. Durch den Hash des vorherigen Blocks entsteht die Kette in der jeder Block Informationen über seinen Vorgänger speichert. Ein neuer Block entsteht durch einen sogenannten proof of work. Dieser ist das Wissen über eine Zufallszahl, deren Hash mit x Null bits beginnt, welche ausschließlich durch das wiederholte ausprobieren von Zahl gefunden werden kann. Dieser Vorgang wird als Bitcoin Mining bezeichnet. Sobald eine solche Zahl gefunden wird, kann der aktuelle Block

abgeschlossen werden und alle folgenden Transaktionen werden in dem nächsten Block gespeichert [Nak08].

Der Algorithmus der bestimmt wie schwer das erbringen des proof of works ist, kann jederzeit dynamisch an die Rechenleistung der Miner angepasst werden so dass im Schnitt alle 10 Minuten ein neuer Block gefunden wird [Sch24]. Um also eine vergangene Transaktion auf der Blockchain zu verändern, müsste ein Angreifer für jeden darauffolgenden Block selbst einen proof of work berechnen. Ein Angreifer kann deswegen nur eine vergangene Transaktion ändern, wenn er alleine mehr Rechenleistung besitzt als alle anderen Miner zusammen.

In dem Moment in dem ein Block abgeschlossen und der nächste begonnen wird, sind alle Transaktionen auf dem abgeschlossenen Block ein Mal bestätigt. Das bedeutet das keine der Transaktionen einen Coin zum zweiten Mal ausgibt. Dies ist eine essentielle Sicherheitsmaßnahme von Blockchain Kryptowährungen da ohne Blockchain dezentralisiert ohne neutralen Überprüfer funktioniert. Es ist also an alle Beteiligten die pseudonymisierten Transaktionen des Block zu überprüfen. Sollte ein Angreifer die Zufallszahl finden und vor Abschluss des Block noch eine doppelte Ausgabe hinzufügen so kann diese Transaktion erst innerhalb des nächsten Block festgestellt werden, was zeigt dass nur weil eine Transaktion bestätigt ist, sie nicht direkt vertrauenswürdig ist. Im Fall von Bitcoin wird empfohlen 6 weitere Blöcke abzuwarten bis eine Transaktion wirklich abgeschlossen ist []. In Kombination mit einer durschnittlichen Berechnungsdauert von 10 Minuten dauert es also ca. eine Stunde bis ein Transaktion auf der Blockchain abgeschlossen ist. Durch den enormen Rechenaufwand der benötigt wird um einen Block abzuschließen, kann grob überschlagen werden wie viel Energie in eine einzelne Bitcointransaktion fließt. In [Dig23] wird eine Transaktion auf 703,25 kWh geschätzt, was ca 470.000 VISA Transaktionen entspricht.

Allein die lange Bestätigungsdauer einer Bitcointransaktion schließt es bereits für die Verwendung in diesem System aus, da eine Stunde nicht mehr als vernachlässigbare Zeit angesehen werden kann und somit nicht Zeitsensitiv ist. Hinzu kommt der gigantische Stromverbrauch, welcher die Skalierbarkeit von Bitcoin selbst in Frage stellt.

3.4 Ethereum

Ethereum ist genauso wie Bitcoin blockchainbasiert. Ein gängiger Irrtum ist jedoch, dass Ethereum selbst keine Währung ist. Ethereum wurde 2013 von Vitalik Buterin (erfunden?) und ist eine Plattform die es ermöglicht Entwicklern Anwendung auf der Blockchain zu entwickeln. Es liefert ein Turing-vollständige Programmiersprache die es Entwicklern erlaub eigene Währungen in unter 20 Zeilen Code zu schreiben [But+13]. Sämtlich dort geschrieben Währungen oder sogenannte Smart Contracts werden über die Ethereum Blockchain pseudonymisiert öffentlich zur Verfügung gestellt.

Durch die Verwendung der Blockchain decken sich einige Eigenschaften mit Bitcoin. Es besteht genauso aus einer Reihe an Blöcken, welche kontinuierlich ihren Vorgänger referenzieren. Bis 2022 nutzte Ethereum ebenfalls einen proof of work Ansatz. Doch seit 2022 basiert Ethereum einem proof of stake Konzept [TEA24]. Das bedeutet dass es eine Gruppe an Validierern gibt, welche die Transaktionen innerhalb der Blöcke überprüfen. Um ein Validierer zu werden, muss ein Starteinsatz von 32 Ether gezahlt werden. Alternativ kann sich ein Nutzer einem Validiererpool anschließen und einen kleineren Starteinsatz zahlen. Dafür muss er jedoch die erzielten Gewinne teilen.

Die Aufgabe eines Validierers ist es unzulässige Transaktionen festzustellen. Im Fall eines Angriffs auf einen Block, wird der Block von Gasper (Einer Mischung des Casper-FFG Protokolls und LMD Ghost Algorithmus [But+20].) markiert. Anschließend entscheiden die Validierer ob der Block zugelassen oder blockiert werden soll. Validierer die sich bösartig Verhalten werden dadurch bestraft, dass ihr Starteinsatz nach und nach “verbrannt“ wird. Mit verbrannt ist gemeint, dass der Einsatz an ein Wallet ohne private key gesendet wird, was die Coins unwiederruflich zerstört. Durch diesen alternativen

Ansatz kann der große Rechenaufwand von Bitcoin umgangen werden.

Die Verwendung einer Blockchain stellt hier wieder das Problem der Zeitsensitivität. Zwar dauert das Erstellen eines Block bei Ethereum nur 12 Sekunden [YCH]. Dafür werden allerdings eine Mindestanzahl von 30 Blocks empfohlen bevor eine Transaktion für gültig erklärt wird. Somit entsteht eine Wartezeit von ca 6 Minuten, was bei weitem besser ist als Bitcoin. Trotzdem sind 6 Minuten kein vernachlässigbare Zeit, weshalb auch Ethereum nicht für dieses System verwendet werden kann.

3.5 Privacy Pass

Ein VPN bietet einige Vorteile in der regulären Kommunikation über das Internet. Bspw. erhöht er die Anonymität des Nutzers, da dessen private IP Adresse so geschützt bleibt. Allerdings gibt es auch Nachteile die häufig übersehen werden. Einer davon ergibt sich daraus, dass wenn der Internetverkehr von hunderten Benutzern des VPNs über die gleiche IP Adresse verteilt wird, dann genügt ein bösartiger Nutzer, der die IP Adresse für einen Angriff oder o.ä. nutzt, um der IP einen schlechten Ruf bei populären Content Delivery Networks wie Cloudflare zu verschaffen. Ein Content Delivery Network (kurz CDN) ist dafür zuständig häufig angefragt Websiten wie Google.com oder Netflix.com in seinem Cache aufzubewahren und so die Zugriffszeit welche durch physikalisch große Distanzen zwischen Nutzer und Server entsteht zu verkürzen [aka]. Zusätzlich liefert ein CDN ein Menge an Sicherheitsfunktionen, wie unter anderem IP Adressen mit schlechtem Ruf ein CAPTCHA präsentieren, um Botzugriffe zu verhindern. Daraus resultiert dass ein regulärer Nutzer eines VPNs erheblich mehr CAPTCHAs lösen muss, als ein Nutzer der keinen VPN verwendet.

3.5.1 Umgehen vom CAPTCHAs

Privacy Pass ist eine Browsererweiterung die es ermöglicht, vorab ein CAPTCHA zu lösen und damit eine Menge an Token zu erhalten. Solange ein Nutzer über mindestens einen Token verfügt, kann er das nächste Mal, wenn ein CDN ihn aufgrund eines schlechten IP Rufwertes zum lösen eines CAPTCHAs auffordert, anstatt einen Token einlösen und kann die Aufgabe so überspringen. Dadurch erhöht sich die Nutzerfreundlichkeit unter der Verwendung eines VPNs da die Anzahl an zu lösenden CAPTCHAs rapide sinkt. [Dav+18]

3.5.2 Funktionsweise

Das System ist in eine sogenannte Signierphase und Einlösephase aufgeteilt. Die Signierphase startet nachdem der Nutzer erfolgreich ein CAPTCHA gelöst hat. Sie ist dafür zuständig dem Nutzer eine Anzahl an Coins auszustellen die durch den Server signiert sind. Die Einlösephase beginnt wenn ein CDN ein CAPTCHA für den Zugang zu einem Webinhalt fordert. Bei ihr wird einer der gespeicherten signierten Token eingetauscht um die Lösung des CAPTCHAs zu überspringen.

Signierphase Erstellen und signieren von Token

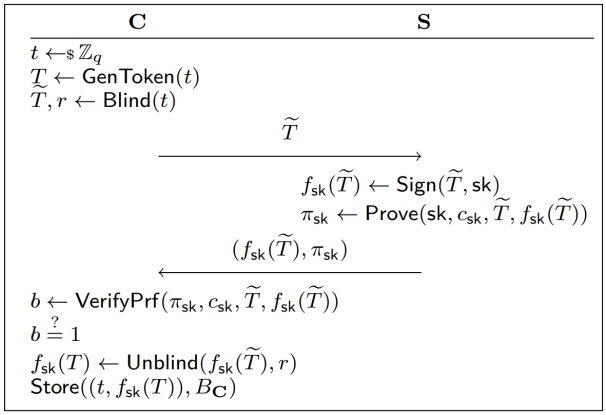

Abbildung 3.2: Signierphase von Privacy Pass [Dav+18]

Zuerst erstellt der Nutzer *C* (Client) einen Tokenseed t mit $t \in_R \mathbb{Z}_q$. Daraus generiert er Token T bspw. mit einer Hashfunktion und blindet diesen mit r wie in 3.7 beschrieben, um $\widetilde{T}$ zu erhalten. Dieser geblindete Token wird nun an der Server *S* gesendet, damit dieser in signieren kann. Beim Server angekommen beginnt dieser damit, den Token mit seinem privaten Schlüsselt sk zu signieren. Anschließend erstellt er einen sogenannten Batch Discrete Log Equivalence Proof (BDLEQ), um dem Nutzer zu beweisen, dass er nicht für jeden Nutzer einen eigenen privaten Schlüssel verwendet um ihn über längere Zeit zu deanonymisieren. Der signierte Token und der BDLEQ werden wieder an den Nutzer gesendet, dieser prüft die Korrektheit des Beweises, unblindet den signierten Token und speichert ihn für spätere Verwendung.

Einlösephase Signierten Token einlösen

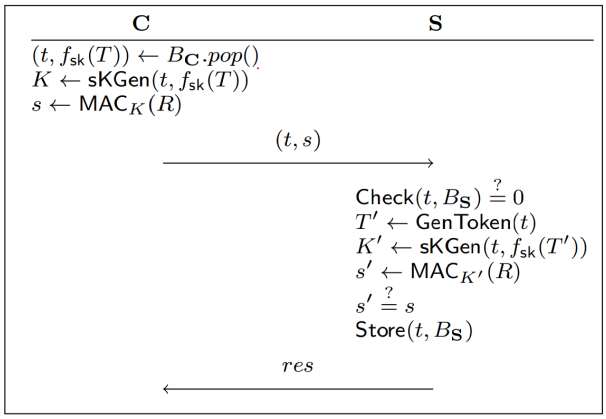

Abbildung 3.3: Einlösephase von Privacy Pass [Dav+18]

Der Nutzer beginnt damit einen gespeicherten signierten Token auszulesen und einen anfragenabhängigen Wert R zu berechnen. Dieser könnte einfach die Domain der Anfrage sein. Er generiert einen shared key K aus dem Tokenseed und dem signierten Token und verschlüsselt R mit K als key. Daraufhin sendet er das Tupel aus Tokenseed t und $s \leftarrow MAC_K(R)$ mit shared key verschlüsseltem R an den Server. Der Server prüft ob t bereits für eine vorherige Anfrage verwendet wurde. Falls dies nicht der Fall ist, berechnet er auf Grundlage von t alle Schritte des Nutzers und erhält so ein s'. Sollte $s' = s$ gelten, dann ist der Token valide, das CAPTCHA kann übersprungen werden und der Nutzer erhält Zugriff auf die angefragt Webressource.

Im Grund lässt Privacy Pass gut auf die Verwendung im Datentreuhändermodell anwenden. Unter den Annahmen, dass ein Datennutzender hier ein Nutzer ist, kann er Token bei dem Datentreuhänder

erwerben, speichern und zu einem späteren Zeitpunkt wieder ausgeben. Da die Token blind signiert werden wird vermieden, dass der Datentreuhänder die in den unterschiedlichen Phasen verwendeten Token zueinander verlinken kann und Zusammenhänge zwischen einlösen und ausgeben ziehen kann. Der Datentreuhänder kann eine weitere überprüfende Instanz verwenden wie in [Dav+18] beschrieben wird, um den Aufwand der Überprüfung auszulagern.
Einige Punkte sprechen jedoch gegen die direkte Verwendung von Privacy Pass.

1. Die Token haben keinen Wert. Bei Bezahlungsmittel ist es essentiell einen bestimmten Preis bezahlen zu können. Dafür wären entweder mehrere unterschiedliche Token von Nöten oder Token welche einen monetäre Wert gespeichet haben. Ansonsten kann nur in Vielfachen von dem einen Token gezahlt werden was dazu führt, dass es keine präzise Preisvergabe gibt oder bei jeder Transaktion eine große Menge an Token validiert werden müssen.

2. Ein Nutzer interagiert sowohl zum Einlösen als auch zum Ausgeben nur mit dem Server und nicht mit einem Verkäufer dem er seine Token im Tausch anbietet. Es existiert hier also kein 3. Akteur wie bei GNU Taler 3.2 der erhaltete Token einlöst.

3. Die Token können ähnlich wie bei GNU Taler 3.2 nicht direkt an den Verkäufer weitergegeben werden, da dadurch eine direkte Verbindung zwischen Datennutzenden und Datengebenden entsteht, welche schützenswerte Informationen an den Datennutzenden preisgibt.

3.6 Privacy-Preserving Reputation Management

In ihrem Paper beschreiben R Petrlic et al. ein Reputationssystem, um Nutzern den Dienst eines Dienstleisters anonym bewerten zu lassen und so anderen Nutzern eine Einschätzung über die Qualität der Dienstleistung zu geben [PLS14]. Es werden 3 Rollen charakterisiert. Ein Reputation Provider RP, der sich um das Verwalten der verschiedenen Reputationswerte kümmert. Eine Menge an Service Providern SP, welche einen Dienst anbieten, der durch Nutzer bewertet werden soll. Und zuletzt eine Menge an Nutzern U, die die Dienstleistung der Service Provider bewerten, wobei keiner der Nutzer gleichzeitig ein Service Provider SP sein kann, $U \cap SP = \varnothing$.

Der im Paper beschriebene Ablauf des Bewertens, eines $sp \in SP$ durch einen Nutzer $u \in U$, umfasst grob die Erstellung eines Schemas für einen Bewertungsvektors durch sp, welcher von u später verwendet wird, um den sp zu bewerten. Nach dem sp dieser Vektor mit dem RP kommuniziert wurde, kann u eine Dienstleistung von sp in Anspruch nehmen. Dieser antwortet, zusätzlich zur Erbringung der Dienstleistung, mit einer Beschreibung des Bewertungsvektors, einem Schlüsselpaar und einem Token. Dank der Beschreibung kann der Nutzer selbst einen Bewertungsvektor erstellen, in welchen er seine Bewertung der Dienstleistung einbaut. Im Anschluss wird dieser mit dem Schlüsselpaar verschlüsselt, signiert und zusammen mit dem signierten Token, an den RP gesendet. Der Token dient hier zur Erkennung von doppelten Bewertungen.

Die Berechnung der Reputation ist in Zeitslots eingeteilt, damit ein sp nicht in der Lage ist einen alten Reputationswert anzugeben. Wenn nun ein Nutzer u den Reputationswert eines sp's einsehen möchte und der Wert für den gewünschten Zeitraum noch nicht bestimmt wurde, so muss dieser zwischen RP und sp berechnet werden. Hierfür bestimmt RP die Summe aller verschlüsselten Bewertungsvektoren für den Zeitraum und sendet diese, zusammen mit weiteren Prüfwerten an sp. Dieser kann die übermittelten Werte prüfen und auf eine Blacklist hinzufügen, um Replay Attacken von RP auszuschließen. Wenn die Überprüfung gelingt, signiert sp die Summe der Vektoren zusammen mit einer ID und verifiziert somit den neuen Reputationswert an RP.

Bei einer Anfrage des Reputationswertes von sp durch u, schickt der sp eine Reihe an Werte zu u. Diese erlauben es u, den Bewertungsvektor zu interpretieren und zu prüfen, dass der übermittelte Werte sowohl aktuell ist, als auch nicht beeinflusst wurde. Das Konzept von Petrlic et al. bietet unter anderem Schutz vor:

1. **Whitewashing**, bei welchem sich ein sp mit schlechter Reputation als neuer sp ausgeben kann, um somit den Wert zurücksetzen kann.

2. **Transaction-independent Ratings**, bei welchen ein Nutzer die Dienstleistung bewerten kann, obwohl er besagt Dienstleistung nicht in Anspruch genommen hat.

3. **Sybil Attacks**. Ein Nutzer bewertet eine Dienstleistung unter mehreren Identitäten und täuscht so seine Meinung als Gruppenmeinung vor.

4. **Delta Analysis**. Eine teilweise Deanonymisierung des Nutzers durch vergleichen von gesammelten Bewertungen in unterschiedlichen Zeitabständen.

In der vorliegenden Situation kann das Konzept zu großen Teilen verwendet werden. Die Rollen können unter anderen Namen genauso verwendet werden. In diesem Fall wird der Service Provider zum Datengebenden, der Nutzer wird zum Datennutzenden und die zu bewertende Dienstleistung wird zu den übermittelten Daten. Allerdings ist das Konzept darauf ausgelegt, dass zu Beginn eine Kommunikation zwischen dem Datengebenden und Datennutzenden besteht, so dass der Reputationswert direkt vom Datennutzenden abgefragt wird. An dieser Stelle muss es also etwas abgewandelt werden, da zu Beginn keine direkte Kommunikation zwischen einem Datennutzender und einem Datengebenden besteht. Erst nachdem der Treuhänder dem Datennutzenden eine List an Datengebenden mitgeteilt hat, kann eine direkte Kommunikation entstehen. Stattdessen soll der Austausch des Reputationswertes direkt über den Treuhänder geschehen.

3.7 Blinde Signaturen

Das von David Chaum im Jahre 1983 veröffentlichtes Paper “Blind signatures for untraceable payments“ beschreibt den theoretischen Ansatz, dass ein Nutzer eine verifizierbare Signatur für eine Nachricht erhält, ohne dass der Unterzeichner den Inhalt der Nachricht kennt. [Cha83]. Ein Beispiel hierfür wäre ein Zahlungsdienstleister, der eine neuen Coin in dem Umlauf bringen möchte. Dafür sendet er seinen Reputationswert und den unkenntlich gemachten Coin an eine Zentrale Institution. Wenn diese den Reputationswert als hoch genug ansieht und somit den Zahlungsdienstleister als vertrauenswürdig erkennt, so kann sie den Coin blind signieren und an den Dienstleister zurückschicken. Wenn nun ein späterer Inhaber des Coins, prüfen möchte, ob sein Coin von einem vertrauenswürdigen Dienstleister erstellt wurde, so kann er die blinde Signatur mit dem öffentlichen Schlüssel der Institution entschlüsseln. Das Ergebnis dieser Berechnung zeigt, dass die Institution den Coin tatsächlich signiert hat und sie dem Coinaussteller vertraut. Der hierbei essentielle Punkt ist, dass diese Institution nicht weiß, was sie unterschreibt. Somit kann die Institution die Beziehung des Dienstleisters und des Coininhabers nicht nachverfolgen.

Die für das Verfahren nötigen Rechenschritte sind im Folgenden beschrieben. Angenommen der Unterzeichnende verfügt über eine private Signierfunktion s' und eine öffentliche Funktion s, sodass $s(s'(x)) = x$. Der Nutzer verfügt über die privaten Funktionen c und dessen invers c', sodass $c'(s'(c(x))) = s'(x)$.

1. Der Nutzer beginnt nun damit, sich ein x auszusuchen. Dieses wird durch zufällige Redundanz vor Kollisionen geschützt und mit $c(x)$ unkenntlich gemacht.

2. Anschließend erhält der Unterzeichnende $c(x)$, berechnet $s'(c(x))$ und schickt den entsprechenden Wert an den Nutzer zurück.

3. Wenn der Nutzer nun $c'(s'(c(x))) = s'(x)$ berechnet, so erhält er den signierten Ursprungswert x ohne, dass der Unterzeichnende x je kannte.

Daraufhin kann jede weitere Person die Unterschrift überprüfen, indem diese die öffentliche Funktion s verwenden um $s(s'(x))$ zu berechnen und das Ergebnis mit x abgleichen.

Blinde Signaturen sind bei dem Ansatz von privatsphäreschützenden Zahlungs- und Reputationssystemen einer der Kernbausteine der verwendet wird, um einen vertrauenswürdigen Austausch zwischen den Parteien zu ermöglichen. Sie liefern die Grundlage der Kommunikation+0

3.8 Partiell Blinde Signaturen

Partiell blinde Signaturen ähneln sich vom Effekt stark zu blinden Signaturen von Chaum aus 3.7. Der jedoch entscheidende Unterschied ist, dass es bei partiell blinden Signaturen einen Infowert gibt der sowohl Nutzer als auch Signierende bekannt ist. Dieser kann genutzt werden, damit der Signierende möglicherweise Prüfung ausführen kann und zu entscheiden ob er die Nachricht wirklich signieren möchte. Diese Eigenschaft wird im späteren Verlauf bspw. dazu verwendet einen Coin partiell blind zu signieren. Hierbei ist es essentiell, dass der Signierende den monetäre Wert des Coins prüfen kann, da ein Nutzer sonst freie Kontrolle über den Wert seines eigenen Coins hat. Mit partiell blinden Signaturen, kann genau dies gewährleistet werden. Dabei ist die monetäre Wert des Coins die Info und der kryptographische Wert des Coins die Nachricht. Nun kann der Signierende prüfen, dass der monetäre Wert der vereinbarte Wert ist und kann den kryptographischen Wert des Coins blind signieren. Der springende Punkte ist der Zusammenhang zwischen Info und Nachricht, da sonst der monetäre Wert des Coins nach dem Signieren durch den Nutzer verändert werden könnte. Um dies zu verhindern, verwendet das Verfahren den Hash der Info als Teil der Signatur, so dass diese nur gültig ist, solange die Info unverändert bleibt.

3.8.1 Funktionweise

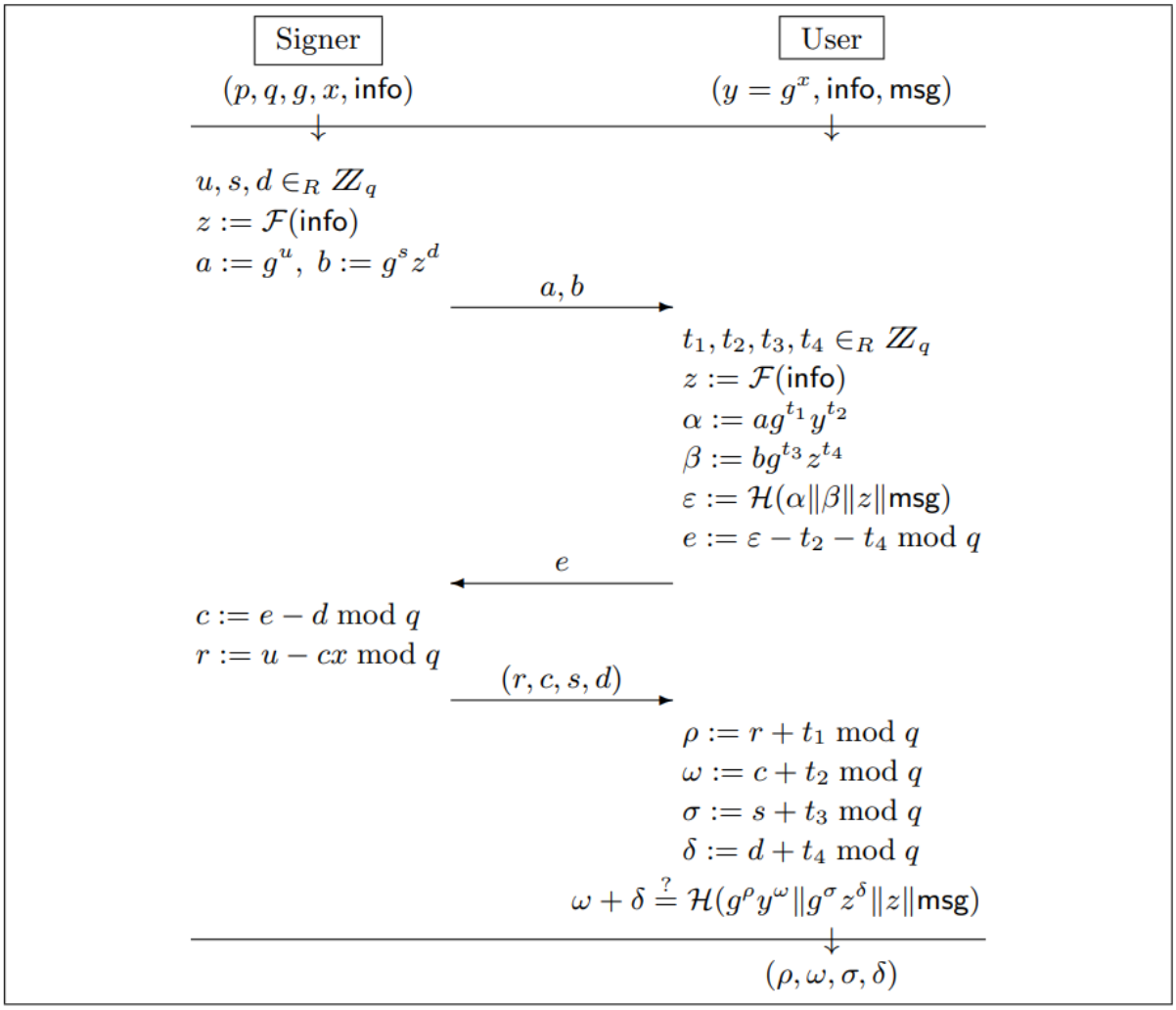

Abbildung 3.4: Ablauf einer partiell blinden Signatur [AO00]

Die Durchführung einer partiell blinden Signatur besteht aus 4 Schritten. Diese 4 Schritte sind notwendig, um die Abhängigkeit von dem msg und info herzustellen. Es wird davon ausgegangen, dass vor dem ersten Schritt beide Parteien über den Infowert verfügen. Der Signierende bildet zuerst den Hashwert der info signiert diesen und schickt ihn an den Nutzer. Dieser bildet den gleichen Hashwert der info und verwendet die erhaltenen Werte zusammen mit einer Reihe an Zufallszahlen, um α und β zu berechnen. ε ist der darauf folgende Hash von α, β, z und msg. Hier fließt die bereits signierte info über α und β mit ein. Kurz darauf empfängt der Signierende e, welches nun sowohl info als auch msg beinhaltet. Dieses e kann er nun verwenden, um info und msg zu signieren und die aus den Berechnungen resultierende Werte an den Nutzer zu senden. Der Nutzer kann dadurch $\rho, \omega, \sigma, \delta$ zu bestimmen und zu speichern. Möchte der Nutzer zu einem beliebigen Zeitpunkt die Signatur auf ihre Gültigkeit prüfen, so kann er $\omega + \delta$ mit $H(g^\rho y^\omega || g^\sigma z^\delta || z || msg)$ vergleichen. Wenn die Werte übereinstimmen, ist die Signatur valide.

Durch den Verwendung von info als Teil der Signatur ist gewährleistet, dass die Signatur nur bei einer unveränderten info und msg valide bleibt. Denn sollte sich die info oder msg ändern, dann würde die Rechnung zum Überprüfen einen anderen Hash erzeugen, welcher nicht mehr mit $\omega + \delta$ übereinstimmt. Insgesamt ist das partiell blinde Signaturen Schema etwas umständlicher als Chaums blinde Signaturen. Dennoch bietet es die Möglichkeit einen öffentlichen Wert zu kommunizieren und dessen nachträgliche Änderung zu verhindern.

3.9 Elliptische Kurven Kryptographie

Die Kryptographie über elliptische Kurven (ECC - elliptic curve cryptography) ist ein Zweig der Kryptographie welcher bereits seit 1985 besteht [Mil85] und trotz dessen nur wenig Aufmerksamkeit genießt . Hierbei geht es um das Ver- und Entschlüsseln von Nachrichten anhand von Punkten auf einem elliptischer Graph, was verglichen mit der weit verbreiteten Faktorisierungskryptographie große Performanzsteigerung liefert.

3.9.1 Trapdoor functions

Das asymmetrische RSA Verschlüsselungsverfahren ist heute weltbekannt und wird in über 90% der Onlinekommunikation verwendet [Kee21]. Dessen Sicherheit basiert genauso wie die von ECC auf sog. *Trapdoor functions*. Ein Trapdoor function ist ein mathematisches Problem, welches in eine Richtung leicht zu berechnen ist, jedoch das Inverse enorm schwer. Im Falle von RSA ist die Trapdoor function das Faktorisierungsproblem. Es ist leicht 2 sehr große Zahlen miteinander zu multiplizieren. Allerdings ist es enorm schwer bei einem gegebenen Produkt dessen Faktoren zu bestimmen, insbesondere wenn die Faktoren jeweils Primzahlen sind. Dies ist der Grund weshalb RSA sicher ist und zuverlässig die Kommunikation schützt. Bei ECC ist die Trapdoor function, die die Sicherheit der Verschlüsselung ausmacht, eine andere. Jedoch kann anhand dieser, ein public und private key generiert werden wie es für asymmetrische Kryptographie nötig ist.

3.9.2 Funktionsweise

Um die Trapdoor function hinter ECC zu verstehen müssen wir uns zuerst elliptische Kurven anschauen. Die algebraische Struktur von Elliptische Kurven ist eine Gruppe, welche aus einer Menge von Punkten M und einer binären Operationen $\circ$ auf 2 Punkten der Menge besteht. Die Definition einer Gruppe fordert, dass die Element die Eigenschaften der Assoziativität, Identität, Existenz eines inversen Elements und je nach Quelle auch Abgeschlossenheit erfüllen. [Ara16][Bog08] Zudem müssen die Koordinaten aller Punkte $(x,y) \in M$ die in der Menge liegen aus dem endlichen Feld stammen, sowie die Gleichung:

$$y^2 = x^3 + ax + b \tag{3.1}$$

erfüllen. Zusammen bildet die Menge an Punkten M einen Graph der je nach Wahl der Parameter a,b einer Form aus Abbildung 3.5 ähnelt. Aufgrund der Gleichung 3.1 entsteht der Zusammenhang, dass ein Gerade durch 2 zufällig gewählte Punkte $P,Q \in M$, den Graph immer an genau einer dritten Stelle schneidet. Zusätzlich ist der Graph an der X-Achse durch das y^2 gespiegelt. Mit diesen Eigenschaften kann nun die Operation $\circ$ definiert werden.
Diese arbeitet wie folgt: Bei Eingabe von $A,B \in M$ finde den invertierten dritten Schnittpunkt mit dem Graph. Dieser sei hier mit C beschrieben. Ein Ausführung von $A \circ B = C$ ist in Abbildung 3.6 verdeutlicht.

Die Operation $\circ$ kann beliebig oft hintereinander mit dem jeweils neu entstehenden Punkt ausgeführt werden. Genau hier liegt die Trapdoor function von ECC. Mit Kenntnissen über der Startpunkt A und die Anzahl der Ausführungen n, ist es einfach den Endpunkt E zu berechnen. Wenn allerdings nur der Endpunkt E und Startpunkt A gegeben sind, ist es sehr rechenaufwändig die Anzahl an Ausführungen zu bestimmen.

3.9.3 Anwendung in der Kryptographie

Allgemein ist zu beachten, dass elliptische Kurven die für ein kryptographischen Verwendung in Frage kommen, eine andere Form haben, als die eben aufgeführt Abbildungen (3.5,3.6). Entscheidend ist hierfür die Wahl des endlichen Feldes. In der Kryptographie wird hier meist $\mathbb{Z}_p$ verwendet. Diese Wahl bringt 2 Eigenschaften mit sich:

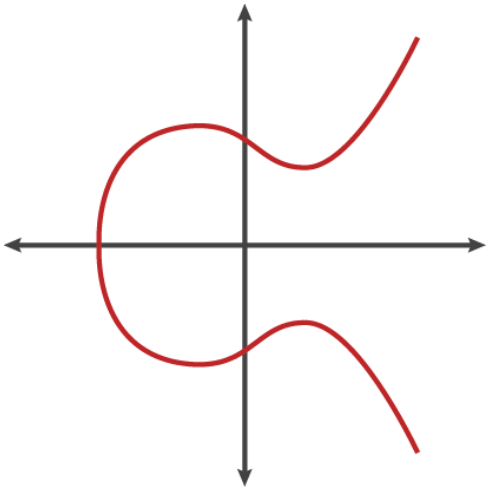

Abbildung 3.5: Mögliche Form einer elliptischen Kurve [Sul13]

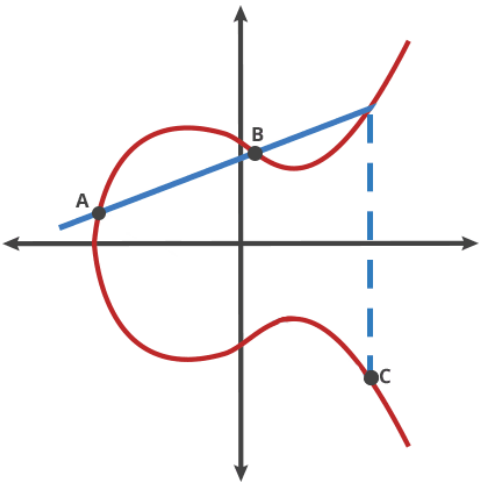

Abbildung 3.6: Operation $\circ$ auf Punkten $A, B \in M$ [Sul13]

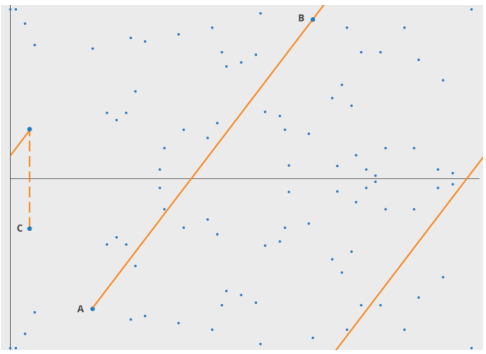

Abbildung 3.7: Graph nach Anwendung von $\mathbb{Z}_p$ [Sul13]

1. Das p in $\mathbb{Z}_p$ besagt, dass alle Werte $x \in \mathbb{Z}_p$, $0 \leq x \leq p-1$ erfüllen müssen. Durch die Wahl dieses Wertebereichs, können die X und Y-Achse nie den Wert p überschreiten sondern beginnt anstatt wieder bei 0.

2. Die zweite Eigenschaft ist die ausschließliche Betrachtung ganzer Zahlen als X und Y-Koordinate. Hierdurch wird aus dem Graphen eine Menge an zufällig aussehend gewählten Punkten. Der Graph für die kryptographische Verwendung einer elliptische Kurve ist ein Abbildung 3.7 visualisiert.

Nun muss ein passendes p sowie a, b bestimmt werden, um die genaue Menge an Punkten zu spezifizieren. Hierfür gibt es bereits ein große Auswahl und Werten in der Literatur [LM10][ML13]. Eine der prominentesten ist die "Curve25519" von Daniel J.Bernstein [Ber06]. Sie hat den Werte $p = 2^{255} - 19$ (daher der Name) und die elliptische Funktion $y^2 = x^3 + 486662x^2 + x$.

3.9.4 Vor und Nachteile

Allgemein betrachtet ist die Tatsache, dass ECC eine schneller Berechnungszeit liefert als RSA, nicht von der Hand zu weisen [Sul13]. Da die Laufzeiten von Verschlüsselung und Entschlüsselung durch RSA und ECC asymptotisch nicht gleich verhalten ist es schwer eine endgültige Antwort zu nennen. Allerdings schlägt ECC RSA bei der Gesamtzeit für Ver- und Entschlüsselung je nach Nachrichtengröße um einen Faktor von ~$3-20$ [MY18][Bao22]. Zudem ist eignet sich ECC vorallem als Ver- und Entschlüsselungsmethode auf rechenschwachen Geräte wie Mobiltelefonen aufgrund von kleineren Schlüsselgrößen[1]. So können Rechenaufwand, Energieverbrauch und RAM-Auslastung gesenkt werden [GS11].

Allerdings wurde 2007 bekannt, dass der Pseudo Random Number Generator (PRNG) Dual_EC_DRBG eine potentielle Backdoor hatte, die es Nutzern mit Informationen über einen Wert die Zufälligkeit problemlos knacken lies [Gre13]. Dual_EC_DRBG wurde 2005-2006 zusammen von NIST (National Institute of Standarts and Technology) und der NSA veröffentlicht und basiert die Auswahl der zurück gegebenen pseudozufälligen Zahlen auf Berechnung über elliptischen Kurven. Dieser Vorfall schwächt bis heute das generelle Vertrauen in ECC.

[Sul13]

3.10 Abschluss

1. Um eine security bit level von 256bits mit RSA zu erreichen, ist eine Schlüssellänge von 15360bits nötig während es bei ECC lediglich 512bits sind.[MY18]

4 Entwurf der Systeme

4.1 Ziel

Um einen privatsphäreschützenden Anreiz für die Benutzung eines Datentreuhänders zu schaffen, wird im folgenden ein Bezahlsystem vorgestellt. Dieses Bezahlsystem erlaubt es Datengebenden für ihre Daten im gegenzug einen finanziellen Ausgleich zu verlangen, ohne dabei persönliche Informationen preisgeben zu müssen. Weder der Datennutzende noch der Datentreuhänder kann bei einer regulären Transaktion bestimmen, welcher Datengebende ein Bezahlung erhält. Der hierfür vorgesehen Ablauf sieht vor, dass ein Datengebender zuerst seine Daten an den Datennutzenden überträgt und ihm etwas Zeit zur Verwertung der Daten lässt. Ein Datennutzender erhält maxmimal 3 Tage Zeit, um den Zahlvorgang abzuschließen. Tut er dies nicht innerhalb der 3 Tagen, so kann der Datentreuhänder ihn von der weiteren Verwendung der Plattform ausschließen.

Damit der Datengebende seine Anonymität nicht ausnutzen kann, gibt es 2 Mechanismen zur Betrugserkennung. Der erste ist eine Offenlegung der Kommunikation im Streitfall. Sollte sich eine der beiden Parteien betrogen fühlen, so kann sie vom Datentreuhänder eine Schlichtung verlangen. Dafür werden die für die Kommunikation verwendeten Einmalschlüssel dem Datentreuhänder mitgeteilt, sodass dieser alle Schritte nachvollziehen kann und entscheiden kann ob ein Betrug vorliegt oder nicht.

Der zweite Mechanismus ist eine Reputationsvergabe. Wenn ein Datengebender besonders qualitativ gute Daten liefert, wird er durch den Datennutzenden mehr bezahlt, was sich in der Reputation widerspiegelt. Wenn er hingegen besonders schlechte Daten liefert, so kann ein Datennutzender sich weigern ihn zu bezahlen, was die Reputation runter zieht. So kann bei jeder Nachfrage nach Daten von einem Datennutzenden ein Reputationsschwellwert festgelegt werden, um Datengebende mit zu schlechter Reputation vor dem einreichen ihrer Daten abzuhalten.

4.2 Geld als Anreiz

Da im Datentreuhandmodell ein Datennutzender meist durch ein Unternehmen oder eine Forschungseinrichtung repräsentiert wird, ist es verstellbar, dass der Datennutzende selbst eine Dienstleistung anbieten oder in Aussicht stellen kann. Hier wurde aber Geld als Anreiz gewählt, da die Verwendung solcher Dienstleistungen als Anreiz einige Nachteile mit sich ziehen kann.

1. **Begehrtheit** Kein Handelsgut oder Dienstleistung ist so universelles begehrt wie Geld. Zwar ist es möglich das ein Business welches hier als Datennutzender auftritt, für die Preisgabe der Daten eine Dienstleistung anbietet. Bspw. können Social Media Plattformen wie X konstenlose zeitlichbedingte Premiumabonnenments oder vergleichbar Angebote in Aussicht stellen. Jedoch sind ist das Interesse an solchen Angeboten subjektiv, was sie als mögliche Anreize zwar denkbar aber eher suboptimal macht.

2. **Hoher Aufwand** Wenn der Anreiz durch den Datennutzende gestellt wird, wird dieser in jedem einzelnen Fall unterschiedlich sein. Aus der Kommunukationsstruktur folgt das alle unterschiedlichen Kompensationen auf irgendeinem Weg über der Datentreuhänder laufen müssen. Daraus ergibt sich einen enormer zusätzlicher Aufwand an Entwicklung und Implementation für jeden Datennutzenden.

3. **Verfügbarkeit** Manche Datennutzende wie bspw. Forschungsinstitutionen haben keine Dienstleistungen oder Produkte die sie Datengebenden anbieten können. Sie sind ausschließlich an der

Forschung interessiert und können den Datengebenden nichts bieten was sie zum preisgeben ihrer Daten veranlassen würde.

4. **Wahrung der Anonymität** Viele C2B Treuhänder legen Wert auf die Anonymität oder Pseudonymität der Datengeber. Diese kann jedoch dadurch, dass eine Datennutzender die versprochene Dienstleistung kontrolliert verloren gehen. Es ist bspw. denkbar dass ein Datennutzender eine vielversprechenden Anreiz verspricht, diesen aber hinter einer Registrierung und Angaben von persönlichen Daten verbirgt, so dass die durch den Treuhänder etablierte anonymität umgangen wird.

5. **Einhalt der Leistung** Der Datentreuhänder kann bei der Auslagerung von Anreizen an den Datennutzenden nicht zuverlässig kontrollieren, dass der Datennutzende die versprochene Dienstleistung wirklich einhält.

Durch eine die Verwendung von Geld als Anreiz werden diese Punkte umgangen. Das Interesse der Datengebenden ist nun ausschließlich abhängig von dem Ruf und dem persönlichen Interesse an einem Datengebenden und nicht von dessen Anreizangebot. Die in Anspruchnahme des versprochenen Anreizes ist für jeden Datennutzenden gleich Implementiert und hängt maximal von der Wahl anderer Parameter ab. Es kann davon ausgegangen werden dass alle Datennutzenden über Geld verfügen und dieses als Kompensation verwenden können. Aufgrund der (trivialität?) des Wertes von Geld sind bereits viele Wegen bekannt wie diesen Wert anonym oder pseudony an andere überträgt. Der Datentreuhänder hat die volle Kontrolle, dass ein zu zahlender Geldbetrag vom Datennutzenden abgegeben wird und auch bei Datengebenden ankommt.

4.3 Generelles

Im gegensatz zu GNU Taler soll das Bezahlsystem hauptsächlich den Datengebenden und dessen Informationen schützen. Die privatsphäre des Datennutzenden ist von geringerer Bedeutung, da im Datentreuhändermodell der Datennutzende meist durch ein Unternehmen oder eine Forschungseinrichtung dargestellt wird. Hinzu kommt dass durch die Struktur eines Datentreuhänders, der Datennutzende dazu gezwungen ist eine Menge an Informationen anzugeben, um vom Datentreuhänder zugelassen zu werden. Der Datengebende hingegen ist eine einzelne Privatperson die ihre gesammelten Daten bewusst an Unternehmen oder Forschungseinrichtungen verkaufen möchte. Der Schutz der Privatperson ist hier also von größerem Interesse.

Unter diesen Umständen ist eine herkömmliche Überweisung per Bank deutlich zu unsicher. Sie benötigt Information wie die IBAN des belasteten Kontos und die IBAN des empfangenden Kontos, um zu wissen von wo nach wo eine Geldsumme fließen soll. Dadurch werden bereits zu viele Informationen über den Datengebenden bekannt, da der Datennutzende nun genau bestimmen kann von wem die erworbenen Daten stammen und ob er bereits vorher schon von demselben Datengebenden Daten gekauft hat. Zusätzlich kann die Bank nachvollziehen, dass eine Privatperson sich als Datengebender angemeldet hat und wem sie ihre Daten bereits verkauft hat. Um dieses Problem zu lösen wird hier eine der gängigsten anonymen Onlinezahlmethoden verwendet: Kryptowährungen. Die Abschnitte 3.2, 3.3, 3.4, 3.5 führen bereits wenige davon auf und beschreiben warum sie nicht im Datentreuhändermodell verwendbar sind. Aufgrund dessen wird hier eine neuer Coin eingeführt. Jedoch ist dieser weder Blockchain basiert, noch für Spekulationszwecke geeignet. Ein Coin kann - so wie bei den meisten Kryptowährungen - ausschließlich bei einem Exchange gegen Geld getauscht werden. Bei dem initialen tauschen mehrerer Coins ist der monetäre Wert in Euro fest im Coin gespeichert und kann nicht verändert werden.

An dieser Stelle ist anzumerken, dass keine Regulierungen oder Identifizierung zum erwerben eines Coins benötigt werden. Jedoch sind Datennutzende die einzigen die Coins sicher ausgegeben können, da das Bezahlsystem nur dazu verwendet werden kann, Coins von Datennutzenden zu Datengebenden zu übertragen. Datengebende können zwar selbst auch Coins erwerben, allerdings ist der Handel

nicht über dass hier eingeführt Bezahlsystem möglich. Daher bleibt nur der reguläre Tausch, welcher außerhalb des Systems stattfindet und keine Betrugserkennung besitzt. Unter der Berücksichtigung dass das einzige was dem Coin einen festen Wert sicher stellt, die Signatur des Exchanges ist, kann leicht ein nicht verifizierter Coin erstellt werden und im unsicheren Tausch angeboten werden. Der andere Tauschpartner hat keine Möglichkeit den Coin vor Abwicklung des Tauschs auf seine Echtheit zu prüfen, was ein riesen Potentiall für Betrug zulässt. Aufgrund dessen ist anzunehmen, dass nur Datennutzende Coin erwerben werden, da der Tausch unter Datengebenden deutlich zu unsicher ist.

Nachdem ein Datennutzender einen Coin erworben hat, kann er diesem lokal speichern bis er ihn ausgeben möchte. Sobald sich ein Datengebender seine Daten mitteilt, hat der Datennutzende 3 Tage Zeit die Daten auszuwerten. Während dieser 3 Tage kann er maximal von 10 Datengebenden Daten empfangen. Dies ist ein Schutzmechanismus der einen bösartigen Datennutzenden daran hindert unbegrenzt Daten zu sammeln ohne für diese zu bezahlen. Es ist denkbar den Wert bei Datennutzenden welche bereits lange ohne Betrug agieren zu erhöhen um eine bessere Parallelisierung der Verarbeitung zuzulassen. Wenn ein Datennutzender nach Ablauf der 3 Tage noch keine Transaktion zu dem Datengebenden abgeschlossen hat und dieser bei dem Datentreuhänder eine Betrugsanklage gegen den Datennutzenden vorlegt, so kann der Datentreuhänder sich dazu entschließen den Datennutzenden entweder zeitlich bedingt oder dauerhauft von der weiteren Verwendung der Plattform auszuschließen. Ein gutwilliger Datennutzender kann nach der Auswertung der Daten eine passende Summe an Coins aus seinem lokal speicher auslesen und an den Datengebenden sendet. Sobald dieser die Coins empfängt, prüft er die Signatur der Coins. Wenn diese mit der des Exchanges übereinstimmt, leitet er sie an den Exchange weiter um sich den monetären Wert auf sein auf bspw. seinem Konto gut schreiben zu lassen. Stimmt die Signatur nicht mit der des Exchanges überein, so kann der Datengebende auch hier wieder einen Betrug beim Datentreuhänder melden.

4.4 Verschlüsselung

Da in der Kommunikation indirekt Geld transferiert wird, ist es von höhster Priorität, die gesendeten Nachrichten vor dem Mithören oder Verändern durch unbefugte dritte zu schützen. Für diesen Schutz wird die in 3.9 erklärte elliptische Kurven Kryptographie verwendet. Mit der Verwendung des public keys des Empfängers wird sichergestellt, dass nur Personen mit Wissen über den zugehörigen private key die Nachricht lesen können. Gleichzeitig wird ein Hash der unverschlüsselten Nachricht gebildet und mit dem private key des Sender signiert. Auf diesem Weg kann der Empfänger die Nachricht entschlüsseln, selbst den Hash bilden und prüfen ob der signierte Hash mit dem selbst gebildeten übereinstimmt. Dadurch kann sowohl die Vertraulichkeit als auch die Integrität der Nachricht sichergestellt werden.

4.5 Coin Generierungsphase

Hier werden die Coins die im späteren Verlauf beider Systeme verwendet werden vom Datennutzenden erstellt und vom Datentreuhänder signiert. Die einzelnen Schritte sind in Abbildung 4.1 verdeutlich und werden im folgenden beschrieben.

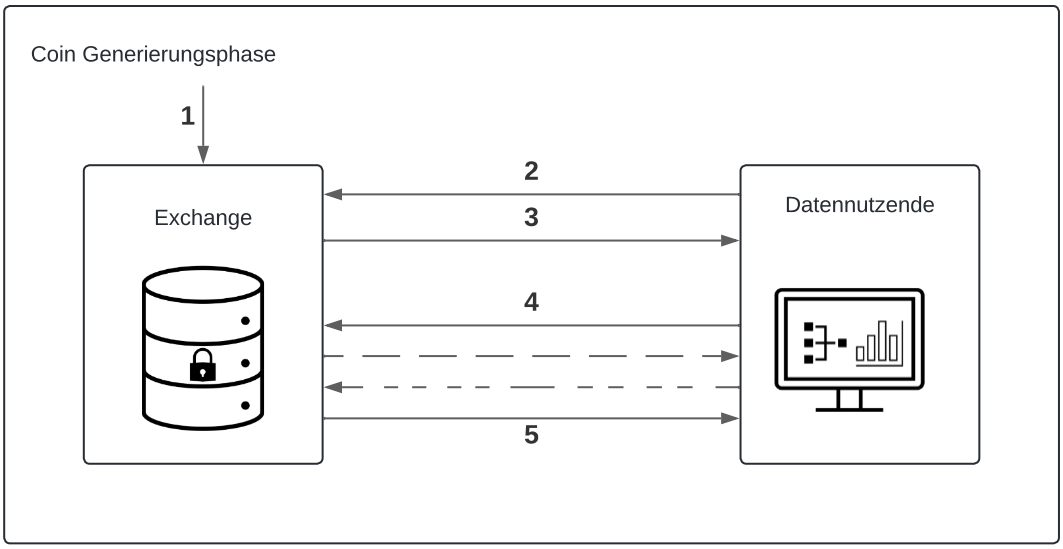

Abbildung 4.1: Coin Generierungsphase Ablauf

1. Zahlungseingang $(apk \leftarrow AccountPublicKey, ES \leftarrow ErhalteneSumme)$
Der Exchange erhält über einen beliebigen Weg eine Summe und einen Konto public key. Die genaue Umsetzung des Zahlungseingang liegt bei dem Exchange. Er kann alles zwischen einer Banküberweisung mit der apk als Verwendungszweck, bis hin zu einem Briefumschlag mit Bargeld und einem leserlich aufgeschriebenen apk sein. Nach Eingang einer Zahlung erstellt der Exchange einen sogenannten CoinGenerationToken bestehend aus $(nonce, apk, ES, spent \leftarrow false)$, verschlüsselt ihn mit dem apk und speichert diesen intern.

2. CoinGenerationToken abrufen (apk)
Da der Exchange nur den apk erhält, hat er keine Möglichkeit den CoinGenerationToken einem Datennutzenden zuzuweisen. Jetzt kann jeder beliebige Nutzer alle Einträge für einen apk abfragen. Allerdings ist nur der Datennutzende in der Lage den CoinGenerationToken zu entschlüsseln, der den passenden secret key zum apk hat. Dies ist im Idealfall nur der Datennutzende der die Transaktion getätigt hat.

3. CoinGenerationToken übertragen $[(nonce, apk, ES)]$
Auf eine CoinGenerationToken Anfrage des Datennutzenden antwortet der Exchange mit allen noch nicht eingelösten Token für den apk. Nachdem der Datennutzende die Token erhalten hat, kann er diesen zuerst mit dem passenden secret key entschlüsseln. Im Anschluss kann er selbst Coins erstellen. Ein Coin besteht aus $(nonce, value)$. Bei der Erstellung sind 2 Sachen zu beachten. Zuerst muss der summierte Wert aller erstellten Coins gleichgroß sein wie die ES des Tokens. Des weiteren gibt es eine Menge an möglichen Werten PV, sodass $\forall value \in PV$ gilt. Dies ist vor allem wichtig, da $value$ beim Signieren und Einlösen des Coins für den Exchange sichtbar ist und dieser bei einer Wahl von selten vorkommenden $value$ ein Verbindung zwischen den Phasen herstellen kann.

4. Signierung Anfragen $(nonce, apk, ES), [(value)]$
Eine Signieranfrage ist der Start eines Durchlaufs von partiell blinden Signaturen, bestehend aus einem CoinGenerationToken und einer Menge an $values$. Zuerst kann der Exchange prüfen ob der gesendete Token noch nicht eingelöst wurde. Anschließend summiert er alle $value$ auf und prüft ob die Summe gleich dem ES des Tokens ist. Wenn beide Überprüfungen akzeptieren, kann der Exchange mit dem partiell blind signieren anfangen und bei seinem gespeichert Token $spent \leftarrow true$ setzen, um eine doppelte Einlösung zu verhindern.

Gestrichelte Pfeile $[(a,b)],[(e)]$
Die Pfeile zwischen 4. und 5. sind hier gestrichelt eingetragen, da sie notwendige Kommunikationsschritte von partiell blinden Signaturen abbilden. Sie sind essentiell für die Funktionsweise und wurden bereits in 3.8 erklärt, weshalb sie hier nur zur Vollständigkeit aufgelistet werden.

5. Signieranfrage Antwort $[(r,c,s,d)]$
Vorrausgesetzt alle Überprüfungen aus Schritt 4 akzeptieren, so erhält der Datennutzende nun eine Menge an partiell blinden signierten Coins und kann diese für spätere Verwendung lokal speichern. Sollten die Überprüfungen nicht akzeptieren, so kann der Datennutzende entweder bei Schritt 4 mit anderen Coins oder bei Schritt 2 wieder ansetzen.

Auf diese Weise kann ein Datennutzender Geld bei einem Exchange einzahlen und eine dem Geldbetrag entsprechende Menge an Coins erhalten, ohne dass der Exchange die von ihm ausgehändigten Coins nachverfolgen kann. Gleichzeitig ist es für einen Datennutzende nicht möglich Coins zu erhalten für die er keinen monetären Gegenwert bereitgestellt hat.

Zusätzlich ist zu erwähnen, dass Exchanges finanziell motiviert sein können oder eine Gebühr verlangen möchten, um bspw. laufende Serverkosten zu decken. Sollte dies der Fall sein, kann vor dem Beginn der Coin Generationphase vom dem Exchange eine feste Gebühr vorgeschrieben werden. Wenn sich ein Datennutzender trotz der Gebühr dazu entscheidet diesen Exchange zu verwenden, so kann der Exchange nach dem Eingang einer Zahlung in Schritt 1, einen CoinGenerationToken erstellen, welcher die Gebühr bereits abzieht. Wenn also ein Datennutzender eine Transaktion von 100€bei Exchange mit 10% Gebühr tätigt, so erstellt der Exchange einen CoinGenerationToken für den angegebenen apk mit $ES \leftarrow 90$€. Dadurch kann eine Exchange seine finanziellen Interessen geltend machen. Auch wenn es nicht das Ziel eines Exchanges sein sollte wirtschaftlich zu handeln, lässt so immerhin verhindern, dass ein Exchange mit den Betriebskosten Minus macht.

4.6 Bezahlvorgang

Nun wo die Coins erstellt und signiert sind, kann damit angefangen werden sich mit dem ausgeben dieser Coins zu beschäftigen. Dafür gibt es das Bezahlsystem welches es Datennutzenden erlaubt die Daten von Datengebenden anzufragen. Diese können der Anfrage nachkomme wofür sie der Datennutzende im Gegenzug mit den Coins entlohnt. Damit bei dem Vorgang die Privatsphäre des Datengebende so gut wie möglich geschützt wird, findet der Austausch der Daten und Coins über Postfächer beim Datentreuhänder statt. Dadurch entsteht keine direkte Kommunikation zwischen Datengebenden und Datennutzenden welche potentiell schützenswerte Informationen wie IP oder dergleichen preisgibt. Wie der Ablauf genau funktionert wird im folgenden erklärt.

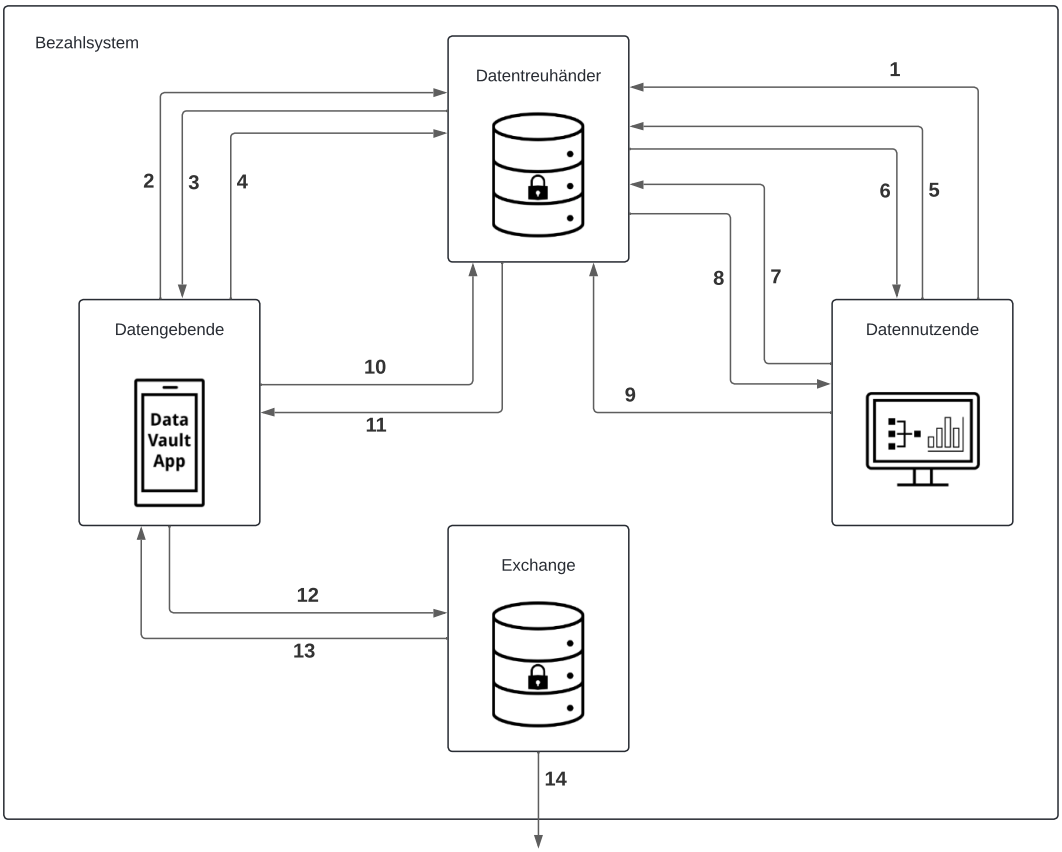

Abbildung 4.2: Bezahlsystem Ablauf

1. Aufruf starten $(datatype, reputationThreshold, price)$
Der Datennutzende erstellt beim Datentreuhänder einen Aufruf zur Datenteilung. Damit verkündet er an alle Datengebenden, dass er Daten von einem bestimmten Typ sucht und nennt direkt wieviel er für diese Daten bereits ist zu bezahlen. Da der Datennutzende nach Anlegung des Aufrufs direkt Daten erhalten kann für die er Zahlen muss, braucht es eine Möglichkeit ein gewisses Vertrauensminimum festzulegen. Der $reputationThreshold$ gibt dabei einen Schwellwert an, so dass der Aufruf nur für Datengebende mit einer Reputation über diesem Schwellwert sichtbar ist.

Im Datentreuhändermodell ist noch ein weiterer Kommunikationstart denkbar. Es besteht die Option, dass ein Datengebender seine Daten beim Datentreuhänder anbietet und sich Datennutzende aus einer bereits bestehenden Auswahl von Angeboten welche aussuchen. Jedoch skaliert dieser Ansatz schlechter da der Datennutzende in der Regel an großen Mengen von Daten interessiert sind, was zu mehr Rechenaufwand auf Seiten des Datennutzenden führt. Deswegen wird von dem beschrieben Aufrufansatz ausgegangen.

2. Aufrufliste anfragen $(DG\text{-}ID)$

Erst nach reputationssytem

3. Aufrufliste Antwort $([callDetails, DN\text{-}publicKey])$
Nachdem der Datengebenden seine Reputation bewiesen hat, kann der Datentreuhänder seine lokal Datenbank mit Aufrufen durchsuchen und alle Aufrufe welche einen Reputationsschwellwert niedriger als die bewiesen Reputation haben in eine Liste aufnehmen. Damit besteht jeder Eintrag der List dem publicKey des Datennutzenden der den Aufruf gestartet hat und Aufrufdetails die wichtige Informationen bezüglich des Aufrufs geben. Die genauen Inhalte der Aufrufdetails sind Treuhänderabhängig, da je nach Gebiet des Treuhänders verschieden Details über den Datennutzende für die Entscheidung des Datengebenden wichtig sein können. In den meisten Fällen umfassen die Aufrufdetails Daten wie die

angefragte Datentypkategorie (GPS-Daten, persönliche Daten, etc.), den Namen des Datennutzenden, den Preis den jeder Datengebende ausgezahlt bekommen soll, einen Titel oder eine Beschreibung. Auch Informationen wie ob es sich bei dem Datennutzenden um eine Forschungseinrichtung handelt werden können hier kommuniziert werden.

4. Daten teilen $(DN\text{-}pk, \{sharedKey\}_{DN-pk}, \{referenceCode, dataLocation, dataKey\}_{sharedKey})$
Der Datengebende hat die genannten Aufrufdetails erhalten und sich dazu entschieden, seine Daten für einen Aufruf mit einem Datennutzenden zu teilen. Generell gilt wie in 2.2 erklärt, dass ein Datengebender seine Daten nicht selbst speichert sondern sie verschlüsselt beim Datentreuhänder lagert. Er behält nur den Speichort und Schlüssel der Daten um jederzeit Zugriff auf diese zu haben. Um seine Daten mit einem Datennutzenden zu Teilen, genügt es daher den Speicherort und Schlüssel zu übertragen.

Für die Übertragung des Speicherorts und Schlüssels wird ein Postfach beim Datentreuhänder verwendet. So kann ein direkter Austausch zwischen Datengebenden und Datennutzenden verhindert werden. Ein Datengebender kann Ort und Schlüssel verschlüsselt in das Postfach legen und nur der Datennutzende der das Wissen über den passenden privaten Schlüssel besitzt kann Ort und Schlüssel wieder entschlüsseln.

Einer der Betrugserkennungsmechanismen ist die Offenlegung der Kommunikation im Streitfall, so dass der Datentreuhänder die Nachrichten einsehen kann und entscheiden kann welche der Parteien recht hat. Damit bei dieser Offenlegung nur genau die Nachrichten entschlüsselt werden, die Teil einer spezifischen Kommunikation zwischen Datengebenden und Datennutzenden sind, wird für jede Abwicklung eines Tausches ein neuer symmetrischer Schlüssel erstellt.

Damit ein Datengebender einen Eintrag in das Postfach zu Teilen von Daten senden kann, muss er zuerst die folgenen Informationen bestimmen. Zuerst wird der publicKey des Datennutzenden im Klartext angegeben, da ein Datennutzender beim Abfragen des Postfaches so bestimmen kann, welche Einträge für ihn bestimmt sind. Anschließend wird ein neuer symmetrischer Schlüssel $sharedKey$ erstellt. Dieser wird mit dem publicKey des Datennutzenden verschlüsselt, damit mit dieser den $sharedKey$ kennt. Nun kann der Datengebende den $sharedKey$ verwenden, um das Tupel aus $dataLocation$, $dataKey$ und $referenceCode$ zu verschlüsseln. Hierbei sind $dataLocation$ und $dataKey$ die eben beschriebenen Speicherort und Schlüssel der Daten beim Datentreuhänder. Dabei sei angemerkt, dass sowohl $dataLocation$ als auch $dataKey$ mehr als nur eine Variable sein können und von dem vom Datentreuhänder verwendeten Speicherverfahren und Verschlüsselungsverfahren abhängen. Der $referenceCode$ ist eine zufällige Zahl, die später bei Schritt 9 verwendet wird, um eine Zugehörigkeit von Übertragen der Daten und Coins herzustellen.

5. Datenpostfach abfragen $(DN\text{-}pk)$
Der Datennutzende kann sich zu einem beliebigen Zeitpunkt dazu entscheiden, alle Nachrichten die den angegeben publicKey enthalten anzufragen. Dafür muss er den gewünschten publicKey mitübertragen, um den Datentreuhänder zu signalisieren, an welchen Nachrichten er interesse hat. Der publicKey muss hier bei jeder Anfrage spezifiziert werden, da zum einen keine Form an Authentifizierung stattfindet und jeder alle Nachrichten eines publicKeys anfragen kann. Zum anderen bleibt dem Datennutzenden so die Möglichkeit offen, für jeden Aufruf einen eigenes Schlüsselpaar zu verwenden. Dies kann bspw. große Vorteile bei der internen Organisierung haben.

6. Datenpostfach Antwort $([\{sharedKey\}_{DN-pk}, \{referenceCode, dataLocation, dataKey\}_{sharedKey}])$
Nachdem der Datentreuhänder die Anfrage an das Datenpostfach erhalten hat, erstellt er eine Liste mit Postfacheinträgen, die den publicKey der Anfrage im Klartext angegeben haben. Die einzelnen übertragenen Postfacheinträge können dabei auf den publicKey verzichten, da dieser nur zu Identifizierung genutzt wird. Der Datennutzende kennt den publicKey ohnehin da er ihn in der Anfrage definiert. Die vollständige alle Postfacheinträge kann zurück an den Datennutzenden gesendet werden.

7. Daten anfragen $(dataLocation)$
Der Datennutzende kann nun für jeden empfangen Postfacheintrag, den passenden secretKey verwenden, um den sharedKey des Eintrags zu entschlüsseln. Solange der Datengebende sich an das Protokoll gehalten hat und den Schlüssel den er zum verschlüsseln des Datentupels verwendet hat auch als Klartextschlüssel angegeben hat, funktioniert dieser Schritt einwandfrei und der Datennutzende erhält Zugang zu $dataLocation$, $dataKey$ und $referenceCode$. Jetzt kann er den Datentreuhänder nach den Daten an der angegebenen $dataLocation$ fragen, welche mit dem $dataKey$ verschlüsselt sind.

8. Daten Antwort $(\{data\}_{dataKey})$
Die durch die $dataLocation$ beschriebenen Daten werden durch den Datentreuhänder ausgelesen. Da auch hier wieder keine Authentifizierung stattfindet, kann jeder einen beliebigen Datensatz beim Datentreuhänder anfragen erhält die an der Stelle gespeichert Daten. Da allerdings nur der Datennutzende, welcher den $dataKey$ vorher erhalten die Daten lesen kann, ist das wahllose anfragen von Daten beim Treuhänder sinnlos, da ohne Wissen über die $dataLocation$ nur Teile eines Datensatzes oder Überlappungen in andere Datensätze geschehen können, welche verschiedene $dataKeys$ benötigen würden. Der Datentreuhänder kann daher mit den ausgelesenen Daten antworten ohne Schlüsse ziehen zu können, wer die Daten angefragt hat oder ob die Nachfrage Teil eines Tausches zwischen Datengebenden und Datennutzenden ist.

Nach erhalt der verschlüsselten Daten kann der Datennutzende den zu den Daten gehörenden $dataKey$ verwenden, um die Daten zu entschlüsseln und an den Klartext zu gelangen. Jetzt kann der Datennutzende die Daten für seine Zwecke verwerten.

9. Coins in Postfach legen $(referenceCode, \{[nonce, value]\}_{sharedKey})$
Mit der Verwertung abgeschlossen kann der Datennutzende nun den Datengebenden bezahlen. Dafür läd er eine Liste an Coins aus seinem Speicher die in der Summe gleich dem im Aufruf ausgeschrieben Preis sind. Er verwendet den in Schritt 6 erhaltenen $sharedKey$, um die Liste an Coins zu verschlüsseln. Anschließend verwendet er den ebenfalls in Schritt 6 erhaltenen $referenceCode$, um zusammen einen Eintrag an das Coinpostfach zu senden.

Da der $referenceCode$ von dem Datengebenden mit dem $sharedKey$ verschlüsselt wurde, ist davon auszugehen, dass ein Eintrag mit dem gleichen $referenceCode$ nur von dem Datennutzenden stammen kann, der den Aufruf gestartet hat und die Daten erhalten und verwertet hat.

10. Coinpostfach abfragen $(referenceCode)$
Da ein Datengebender nach dem Erstellen eines Datenpostfacheintrags den dort genutzten $referenceCode$ lokal als offen abspeichert, kann er nun bei Bedarf den Datentreuhänder fragen, ob es bereits einen Eintrag in dem Coinpostfach mit diesem $referenceCode$ gibt. Er kann somit den Datentreuhänder fragen ob er für das Teilen seiner Daten bereits bezahlt wurde.

11. Coinpostfach Antwort $(\{[nonce, value]\}_{sharedKey})$
Wenn der Datennutzende bereits einen Eintrag mit den verschlüsselten Coins eingesendet hat, so werden die unter dem $referenceCode$ angegebenen verschlüsselten Coins an den Datengebenden zurück gegeben. Der Datengebende kann daraufhin den $sharedKey$, welcher in Kombination mit dem $referenceCode$ gespeichert wurde, verwenden um die erhaltenen Coins wieder zu entschlüsseln. Wenn er anschließend die Liste an Coins erhält, kann er überprüfen, dass die Summe der Coins auch den Preis des Aufrufs ergibt und dass die partiell blinde Signatur, welche in der Coin Generierungsphase Schritt 4. und 5. ausgestellt wurde, auch tatsächlich valide ist. Sollten diese Tests beide akzeptieren, so kann der Datengebende die Coins lokal speichern bis er sie einlösen möchte.

12. Coins einlösen $([nonce, value], address)$ Ein Datengebender kann seine angesammelten Coins jederzeit einem Exchange senden. Dafür kann er die in Schritt 11 gespeicherten Coins aus dem

Speicher lesen und gesammelt oder einzeln zusammen mit einer Form an Zahlungsadresse an einen Exchange übertragen. Dabei ist die Form der $address$ vom Exchange abhängig. Bei der Implementation wird das Übertragen einer IBAN verwendet, allerdings ist es genauso gut möglich stattdessen eine Bitcoin Walletadresse oder eine beliebige andere Zahlungsmöglichkeit zu verwenden. Ein Exchange kann auch mehrere Zahlungsoptionen gleichzeitig anbieten in welchem Fall dann die Zahlungsoption als zusätzlicher Parameter mit übertragen werden muss.

13. Bestätigung $([nonce, status \in \{true, false\}])$ Nachdem die Coins beim Exchange eingetroffen sind, überprüft dieser ob die Coins bereits vorher schon eingelöst wurden und ob die partiell blinde Signatur gültig ist. Anschließend meldet er dem Datengebenden den Status der Coins zurück. Dafür erstellt er eine Liste, die für jeden Coin den $nonce$ zur Zuteilung, sowie einen boolean Wert der angibt ob der jeweilige Coin akzeptiert wurde, und sendet diese zurück.

14. Geld austeilen $(address)$ Hier wird abschließend eine ausgehende Zahlung vom Exchange zu der angegebenen Zahlungsadresse getätigt. Die genau kommunizierten Daten sind hier von der Zahlungsoption abhängig. Allerdings muss hier sichergestellt werden, dass der ausgehende Betrag genau dem Wert der eingelösten Coins gleicht.

4.7 Reputationsvergabe

5 Implementation

5.1 TRESOR Projekt

6 Auswertung

6.1 Bewertung der Sicherheit

6.1.1 Definition der Angreifermodelle

6.2 Analyse des Rechenaufwands

6.2.1 Metrics

6.3 Setup / Implementation

7 Bewertung

7.1 Erfüllen der Anforderungen

7.2 Wie kann ein privatsphäreschützender Anreiz zur Benutzung eines Datentreuhändermodells geschaffen werden?

7.3 Wie kann dieser Anreiz gegen Missbrauch geschützt werden?

7.4 (Discussion)

8 Begrenzungen

9 Zusammenfassung

Eidesstattliche Erklärung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit im Bachelorstudiengang Software-System-Entwicklung selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel — insbesondere keine im Quellenverzeichnis nicht benannten Internet-Quellen — benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht. Ich versichere weiterhin, dass ich die Arbeit vorher nicht in einem anderen Prüfungsverfahren eingereicht habe.

Hamburg, den 23. September 2024

Knut Hoffmeister

Literatur

[] *What is a cryptocurrency transaction blockchain confirmation?* URL: https://www.bitcoin.com/get-started/what-is-a-confirmation (besucht am 12. 09. 2024).

[aka] akamai. *Was ist ein CDN (Content Delivery Network)?* URL: https://www.akamai.com/de/glossary/what-is-a-cdn#:~:text=Bei%20einem%20CDN%20handelt%20es,die%20auf%20die%20Inhalte%20zugreifen. (besucht am 21. 08. 2024).

[AO00] Masayuki Abe und Tatsuaki Okamoto. *Provably secure partially blind signatures*. In: *Advances in Cryptology—CRYPTO 2000: 20th Annual International Cryptology Conference Santa Barbara, California, USA, August 20–24, 2000 Proceedings 20*. Springer. 2000, S. 271–286.

[Ara16] Bálint Aradi. *Einführung in die Gruppentheorie*. 2016.

[Bao22] Jiaxu Bao. *Research on the security of elliptic curve cryptography*. In: *2022 7th International Conference on Social Sciences and Economic Development (ICSSED 2022)*. Atlantis Press. 2022, S. 984–988.

[Ber06] Daniel J Bernstein. *Curve25519: new Diffie-Hellman speed records*. In: *Public Key Cryptography-PKC 2006: 9th International Conference on Theory and Practice in Public-Key Cryptography, New York, NY, USA, April 24-26, 2006. Proceedings 9*. Springer. 2006, S. 207–228.

[Bla+20] Aline Blankertz u. a. *Datentreuhandmodelle-Themenpapier*. In: (2020).

[Bog08] Oleg Vladimirovič Bogopolskij. *Introduction to group theory*. Bd. 6. European Mathematical Society, 2008.

[BS21a] Aline Blankertz und Louisa Specht. *Wie eine Regulierung für Datentreuhänder aussehen sollte*. In: *Policy-Brief. Juli. Berlin: Stiftung Neue Verantwortung eV https://www. stiftung-nv. de/sites/default/files/regulierung_fuer_datentreuhaender. pdf (Zugriff: 14.02. 2022)* (2021).

[BS21b] ALINE BLANKERTZ und DR LOUISA SPECHT-RIEMENSCHNEIDER. *Neue Modelle ermöglichen*. In: (2021).

[Bun] Bunderdruckerei. *Datenblatt-Datentreuhaender*. URL: https://www.bundesdruckerei-gmbh.de/files/dokumente/pdf/datenblatt-datentreuhaender.pdf (besucht am 03. 05. 2024).

[Bun21] Die Bundesregierung. *Datenstrategie der Bundesregierung*. In: *Die Bundesregierung, Berlin* (2021).

[Bur+16] Jeffrey Burdges u. a. *Enabling secure web payments with GNU Taler*. In: *Security, Privacy, and Applied Cryptography Engineering: 6th International Conference, SPACE 2016, Hyderabad, India, December 14-18, 2016, Proceedings 6*. Springer. 2016, S. 251–270.

[But+13] Vitalik Buterin u. a. *Ethereum white paper*. In: *GitHub repository* 1 (2013), S. 22–23.

[But+20] Vitalik Buterin u. a. *Combining GHOST and casper*. In: *arXiv preprint arXiv:2003.03052* (2020).

[Cha83] D Chaum. *Blind signatures for untraceable payments*. 1983.

[Dav+18] Alex Davidson u. a. *Privacy pass: Bypassing internet challenges anonymously*. In: *Proceedings on Privacy Enhancing Technologies* (2018).

[Dig23] Digiconomist. *Bitcoin average energy consumption per transaction compared to that of VISA as of May 1, 2023*. Graph. 2023. URL: https://www.statista.com/statistics/881541/bitcoin-energy-consumption-transaction-comparison-visa/ (besucht am 12. 09. 2024).

[Fac24] Facebook. *Umsatz von Facebook weltweit nach Segmenten in den Jahren von 2009 bis 2023*. 2024. URL: https://de.statista.com/statistik/daten/studie/151159/umfrage/umsatz-von-facebook-in-2009-nach-segmenten/ (besucht am 27. 03. 2024).

[FK20] Oliver Falck und Johannes Koenen. *Rohstoff"Daten: Volkswirtschaftlicher Nutzen von Datenbereitstellung-eine Bestandsaufnahme*. 113. ifo Forschungsberichte, 2020.

[Goo24] Alphabet; Google. *Alphabet, Werbeumsätze von Google in den Jahren 2001 bis 2023*. 2024. URL: https://de.statista.com/statistik/daten/studie/75188/umfrage/werbeumsatz-von-google-seit-2001/ (besucht am 27. 03. 2024).

[Gre13] Matthew Green. *The Many Flaws of Dual_EC_DRBG*. 2013. URL: https://blog.cryptographyengineering.com/2013/09/18/the-many-flaws-of-dualecdrbg/ (besucht am 10. 07. 2024).

[GS11] Kamlesh Gupta und Sanjay Silakari. *Ecc over rsa for asymmetric encryption: A review*. In: *International Journal of Computer Science Issues (IJCSI)* 8.3 (2011), S. 370.

[Ham22] Universität Hamburg. *Tresor Projekt - About*. 2022. URL: https://tresor-projekt.de/about/ (besucht am 28. 02. 2024).

[han17] handysektor.de. *Geheimnis Staumelder: Wie funktioniert eigentlich Google Maps?* 2017. URL: https://www.handysektor.de/artikel/geheimnis-staumelder-wie-funktioniert-eigentlich-google-maps (besucht am 07. 08. 2024).

[Har18] Jack Hardinges. *What is a data trust?* 2018. URL: https://theodi.org/insights/explainers/what-is-a-data-trust/?saved#cookies-form (besucht am 15. 04. 2024).

[JK16] Tun-Min Catherine Jai und Nancy J King. *Privacy versus reward: Do loyalty programs increase consumers' willingness to share personal information with third-party advertisers and data brokers?* In: *Journal of Retailing and Consumer Services* 28 (2016), S. 296–303.

[KB21] Jürgen Kühling und Benedikt Buchner. *Datentreuhänder*. 2021.

[Kee21] Lila Kee. *RSA is dead - We Just Haven't Accepted It Yet*. 2021. URL: https://www.forbes.com/sites/forbestechcouncil/2021/05/06/rsa-is-dead---we-just-haventaccepted-ityet/ (besucht am 24. 06. 2024).

[LM10] Manfred Lochter und Johannes Merkle. *Elliptic curve cryptography (ECC) brainpool standard curves and curve generation*. Techn. Ber. Independent Submission, 2010.

[Mil85] Victor S Miller. *Use of elliptic curves in cryptography*. In: *Conference on the theory and application of cryptographic techniques*. Springer. 1985, S. 417–426.

[ML13] Johannes Merkle und Manfred Lochter. *Elliptic curve cryptography (ECC) brainpool curves for transport layer security (TLS)*. Techn. Ber. Internet Engineering Task Force (IETF), 2013.

[MY18] Dindayal Mahto und Dilip Kumar Yadav. *Performance analysis of RSA and elliptic curve cryptography.* In: *Int. J. Netw. Secur.* 20.4 (2018), S. 625–635.

[Nak08] Satoshi Nakamoto. *Bitcoin: A peer-to-peer electronic cash system*. In: *Decentralized business review* (2008).

[PLS14] Ronald Petrlic, Sascha Lutters und Christoph Sorge. *Privacy-preserving reputation management*. In: *Proceedings of the 29th Annual ACM Symposium on Applied Computing*. 2014, S. 1712–1718.

[Pro13] Malte Stöfen Prof. Dr. Volker Beeck Prof. Dr. Thilo Seyfriedt. *Treuhandschaft*. 2013. URL: https://wirtschaftslexikon.gabler.de/definition/treuhandschaft-47435/version-270699 (besucht am 22. 04. 2024).

[Ric] Frederik Richter. *DDV-Webtalk Datentreuhänder - Modelle*. URL: https://youtu.be/10UbvTOrjiM?si=bg-4xdCNRH9b93-D (besucht am 03. 05. 2024).

[Sch24] Peter Schöllauf. *What is Bitcoin? Complete Beginner's Guide [2024]*. 2024. URL: https://www.blockpit.io/blog/what-is-bitcoin (besucht am 12. 09. 2024).

[Sul13] Nick Sullivan. *A (Relatively Easy To Understand) Primer on Elliptic Curve Cryptography*. 2013. URL: https://blog.cloudflare.com/a-relatively-easy-to-understand-primer-on-elliptic-curve-cryptography (besucht am 17. 06. 2024).

[TEA24] THE INVESTOPEDIA TEAM. *What Is Ethereum and How Does It Work*. 2024. URL: https://www.investopedia.com/terms/e/ethereum.asp (besucht am 12. 09. 2024).

[TRE24] TRESOR. *Ergebnisse der Nutzerstudie zur Anforderungsdefinition*. 2024. URL: https://tresor-projekt.de/news/user-study-results/ (besucht am 30. 08. 2024).

[YCH] YCHARTS. *Ethereum Average Block Time (I:EBT)*. URL: https://ycharts.com/indicators/ethereum_average_block_time (besucht am 12. 09. 2024).

[ZDF23] ARD; ZDF. *Anzahl der Internetnutzer in Deutschland in den Jahren 1997 bis 2023*. 2023. URL: https://de.statista.com/statistik/daten/studie/36146/umfrage/anzahl-der-internetnutzer-in-deutschland-seit-1997/ (besucht am 07. 08. 2024).