Untersuchung eines vollständig dezentralen, nicht-diskriminierenden und Privatsphäre-schützenden Handelsnetzwerk für digitale Werte ("Krypto-Anlagen")

Forschungsbericht[1] von

Stefan Marić
Bachweg 2
6207 Nottwil
stefan.maric@gmx.ch

[1] Dieser Forschungsbericht wurde in Zusammenarbeit mit der Hochschule Luzern – Informatik – verfasst.

Abstract

Dieser Forschungsbericht wird für das Unternehmen «Kopanyo AG» verfasst. Das Unternehmen entwickelt einen quelloffenen Software-Prototypen. Dieser Software-Prototyp soll gemäss Spezifikation ein vollständig dezentralisiertes, nicht-diskriminierendes und Privatsphäre-schützendes Handelsnetzwerk (diva.exchange) für digitale Werte abbilden. Das Ziel dieses Forschungsberichtes beinhaltet eine Marktanalyse sowie den Versuch die folgenden Hypothesen, welche durch den Software-Prototypen abgebildet werden, zu falsifizieren:

[*H1*] mit einer steigenden Zahl von Netzwerkknoten reduziert sich die Ausfallwahrscheinlichkeit des gesamten Netzwerkes.
[*H2*] eine steigende Zahl von Netzwerkknoten führt zu einer steigenden Transaktions-Kapazität des gesamten Netzwerkes.
[*H3*] eine steigende Zahl von Netzwerkknoten erhöht die Resistenz gegen Brüche der Privatsphäre der Teilnehmer im Netzwerk.
[*H4*] eine steigende Blockhöhe beeinflusst das System und dessen Performanz nicht.

Um dieses Ziel zu erreichen werden deduktive sowie empirische Methoden in diesem Forschungsbericht angewendet. Das deduktive Vorgehen beinhaltet die Literaturrecherche und deren Ergebnisse. Daraus werden Herleitungen aufgestellt, welche für die Falsifikation der Hypothesen *[1]*, *[2]* und *[3]* verwendet werden. Das empirische Vorgehen wird für die Hypothese *[4]* durchgeführt und steht in einer Wechselwirkung mit der Weiterentwicklung des Software-Prototypen.

Die Hypothesen *[1]*, *[2]* und *[3]* konnten nicht falsifiziert werden. Diese Erkenntnis ist für das Projekt positiv. Die Auswertung und die Interpretation der Testdaten lassen die Annahme zu, dass das System ab einer gewissen Blockhöhe degradiert. Die Hypothese *[4]* wurde somit falsifiziert. Diese Erkenntnis sollte den Entwicklern der Blockchain mitgeteilt werden. Die Iroha-Blockchain ist ein neues Projekt, welches am 06.05.2019 publiziert wurde. Deswegen sind Fehler in dieser Phase vermutlich als nicht gravierend einzustufen. Für die Marktanalyse wurden zwei ähnliche Projekte untersucht, welche mit dem Auftraggeber abgestimmt wurden. Die Analyse hat gezeigt, dass diese Projekte Ähnlichkeiten aufweisen, sich jedoch in den wichtigsten Punkten (Anonymität, handelbare Kryptowährungen und eigener Kryptowährung) von diesem Projekt – diva.exchange – unterscheiden.

Für das weitere Projektvorgehen und den Ausblick gilt es die Resistenz gegen Replay-Attacken zu beweisen. Zu Beginn des Projektes wurde entschieden, dass die Blockchain im öffentlichen Netzwerk (Clearnet) integriert wird, weil im Darknet die Performanz klar geschwächt wurde. Die Anonymisierungsschicht garantiert den Benutzern, dass Transaktionen nicht auf ihre Identität schliessen lassen. Das Schreiben auf die Blockchain wird von einer Stellvertreterfunktion durchgeführt, welche das Schreiben im Auftrag von einem Sender übernimmt. Durch diese Implementation entsteht die Gefahr, dass die Nachricht von einem Mittelsmann abgefangen und kopiert wird. Er könnte die Nachricht nach einer gewissen Zeit als Kopie abschicken und der Stellvertreter würde sie als korrekt erachten und somit erneut auf die Blockchain schreiben. Als Lösung wurden zwei Mechanismen implementiert. Die Nachricht wird durch einen Zeitstempel erweitert. Dadurch ist die Gültigkeit der Nachricht von der definierten Zeitspanne des Zeitstempels abhängig. Der zweite Mechanismus (Zustandsmechanismus/Deltamechanismus) erweitert die Nachricht um den Ist-Zustand der Datenbasis vom Sender, welcher dem Stellvertreter ebenfalls bekannt ist, weil er auf der öffentlichen Blockchain vorhanden ist. Durch das Signieren, das Hinzufügen eines Zeitstempels und der Datenbasis vom Sender sind Replay-Attacken vermutlich nicht möglich. Das muss jedoch überprüft und bewiesen werden.

Inhaltsverzeichnis

1 Einleitung und Ausgangslage

Im Mai 2019 sorgte ein Diebstahl in der Kryptoszene für Schlagzeilen, wobei es Hackern gelang, über 7'000 Bitcoins zu entwenden. Der Schaden dieses Vorfalls wurde zu diesem Zeitpunkt auf einen Wert von 42 Millionen Schweizer Franken geschätzt. Dieser Diebstahl war ein Hackerangriff auf eine der grössten Krypto-Börsen – Binance – der Welt [1].

Es ist nicht der erste Vorfall, wo es Angreifern gelang, digitale Werte (bspw. Bitcoin oder Ethereum) von einer Krypto-Börse zu stehlen [2]. Es gibt mittlerweile schon einige Kypto-Börsen, welche sich in der Architektur unterscheiden. Dabei unterscheidet man zwischen zentralisierten (bspw. Binance) und dezentralisierten Börsen (bspw. Bisq), welche das Handeln von Kryptowährungen auf ihren Plattformen anbieten. Zentralisierte Börsen sind so ausgerichtet, dass alles auf und von einer zentralen Einheit verwaltet wird. Die Verwaltung des digitalen Vermögens auf sogenannten «*Wallets*[2]» (digitale Brieftasche) ist ein Beispiel dafür. Dezentrale Krypto-Börsen weisen keine zentrale Einheit auf. Das digitale Vermögen wird von den Benutzern verwaltet. Durch diese Gegebenheiten sind zentrale Krypto-Börsen vermutlich ein beliebteres Ziel für Angriffe. Die Gemeinsamkeit solcher Plattformen oder Desktopanwendungen ist der mangelnde Schutz der Privatsphäre der Benutzer. Bei zentralisierten Börsen müssen die Benutzer ihre Identität durch mehrere Dokumente (bspw. Pass, Stromrechnung und Selfie) beweisen. Dezentralisierte Börsen bieten einen gewissen Anonymitätsgrad an, wobei dieser nicht zwingend garantiert/gewährleistet werden kann. Denn die Identität eines Benutzers kann mit Hilfe von entsprechenden Ressourcen, wie bspw. der IP-Adresse, ermittelt werden. Deshalb spricht man im Krypto-Bereich von Pseudoanonymität.

Bei dem hier vorliegendem Forschungsbericht handelt es sich um ein Forschungsprojekt, welches vollständig dezentralisierte und Privatsphäre-schützende Handelsnetzwerke thematisiert und untersucht. Dieses Projekt wird zusammen mit dem Auftraggeber – Kopanyo AG – durchgeführt. Sie stellen für die Untersuchung einen Software Prototypen (diva.exchange) zur Verfügung. Ein wesentlicher Bestandteil des Forschungsprojektes ist die Sicherstellung der Anonymität von jedem Benutzer, welche zu jedem Zeitpunkt garantiert wird. Gemäss dem Auftraggeber existierte bis Ende Mai 2019 kein vollständig dezentrales und Privatsphäre-schützendes Handelsnetzwerk für digitale Werte. Es handelt sich bei diesem Projekt um ein Fachgebiet/Fachbereich, auf welchem bis heute nicht oder wenig geforscht wurde.

Das Ziel dieses Forschungsberichtes besteht darin, die Hypothesen (vgl. Kapitel 1.3), welche vom Auftraggeber gestellt wurden, zu falsifizieren. Um dieses Ziel zu erreichen werden deduktive sowie empirische Methoden angewendet. Das deduktive Vorgehen beinhaltet die Literaturrecherche und derer Ergebnisse. Daraus werden Herleitungen getätigt, welche für die Falsifikation der Hypothesen *[1]*, *[2]* und *[3]* verwendet werden. Das empirische Vorgehen wird für die Hypothese *[4]* durchgeführt und steht in einer Wechselwirkung mit der Weiterentwicklung des Software-Prototypen.

Die Gliederung des Forschungsberichtes beinhaltet die verwendeten Themen, welche für die Zielerreichung benötigt werden. Einen wesentlichen Bestandteil bildet die Anonymität. Das Kapitel 2 beschreibt die allgemeine Anonymität im Internet und unter welchen Bedingungen sie erfüllt wird. Daraus wird eine geltende Definition der Anonymität für dieses Forschungsprojekt formuliert.

[2] Ein «*Wallet*» ist eine digitale Brieftasche, welche die Haltung der jeweiligen Kryptowährung ermöglicht.

Weiter beschreibt das Kapitel 0 die einzelnen Komponenten des Software-Prototypen. Dies beinhaltet die wesentlichen Akteure des Systems sowie das Systemverhalten. Um das Systemverhalten zu beschreiben, werden im ersten Schritt Annahmen getätigt, welche ein ideales Systemverhalten beschreiben. Anschliessend werden Annahmen über mögliches korruptes Verhalten und die daraus resultierenden Gefahren beschrieben. Zum Schluss werden mögliche Gegenmassnahmen in Bezug auf korruptes Verhalten vorgestellt.

Den Bestandteil von Kapitel 4 bildet eine Marktanalyse, wo der Vergleich zweier ähnlicher Projekte durchgeführt wird. Die Literaturrecherche in Kapitel 5 beschreibt die externen Komponenten des Software-Prototypen. Das sind die Technologien I2P und die Iroha Blockchain. Dieses Kapitel soll ebenfalls als Dokumentation dienen. Der Auftraggeber möchte zukünftigen Projektbeteiligen dadurch den Einstieg in das Projekt vereinfachen.

Die Durchführung des empirischen Vorgehens wird durch die Testdurchführung in Kapitel 6 beschrieben. Bestandteile dieses Kapitels bilden die Laborumgebung und deren Architektur. Die verwendete Teststrategie, die Testkonfigurationen sowie die Testergebnisse, werden anschliessend festgehalten.

Durch die gesammelten Informationen aus der Literaturrecherche und mithilfe der generierten Testergebnisse, werden in Kapitel 7 die Hypothesen ausgewertet. Es wird konkret begründet, welche Hypothesen falsifiziert und welche nicht falsifiziert werden konnten.

In Kapitel 8 wird das gesamte Projekt reflektiert und zusammengefasst. Dazu gehört eine kritische Beurteilung. Weiteres mögliches Vorgehen wird für den Ausblick vorgeschlagen und beschrieben.

1.1 Thesen

Die Thesen bilden die Eigenschaften des Software-Prototypen ab. Folgende Thesen wurden vom Auftraggeber definiert:

- alle Daten und Prozesse sind vollständig dezentralisiert
- nicht-diskriminierend, dazu gehören insbesondere die Eigenschaften freier Software sowie tiefst mögliche Installations- und Betriebskosten
- ein hinreichend Privatsphäreschützendes Netzwerk erlaubt den Handel (Tausch, d.h. Kauf und Verkauf) von Blockchain-basierten digitalen Werten ("Krypto-Anlagen")
- die digitalen Werte sind vor unautorisierter Nutzung geschützt (Stichworte: Diebstahl, "double spending")
- die Installation kann für jeden aktiven Teilnehmer im Netzwerk vorteilhaft sein

Der Software Prototyp stellt somit die These dar und wird vom Auftraggeber auf einer Linux Plattform zur Verfügung gestellt. Es ist möglich, dass der Prototyp Programmierfehler und Funktionsmängel aufweist. In der ersten Phase des Projektes muss festgelegt werden, wie mit Fehlern und Mängeln umgegangen wird.

1.2 Ziele

Versuche zur Falsifikation der Hypothesen sind das Ziel des Forschungsberichtes. Als Resultat wird ein Dokument erwartet, welches die folgenden Themenkreise vertieft behandelt:

- Beschreibung der Ausgangslage und Voraussetzungen sowie der unterschiedlichen Versuchsanordnungen
- Durchführung der Versuchsreihen und deren Resultate
- Konklusion

1.3 Hypothesen

Gemeinsam wurden mit dem Auftraggeber folgende Hypothesen definiert:

[*H1*] mit einer steigenden Zahl von Netzwerkknoten reduziert sich die Ausfallwahrscheinlichkeit des gesamten Netzwerkes.

[*H2*] eine steigende Zahl von Netzwerkknoten führt zu einer steigenden Transaktions-Kapazität des gesamten Netzwerkes.

[*H3*] eine steigende Zahl von Netzwerkknoten erhöht die Resistenz gegen Brüche der Privatsphäre der Teilnehmer im Netzwerk.

[*H4*] eine steigende Blockhöhe beeinflusst das System und dessen Performanz nicht.

Die Ergebnisse der Hypothesen befinden sich in Kapitel 7.

2 Anonymität und Privatsphäre

Gemäss Duden [3] wird der Begriff Anonymität als «Nichtbekanntsein», «Nichtgenanntsein» oder «Namenlosigkeit» definiert. Aus diesem Grund verstehen wir unter vollständiger Anonymität einen Zustand, bei welchem nicht auf die wahre Identität eines Internetnutzers geschlossen werden kann. Dabei wird zwischen teilweiser und fehlender Anonymität unterschieden. Bei teilweiser Anonymität gibt der Internetnutzer gewisse Identifikationsmerkmale (bspw. Benutzername) mit, welche aber nicht vollumfänglich auf seine wahre Identität zurückschliessen lassen. Wenn hingegen offensichtliche Rückschlüsse auf die wahre Identität eines Benutzers gemacht werden können, spricht man von fehlender Anonymität.

Gary T. Marx [4] beschreibt sieben Identifikationsmerkmale, mittels welcher eine Person charakterisiert werden kann. Diese beinhalteten einerseits den {1} Namen einer Person, welcher sich aus einem Vor- und Nachnamen zusammensetzt. Weiter lässt sich eine Person durch die {2} Adresse (E-Mailadresse, Postanschrift, Telefonnummer, Fax usw.) identifizieren. Ein weiteres Merkmal sind die {3.1} alphanumerischen Symbole. Diese können direkt einer Person zugewiesen werden können. Das können eine Passnummer, AHV-Nummer oder di biometrischen Informationen sein. Ebenfalls gibt es {3.2} Pseudonyme, wie bspw. einen Benutzernamen, welche in der Regel nicht einer Person zugeordnet werden können. Marx beschreibt ebenfalls zwei weitere Identifikationsmerkmale, wobei durch diese die wahre Identität einer Person nicht bekannt ist. Diese sind das {4} Aussehen sowie das {5} Verhaltensmuster einer Person. Ein weiteres Merkmal ist die {6} soziale Kategorisierung. Dabei wird u.a. zwischen Geschlecht, Alter, Sprache und Religionsausrichtung unterschieden/differenziert. Das letzte Identifikationsmerkmal lässt sich durch {7} bestimmte Fähigkeiten oder Wissen beschreiben. Eine bestimmte Fähigkeit kann bspw. Balancieren sein, wobei das Wissen sich bspw. durch den Besitz von Passwörtern beschreiben lässt.

2.1 Anonymität im Internet

Der Einsatz von Technologien zum Schutz der Anonymität im Internet kann durch Verschlüsselungs- oder Weiterleitungsdienste erreicht werden. Ebenfalls ermöglicht die Verwendung von Pseudonymen und speziellen Netzwerktechnologien (bspw. I2P oder Tor) das Verschleiern der eigenen Identität. Dadurch wird beliebigen Dritten die Nachvollziehbarkeit erschwert.
In Anbetracht der Effektivität der eingesetzten Technologien, welche am daraus resultierenden Anonymitätsgrad gemessen werden, könnte die wahre Identität durchaus technisch ermittelt werden. Solch ein Vorgehen ist jedoch einerseits mit erhöhtem zeitlichem Aufwand und andererseits mit grossem Einsatz von Ressourcen verbunden. Durch technische Massnahmen, wie bspw. die Verwendung von Proxyservern, wird dies im besten Fall jedoch verunmöglicht. Entscheidend ist die Differenz zwischen individuell wahrgenommener Anonymität des Nutzers und der tatsächlichen informationstechnologischen Daten. Ohne die Verwendung der erwähnten Technologien, wie bspw. I2P, ist der Nutzer nicht anonym im Internet. Einerseits ist der Rechner durch die IP-Adresse eindeutig identifizierbar und andererseits hinterlässt man auch Spuren bei genutzten Diensten und Webseiten, welche es ermöglichen den Nutzer zu identifizieren, dessen Verhaltensmuster zu erfassen und ein Persönlichkeitsprofil bezüglich seines Surfverhaltens zu erstellen.

2.2 Definition der Anonymität und Privatsphäre für das Projekt

Die Anonymität sowie die Privatsphäre der Benutzer stehen bei diesem Projekt im Vordergrund. Aus diesem Grund wurde für das Projekt diva.exchange mit dem Auftraggeber zusammen folgendes definiert:

«Das System diva.exchange stellt sicher, dass keine Identifikationsmerkmale [4] *eines Benutzers die Systemgrenzen verlassen. diva.exchange stellt daher sicher, dass jeder Benutzer ausschliesslich seine eigenen Identifikationsmerkmale kennt. diva.exchange nutzt einen bedingungslos öffentlichen persistenten Datenspeicher. diva.exchange stellt daher ebenfalls sicher, dass der öffentliche Datenspeicher keine Identifikationsmerkmale von Benutzern beinhaltet.»*

Um einen hohen Anonymitätsgrad zu erreichen wird auf der Netzwerkschicht eine Technologie verwendet, welche die Anonymität sowie Privatsphäre der Akteure gewährleistet. Ebenfalls benötigt es für die Registrierung bei diva.exchange keine Angaben oder Dokumente der Akteure, welche auf deren Identität zurückzuführen lassen. Im Verlauf des vorliegenden Forschungsberichtes wird ein Vergleich zweier ähnlicher Netzwerktechnologien durchgeführt. Dies wird in Kapitel 5.2.1.1 erläutert. Somit ist jedes Indiz, wie bspw. die Zuweisung der IP-Adresse zu einem Akteur, welcher der tatsächliche Sender einer Nachricht ist, ein Bruch der Privatsphäre und Anonymität. Die einzige Komponente, welche im öffentlichen Netzwerk ersichtlich ist, bildet die Blockchain ab. Ebenfalls ist die Blockchain als öffentlich klassifiziert und somit hat jeder Einsicht auf die persistierten Transaktionen und deren Daten. Die Abbildung 1 zeigt den Aufbau und unterscheidet zwischen dem Darknet (I2P) sowie dem Clearnet (öffentliches Netz). Ebenfalls ist ersichtlich, dass es zwei Akteure – «*full*» und «*light*» – gibt. Diese werden in Kapitel 3.2 erläutert. Das I2P-Netzwerk, welches als Darknet illustriert wird, besteht aus diversen Netzwerkknoten. Dabei wird zwischen Netzwerkknoten, welche diva.exchange verwenden («*diva_full_node*») und derer, welche diva.exchange nicht verwenden («*node*»), unterschieden. In Kapitel 3.3.2 wird ein korruptes Systemverhalten erläutert, welches ebenfalls zu einem Bruch der Anonymität sowie der Privatsphäre führen könnte.

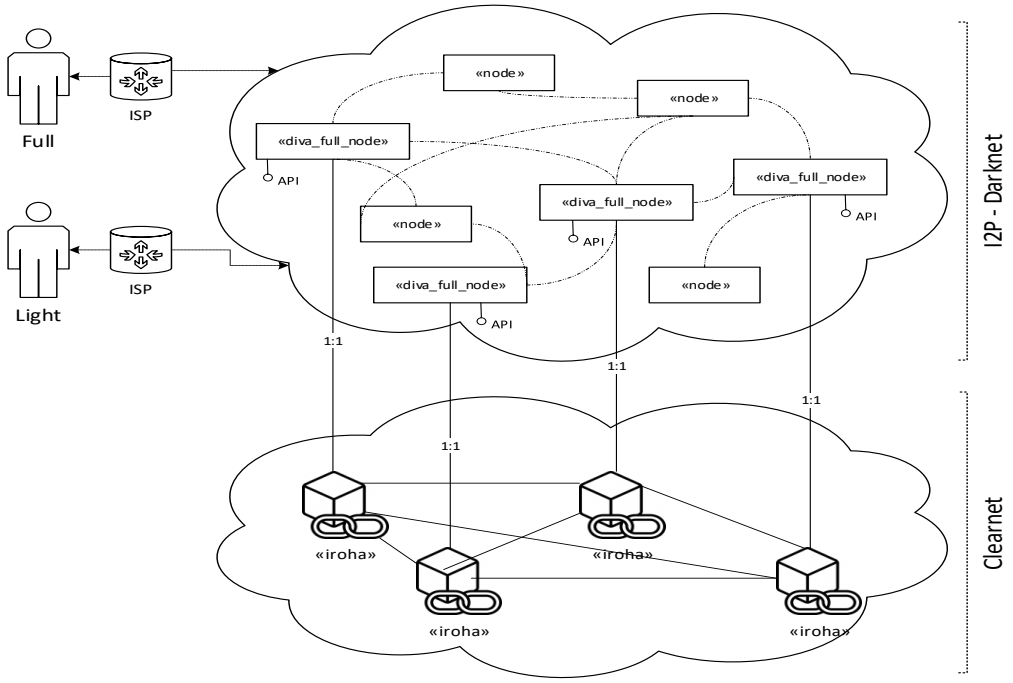

Abbildung 1. Diva full und light Clients - Darknet -Clearnet.

3 Systemmodel

In diesem Kapitel werden der Stack sowie die verschiedenen Akteure von diva.exchange vorgestellt und erläutert. Ausserdem wird das gewünschte sowie das mögliche korrupte Systemverhalten beschriebenen. Die Funktionalitäten des Systems bilden auch einen Bestandteil dieses Abschnittes. Dabei werden präventive Massnahmen gegen ein korruptes Systemverhalten sowie korrupte Akteure beschrieben.

3.1 diva.exchange Stack

In diesem Abschnitt werden die Komponenten, welche bei diva.exchange verwendet werden, vorgestellt. Dabei wird zwischen Komponenten aus Eigenentwicklung (intern) und externer Verwendung (extern) unterschieden. In der Tabelle 1 werden diese erläutert.

Komponente	**Beschreibung**	**Intern/Extern**
Diva	Dies ist die Anwendung, welche sowohl das Verwalten als auch das Handeln von digitalen Werten ermöglicht. Bei der Installation wird zwischen einer «*full*» und «*light*» Version unterschieden. Alle verwendeten Services befinden sich lokal auf dem Endgerät.	Intern
I2P	Dies ist die Komponente, welche die private und sichere Kommunikationsschicht repräsentiert und gewährleistet.	Extern
Hangout	Mittels dieser Komponente wird ein Rendezvous-Server bereitgestellt. Dieser stellt bidirektionale Verbindungen zwischen Internethosts in privaten Netzwerken her, indem er Netzwerkadressübersetzer verwendet. Es wird ebenfalls ein Port-Matching durchgeführt, welches die Kommunikation erlaubt.	Intern
Iroha	Diese Komponente repräsentiert die Blockchain von Hyperledger.	Extern
Logger	Ist eine generische Logger-Komponente.	Extern

Tabelle 1. diva.exchange Komponenten.

3.2 Akteure

Akteure werden in diesem Forschungsbericht als Benutzer B oder Komponenten K des Systems (Hardware und Software) mit spezifischen Rollen R definiert, welche sich auf der Blockchain befinden. Es wird dabei zwischen «*full*» und «*light*» Akteuren differenziert. Ebenfalls können Komponenten innere Komponenten K_{Inner} beinhalten.

B_{Full}: Wird definiert als B, welcher die diva.exchange Software herunterlädt und auf seinem Endgerät (bspw. Laptop) installiert. Dabei sind sämtliche Komponenten enthalten (vgl. Tabelle 1).

B_{Light}: Wird definiert als B, welcher die diva.exchange Software herunterlädt und auf seinem Endgerät installiert. Dabei werden K_{Iroha} sowie $K_{Hangout}$ im Gegensatz zu B_{Full} nicht installiert.

$R_{Developer}$: *B* mit den meisten Rechten und somit der mächtigsten *R*.

R_{User_API}: Ist B_{Full}. Diese *R* ermöglicht das Schreiben auf K_{Iroha}.

$R_{Auctioneer}$: Ist B_{Full}. Diese *R* übernimmt das Verknüpfen von Angebot und Nachfrage im Orderbuch.

$R_{Subscriber}$: Diese *R* wird zum jetzigen Stand (15.01.2010) jedem neuen *B* (B_{Full} oder B_{Light}) standardgemäss zugewiesen. Mit dieser *R* kann ein *B* über die API von K_{Iroha} lesen und über R_{User_API} eigene Angebote ins Orderbuch hineinstellen.

K_{Iroha}: Diese *K* wird bei der «*full*» Variante installiert. Sie repräsentiert die Datenschicht und ist die öffentliche Blockchain. Iroha besteht aus weiteren K_{Inner}. Diese werden in Kapitel 5.3 und den folgenden Unterkapiteln erläutert. K_{Iroha} bietet die Funktionen des Lesens («*read*») und des Schreibens («*write*») an.

K_{I2P}: Diese *K* stellt die Netzwerkschicht dar und garantiert die Anonymität eines *B* und schützt dessen Privatsphäre. Hierbei muss hinzugefügt werden, dass dieses Netzwerk aus verschiedenen Teilnehmern besteht. B_{Full} und B_{Light} sind unteranderem Teilnehmer dieses Netzwerks. Das Netzwerk besteht ebenfalls aus nicht klassifizierbaren I2P-Knoten, welche die diva.exchange Software nicht verwenden. Diese *K* steht in direkter Relation mit den Hypothesen *[1]*, *[2]* und *[3]* (Vgl. Kapitel 1.3).

3.3 Systemverhalten

Um ein korruptes Systemverhalten zu beschreiben, muss im ersten Schritt definiert werden, was von allen Akteuren erwartet wird, damit ein ideales Systemverhalten besteht. Danach werden mögliche Gründe für ein korruptes Systemverhalten und die daraus resultierenden Gefahren erläutert. Zum Schluss werden präventive Massnahmen des Systems vorgeschlagen, welche dazu dienen können, ungewünschtes Verhalten zu unterbinden.

3.3.1 Verhaltensannahmen – Ehrliche Akteure

In Kapitel 3.2 wurden die verschiedenen Akteure und deren Zuständigkeiten definiert. Daraus werden nun Annahmen getroffen, wie sich die Akteure im Idealfall verhalten.

Ehrliche Annahme 1 – A1
Bei den Akteuren B_{Full} und B_{Light} handelt es sich um ehrliche Akteure. Dies bedeutet, dass alle Parteien, die miteinander handeln, nicht versuchen die Gegenpartei zu betrügen. Alle Parteien liefern die ausgehandelten digitalen Werte (Kryptowährungen) an die Gegenpartei bis zu einem definierten Zeitpunkt.

Ehrliche Annahme 2 – A2
K_{I2P} besteht aus diversen Netzwerkknoten mit unterschiedlichen Internet Service Providern, wodurch die Anonymität sowie Privatsphäre jedes *B* garantiert wird.

Ehrliche Annahme 3 – A3
Das System ist so ausgelegt, konzipiert und entwickelt, dass die digitalen Werte der einzelnen Akteure nicht gestohlen oder verfälscht werden können. Ebenfalls werden Transaktionen korrekt überprüft und abgewickelt.

Ehrliche Annahme 4 – A4
Gitlab-Administratoren lassen Codeänderungen nur dann zu, wenn diese genau überprüft worden sind. Da es sich um ein Open-Source Projekt handelt, kann jedermann Änderungen am Code, welcher sich in der Gitlab-Repository [5] befindet, vorschlagen.

3.3.2 Verhaltensannahmen - Korrupte Akteure

Korruptes Verhalten K1 in Bezug auf A1
Bei den Akteuren B_{Full} und B_{Light} handelt es sich mindestens um einen korrupten (unehrlichen) Akteur. Dieser hat das Ziel die Gegenpartei zu betrügen. Unter der Annahme, dass die ehrliche Partei die digitalen Werte liefert, jedoch die unehrliche Partei dies nicht tut, liegt hier ein Betrug vor. Bspw. will der Akteur E_{hrlich} einen Bitcoin gegen fünf Ethereum-Token handeln. Akteur K_{orrupt} möchte fünf Ethereum-Token gegen einen Bitcoin handeln. Hier tritt nun das «*Matching*[3]» ein. Nun wird *E* aufgefordert, seinen Bitcoin an *K* und dessen «*Wallet-Adresse*[4]» zu transferieren. Akteur *E* tut dies und wartet nun auf die Transferierung von *K*. Dieser sendet nicht wie vereinbart die fünf Ethereum-Token an die «*Wallet-Adresse*» von *E*. Nun hat *E* alles verloren und *K* hat sein digitales Vermögen erhöht.

Korruptes Verhalten K2 in Bezug auf A2
Es wird angenommen, dass sämtliche I2P-Nutzer denselben Internet Service Provider nutzen (Bspw. Swisscom). Somit können sämtliche Hops im I2P-Netzwerk nachverfolgt werden. Dies bedeutet, dass der Provider durch Mustererkennungsalgorithmen in der Lage wäre, einen Akteur zu identifizieren. Die Datenpakete sind trotzdem verschlüsselt und können mehrere Empfänger mit verschiedenen Nachrichten enthalten. Die Entschlüsselung dieser Datenpakete wäre theoretisch mit den entsprechenden Ressourcen möglich. Dies wird jedoch in diesem Forschungsbericht nicht thematisiert. Zwischen den «*full*» Clients und den Iroha-Instanzen besteht eine 1:1 Relation. Wiederum könnte hier durch Mustererkennungsalgorithmen mit einer gewissen Wahrscheinlichkeit bestimmt werden, dass es sich bei diesem I2P-Knoten um einen «*full*» Client von diva.exchange handelt.

Korruptes Verhalten K3 in Bezug auf A3
Das System könnte durch Sicherheitslücken oder gezielte Angriffe einer Person Zugang auf die Server ermöglichen. Die digitalen Werte könnten dadurch gestohlen, respektive an andere «*Wallet-Adressen*» transferiert werden. Ebenfalls kann der Angreifer dadurch sensible Daten der Benutzer einsehen, wodurch sie identifiziert werden könnten.

Die Überprüfung und die Korrektheit einer Transaktion wird mit dem «*yet ynother consensous (YAC)*» Konsensalgorithmus durchgeführt. Sämtliche «*full*» Clients im Netzwerk agieren als Prüfstellen einer Transaktion. Sobald die die Mehrheit – $(\frac{Teilnehmer}{2}) + 1$ – erreicht wird, erachtet das Netzwerk die Transaktion als gültig. Bei der Anzahl von 18 «*full*» Clients müssen somit mindestens zehn davon eine Transaktion als gültig erachten, damit die Einigung erreicht wird. Unter der Annahme, dass die Mehrheit der «*full*» Clients korrupt ist, kann die Korrektheit einer Transaktion nicht gewährt werden.

---

[3] Im Gegensatz zu Bisq gibt es zentralisierte Börsen, wie bspw. den Swiss Crypto Exchange (SCX), wo ein automatisches Verknüpfen von Angebot und Nachfrage durchgeführt wird.
[4] Die «*Wallet-Adresse*» repräsentiert eine einzigartige Zeichenfolge und ist der öffentliche Schlüssel des Besitzers. Sie kann mit einer IBAN-Nummer einer Bankkarte verglichen werden. Möchte man eine gewisse Anzahl Bitcoins einer Person überweisen, so benötigt man dessen Wallet-Adresse.

Korruptes Verhalten K4 in Bezug auf A4
Wenn sich ein Angreifer Zugangsdaten des Administrators beschaffen kann, ist er in der Lage Code-Änderungen vorzunehmen und diese den Akteuren (B_{Full} und B_{Light}) zu unterbreiten. Denkbare und gezielt bösartige Codeänderungen könnten möglicherweise folgende sein:

- Deaktivierung des I2P-Netzwerkes (Darknet) und Verwendung des Öffentlichen Netzes (Clearnet) für alle Komponenten.
- Eine Einschleusung eines böswilligen Programmes, welches Daten der Akteure sammelt und diese an eine externe Stelle weiterleitet.
- Source-Code so verändern, dass bei jeder Transaktion die «*Wallet-Adresse*» des Angreifers beim ehrlichen Akteur als Empfängeradresse angezeigt wird. Der Angreifer erhält somit bei jeder Transaktion die digitalen Werte und liefert im Gegenzug keine an die ehrliche Partei.

3.3.3 Gegenmassnahmen

Gegenmassnahme G1 in Bezug auf K1
Es wurden bis dato noch nicht entschieden, welche Gegenmassnahmen implementiert werden, um betrügerisches Verhalten zu unterbinden. Eine einfache jedoch effektive Möglichkeit als Schutzmechanismus wäre, dass jeder Akteur nur die Hälfte der digitalen Werte handeln kann, welche er in einem Tresor («*Wallet-Adresse*») hinterlegt. Dies bedeutet, dass ein digitaler Betrag *x* nur dann gehandelt werden kann, wenn im Tresor der digitale Betrag *2x* vorhanden ist. Damit kann die ehrliche Partei, welche sich an die Regeln haltet, entschädigt werden und die unehrliche Partei kann für die Nichteinhaltung der Regeln sanktioniert werden. Dieser Sicherheitsmechanismus wird anhand eines Beispiels mit jeweils einem Akteur E_{hrlich} und einem Akteur K_{orrupt} aufgezeigt. In der Tabelle 2 sind die digitalen Bestände der Akteure vor dem Handel ersichtlich. Hierbei wird zwischen zwei «*Wallets*» unterschieden. Einerseits dem «*Trading-Wallet*», welches für den Handel benötigt wird und andererseits dem Tresor, welcher mindestens die zweifache Menge des digitalen Wertes des «*Trading-Wallets*» repräsentiert.

Akteur	Trading-Wallet	Tresor	Total
E_{hrlich}	5 Ethereum-Token	10 Ethereum-Token	15 Ethereum-Token
K_{orrupt}	1 Bitcoin	2 Bitcoins	3 Bitcoins

Tabelle 2. Digitale Bestände der Akteure vor dem Handel.

1. *E* möchte 5 Ethereum-Token gegen einen Bitcoin handeln.
    a. Es wird überprüft, ob *E* mindestens zehn Ethereum-Token im Tresor hinterlegt hat.
    b. Ist dies der Fall, kann er seine Bestellung im Orderbuch platzieren.

2. *K* möchte einen Bitcoin gegen 5 Ethereum-Token handeln.
    a. Es wird überprüft, ob *K* mindestens zwei Bitcoins im Tresor hinterlegt hat.
    b. Ist dies der Fall, kann er seine Bestellung im Orderbuch platzieren.

3. Es wird ein «*Matching*» der beiden platzierten Bestellungen durchgeführt.

4. *E* wird aufgefordert, die 5 Ethereum-Token an die «*Wallet-Adresse*» von *K* innert definierter Zeit zu transferieren. Er hält sich daran.

5. *K* wird aufgefordert, einen Bitcoin an die «*Wallet-Adresse*» von *E* innert definierter Zeit zu transferieren. Er hält sich nicht daran.

6. In diesem Fall wird ein Bitcoin aus dem Tresor von *K* automatisch an die «*Wallet-Adresse*» von *E* übertragen. Ebenfalls wird *K* sanktioniert, indem er zusätzlich noch einen Bitcoin an das Netzwerk (diva.exchange) bezahlen muss. Die Sanktionierung wird ebenfalls automatisch durchgeführt.

In diesem Beispiel wird die Transaktion von *E* als korrekt empfunden. Er ist nun im Besitzt von einem Bitcoin, welchen er aus dem Tresor von *K* erhalten hat. Im Tresor von *E* befinden sich weiterhin 10 Ethereum-Token. *K* wiederum ist in Besitz von 5 Ethereum-Token. In seinem Tresor befinden sich keine Bitcoins mehr. Gemäss diesem Szenario hat *E* bei diesem Handel keinen Verlust gemacht. *K* muss einen Verlust von zwei Bitcoins verbuchen. In der Tabelle 3 sind die Bestände der jeweiligen Akteure nach dem Handel ersichtlich. Das Netzwerk ist in diesem Beispiel ebenfalls ein Akteur, da dieses einen Bitcoin von *K* erhalten hat.

Akteur	Trading-Wallet	Tresor	Total
E_{hrlich}	1 Bitcoin	10 Ethereum-Token	1 Bitcoin 10 Ethereum-Token
K_{orrupt}	5 Ethereum-Token	0 Bitcoin	0 Bitcoins 5 Ethereum-Token
Netzwerk	-	1 Bitcoin	1 Bitcoin

Tabelle 3. Digitale Bestände der Akteure nach dem Handel.

Gegenmassnahme G2 in Bezug auf K2
Die Annahme, dass sämtliche I2P-Netzwerknoten denselben Internet Service Provider verwenden kann als unwahrscheinlich eingestuft werden. In Amerika gibt es gemäss der Plattform «*Broadbandnow*» ungefähr 2'677 (15.01.2020) unterschiedliche Internet-Service Provider [6]. Deshalb wird angenommen, dass es weltweit vermutlich über 3'000 unterschiedliche Internetservice Provider gibt oder auch mehr. Durch diese Annahme kann die Wahrscheinlichkeit, dass alle I2P-Netzwerkknoten denselben Internet Service Provider verwenden, als gering eingestuft werden. Eine Gegenmassnahme gegen dieses Szenario gibt es jedoch nicht. Würde eine weltweite Fusion der bestehenden Internet Service Provider Zustandekommen, könnte trotz der I2P-Technologie die Anonymität und Privatsphäre der Akteure nicht mehr gewährleistet werden.

Gemäss Statistik von den betriebenen I2P-Netzwerkknoten, gibt weltweit ungefähr 27'232 I2P-Netzwerkknoten (Stand 14. Januar 2020). Die meisten befinden sich in Amerika, England und Russland. Dies bekräftigt, dass das vorhin genannte Szenario der Fusion als unwahrscheinlich eingestuft werden kann [7].

Die Tatsache, dass zwischen den «*full*» Clients (I2P-Netzwerk) und den Iroha-Instanzen (öffentliches Netzwerk) eine 1:1 Beziehung besteht, ermöglicht die Annahme, dass es sich bei diesen I2P-Netzwerkknoten um solche handelt, welche die diva.exchange Software installiert haben. Mustererkennungsalgorithmen können das mit einer gewissen Wahrscheinlichkeit untermauern. Diese Tatsache kann nicht unterbunden werden. Das bedeutet jedoch nicht, dass Transaktionen von einem Sender auf dessen Identität schliessen lassen können. Der Sender ist nie derjenige, der seine eigene Transaktion auf die Blockchain schreibt. Somit werden die Anonymität sowie die Privatsphäre des Benutzers nicht beeinträchtigt.

Gegenmassnahme G3 in Bezug auf K3
Angriffe auf einzelne Knoten sind möglich, jedoch nicht effektiv, da diva.exchange dezentral ausgerichtet ist. Damit ein Schaden entsteht, müssten alle diva.exchange Knoten angegriffen werden, was viele Ressourcen verwenden würde. Ein Angriff, der mehr Ressourcen (Kosten) benötigt als er Schaden (Ertrag) anrichtet, ist vermutlich nicht sinnvoll.

«*Full*» Clients entscheiden miteinander, ob eine Transaktion korrekt ist oder nicht. Dabei muss die Einigung von mindestens – $(\frac{Teilnehmer}{2}) + 1$ –erreicht werden. Damit gewährleistet werden kann, dass es sich bei den «*full*» Clients nicht um korrupte Akteure handelt, kann ein Reputationssystem verwendet werden. Lediglich Akteure, welche ein bestimmtes Level aufweisen, können «*full*» Clients betreiben. Ausserdem wäre es wahrscheinlich von Vorteil, wenn diese Akteure ein gewisses Sicherheitsdepot hinterlegen, um zu bekräftigen, dass es auch in ihrem Interesse liegt, dass Transaktionen korrekt abgewickelt werden.

Gegenmassnahme G4 in Bezug auf K4
Hierbei gilt die Verwendung von sicheren Passwörtern, welche sich mittels Software (bspw. KeePassX[5]) erstellen und verwalten lassen. Ausserdem kann die Sicherheit durch die Verwendung einer Zwei-Faktor-Authentifizierung erhöht werden. Es ist zu empfehlen, dass Änderungen am Sourcecode und deren Genehmigung, von mehreren Parteien zuerst begutachtet werden müssen. Es benötigt dann von allen Beteiligten die Genehmigung, bevor Änderungen publiziert werden.

[5] KeePassX ist eine Passwortverwaltungssystem-Datenbank. Damit lassen sich Passwörter generieren und der Zugang zu ihnen kann mit einem Passwort, wie auch zusätzlich mit einem privaten Schlüssel, gesichert werden.

4 Marktanalyse

In den folgenden Kapiteln werden zwei Krypto-Börsen untersucht, welche Ähnlichkeiten zum jetzigen Projekt – diva.exchange – aufweisen. Dabei handelt es sich um Bisq und Resistance.

4.1 Bisq Exchange

Dieses Kapitel stützt sich auf den Vortrag des 34. Chaos Computer Kongresses, vom 30. Dezember 2017 in Leipzig, vom Bisq Mitbegründer Chris Beams. Der Vortrag wurde als Video auf einer Webseite hochgeladen [8]. Ebenfalls werden Informationen aus dem öffentlichen Whitepaper von Bisq verwendet [9].

Bei Bisq handelt es sich um eine plattformunabhängige Desktopanwendung, mit welcher Benutzer Bitcoin gegen diverse Landeswährungen (Fiat) oder andere Kryptowährungen handeln können. Den Benutzern wird mit dem Bisq-Protokoll der direkte Handel untereinander angeboten. Somit benötigt es keine Drittparteien für das sichere Handeln. Bei Bisq handelt es sich um ein «*peer-to-peer*» Netzwerk, welches durch die Desktopanwendung und das Bisq-Protokoll kreiert wird. Bisq besteht aus einer internationalen Gruppe von Entwicklern, welche die freie Software der Öffentlichkeit zur Verfügung stellt. Gemäss Chris Beams ist Bisq kein Unternehmen. Die Hauptziele von Bisq sind:

Sicherheit
Durch die dezentrale Ausrichtung befinden sich die Gelder der Bisq Benutzer zu keinem Zeitpunkt auf einem Server. Es können beliebige Dienste («*Wallets*») für die Haltung der Kryptowährungen benutzt werden. Ebenfalls bietet die Bisq Desktopanwendung einen solchen Dienst an und somit fungiert das Endgerät, worauf die Software installiert wird, als «*Wallet*». Benutzer müssen aus präventiven Gründen Sicherheitsleistungen einzahlen, um dem Betrug bei einem Handel entgegenzuwirken. Zusätzlich werden die Sicherheitsleistungen und die gehandelten Werte durch zwei bis drei Multisignaturen in einem Treuhandkonto (Tresor) hinterlegt. Wenn es bei einem Handel zwischen zwei Parteien zu Unstimmigkeiten kommt, tritt ein dezentrales Schiedsrichtersystem ein. Dahinter befinden sich Menschen, die zur Schlichtung und Lösung des Problems beitragen sollen.

Privatsphäre
Persönliche Informationen der Benutzer, wie der Name, das Geburtsdatum und die Wohnadresse werden auf keiner zentralen Einheit, bspw. einem Server oder einer Datenbank, gespeichert. Es benötigt keine Registration, um die Desktopanwendung zu verwenden. Jede Bisq-Desktopanwendung ist ein versteckter Tor-Dienst.

Zensurresistenz
Bisq Benutzer können ohne Einschränkungen oder durch die Einwirkung intermediärer Stellen direkt zwischen einander handeln. Durch das vollständig verteilte «*peer-to-peer*» Netzwerk von Bisq, sind einzelne Fehlerquellen von Nodes für das Gesamte Bisq-Netzwerk nicht gravierend. Das Bisq Netzwerk baut auf Tor auf und erbt somit dessen Zensurwiederstand.

4.1.1 Handelsfunktion bei Bisq

Bisq unterscheidet sich von anderen Börsen, da es keine Webapplikation ist. Ebenfalls gibt es kein automatisiertes verknüpfen der Angebote («*Matching*»). Angenommen, der Benutzer *B* möchte einen Bitcoin gegen 20 Litecoins handeln, dann veröffentlicht er das im Orderbuch. Der Benutzer *L* möchte 20 Litecoins gegen einen Bitcoin handeln und veröffentlicht dies ebenfalls im Orderbuch. Im Gegensatz zu anderen Börsen werden diese beiden Angebote bei Bisq nicht verknüpft («*Matching*»). Der Benutzer *B* oder *L* muss bei Bisq explizit das gewünschte Angebot auswählen, damit es zum Handel kommt. Derjenige Benutzer, der das Angebot des anderen auswählt, wird als «*buyer*» bezeichnet und der andere als «*seller*». Ein Handel zwischen zwei Benutzern kann wie folgt aussehen.

Angenommen, dass der Benutzer *B* Bitcoins in Schweizer Franken *CHF* kaufen möchte. *B* ist somit der «*buyer*» und sucht einen anderen Benutzer *S «seller»*, welcher diesen Handel in *CHF* akzeptiert, damit der Handel stattfinden kann.

1. *B* muss die gewünschte Zahlungsmethode konfigurieren. Dies beinhaltet bspw. die IBAN-Nummer, den Banknamen und Informationen des Bankkontoinhabers.
2. *B* sucht im Orderbuch nach Angeboten in der gewünschten Währung.
3. *B* wählt das gewünschte Angebot aus und bestätigt den Kauf der Bitcoins von *S.*
4. *B* werden nun die Bankkontoinformationen von *S* angezeigt. *B* überweist den Betrag an das Bankkonto von *S* und bestätigt die Überweisung in der Bisq-Desktopanwendung.
5. *B* und *S* warten nun bis die Überweisung ankommt.
6. *S* erhält den Betrag auf seinem Bankkonto und bestätigt den Erhalt in der Bisq-Anwendung.
7. *B* erhält nun die Bitcoins von *S* auf seinem hinterlegten «*Wallet*».

Dies ist eine Variante, wie ein Handel abgewickelt werden kann. Je nach Zahlungsmethode oder Rolle («*buyer*» oder «*seller*»), kann dies jedoch variieren und von den Schritten eins bis sieben abweichen.

Bei diesem Beispiel wurde ein Handel zwischen der Kryptowährung Bitcoin und der Fiat-Währung Schweizer Franken CHF aufgezeigt. Die Bisq-Desktopanwendung hat jedoch keine integrierten Fiat-Schnittstellen zu Banken. Ebenfalls agiert sie nicht als Drittpartei mit einem eigenen Bankkonto, bei welchem die Benutzer die Fiat-Währungen einzahlen müssen und Bisq danach die Fiat-Währung dem Bankkonto des Benutzers überweist. Während des Handels übernimmt die Bisq-Desktopanwendung die Koordination beider Parteien und intragiert/interagiert zwischen ihnen durch Bestätigungen[6]. Der Tatsache entsprechend, dass keine intermediären Stellen während des Handels eingreifen, dauert der Handel mittels der Bisq-Desktopapplikation vermutlich länger als bei zentralisierten Börsen.

[6] Vgl. Schritte eins bis sieben.

4.1.2 Auswertung – Bisq

Die Verfasser vom Bisq Whitepaper stützen sich unterandrem auf die Punkte Sicherheit und Privatsphäre. Sicherheit wird durch die dezentrale Ausrichtung angeboten, da es keine zentrale Einheit gibt, welche die digitalen Werte (bspw. Bitcoin), für die Benutzer haltet. Ausserdem wird durch die Einzahlung von Sicherheitsleistungen, dem Betrug präventiv entgegengewirkt. Diese zwei Punkte können jedoch kritisch betrachtet werden. Angenommen, dass es bei einem Handel zwischen zwei Parteien zu einem Missverständnis kommt, dann tritt eine Drittpartei als Schiedsrichter ein. Die Schiedsrichterpartei ist eine Person, welche sich mit den beiden Parteien in Verbindung setzt, falls es zu Unstimmigkeiten kommt. Sobald die Schiedsrichterpartei sich einmischt, ist die Privatsphäre und die Anonymität der beiden Parteien gebrochen.

In Kapitel 4.1.1 wurde die Handelsfunktion bei einem Kauf von Bitcoin gegen Schweizer Franken CHF erläutert. Dabei wird von den Benutzern die Zahlungsfunktion konfiguriert, indem persönliche Informationen (Vgl. Abbildung 3) angegeben werden, welche die Eindeutigkeit einer Person bestätigen. Bei einem Kauf von Bitcoins gegen Schweizer Franken CHF erscheinen dem «*buyer*» die Kontoinformationen vom «*seller*». Ab diesem Zeitpunkt ist die Privatsphäre vom «*seller*» nicht mehr gewährleistet. Wenn die Schweizer Franken beim «*seller*» auf dem Bankkonto einbezahlt wurden, sieht dieser die Informationen vom «*buyer*». Ab diesem Zeitpunkt ist die Privatsphäre beider Parteien gebrochen, denn beide kennen nun die echte Identität des anderen. Bei einem Handel ohne Fiat-Währungen wird die Privatsphäre nicht gebrochen, da in diesem Fall lediglich die «*Wallet-Adresse*» sowie die «*Onion-ID*» (Vgl. Abbildung 2) des Benutzers angezeigt werden.

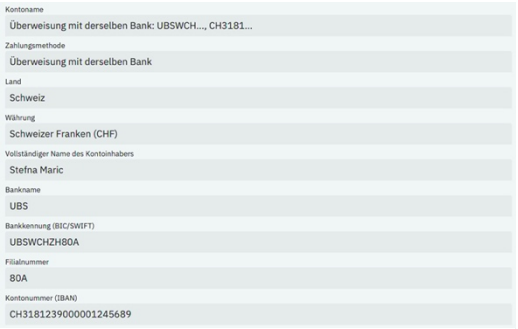

Abbildung 3. Bisq-Desktopanwendung – Bankkonto.

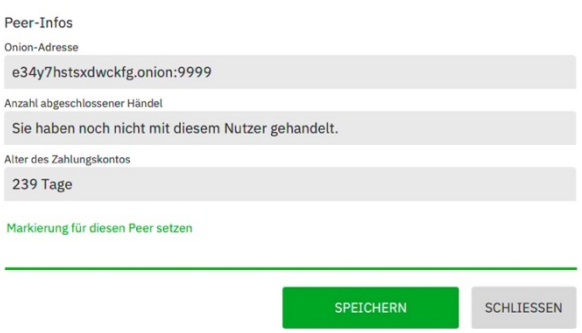

Abbildung 2. Bisq-Desktopanwendung - Peer-Information.

Ein weiterer Punkt, welcher sich vom jetzigen Projekt unterscheidet, bilden die handelbaren Kryptopaare. Bei Bisq wird bei jeder Transaktion immer gegen Bitcoin gehandelt. Das bedeutet, dass ein Handel zwischen zwei unterschiedlichen Kryptowährungen, wobei keine Bitcoin ist, nicht stattfinden kann. Wer bspw. nur in Besitz von Ethereum ist, muss diese zuerst gegen Bitcoin handeln, damit er gegen andere Kryptowährungen, wie bspw. Monero, handeln kann. Dies gilt für den Fiat- sowie Kryptohandel. Bisq unterhält ebenfalls einen eigenen Token (BSQ), welcher dazu verwendet wird, um die Entwickler und Geldgeber zu entschädigen. Dies ist ein weiterer Unterschied zum diva.exchange.

4.2 Resistance Exchange

Das folgende Kapitel stützt sich auf das Whitepaper von Resistance [10]. Resistance ist eine dezentrale Börse, welche durch eine private Blockchain unterstützt wird. Dazu wird der «*RES privacy Coin*» als intermediäre Stelle verwendet. Durch das Interagieren dieser Komponenten wird der Handel sowie das Mining[7] ermöglicht. Das grundlegende Prinzip von Resistance bildet die Privatsphäre. Dafür wird das Zero-Knowledge-Beweis («*zero-knowledge proof*») Protokoll eingesetzt, damit die getätigten Transaktionen auf einem privaten Level stattfinden. Dieses Protokoll stammt aus dem kryptografischen Bereich. Dabei kommunizieren in diesem Protokoll der zu beweisende (Beweiser) sowie der zu verifizierende (Verifizierer) miteinander. Dabei versucht der Beweiser den Verifizierer zu überzeugen, dass er ein Geheimnis mit gewisser Wahrscheinlichkeit kennt, ohne das Geheimnis Preis zu geben [11]. Eine andere Privatsphäre schützende Eigenschaft wird mit der Tor-Funktionalität geboten.

4.2.1 Handelsfunktion – Atomic Swaps

«*Atomic Swaps*» beschreiben eine Technik, die den Tausch zwischen zwei Kryptowährungen ermöglicht. Dabei befinden sich diese zwei Kryptowährungen auf zwei unterschiedlichen Blockchain-Netzwerken (bspw. Ethereum- und Bitcoin-Blockchain). Dieser Prozess – mittels «*Smart Contracts*» – ermöglicht Parteien den direkten Handel ihrer Assets. Dadurch wird ein Tausch von den digitalen Brieftaschen initiiert und es benötigt keine intermediäre Stelle, welche das «*Matching*» der jeweiligen Kryptopaare durchgeführt [12]. Bei zentralen Börsen wird das «*Matching*» jeweils durch einen Mittelsmann, der Börse selbst, realisiert. «*Atomic Swaps*» (Vgl. 4.2.1.1) werden dazu verwendet um «*peer-to-peer*» Trades über verschiedene Blockchains innert definierter Zeit durchzuführen.

4.2.1.1 Handelsfunktion bei Resistance – Atomic Swaps

Anhand dieser Schrittbeschreibung wird aufgezeigt, wie der «*Atomic Swap*» bei Resistance durchgeführt wird.

1. B_{ob} gibt eine Bestellung auf Resistance ein. Bspw. will er 10 Ethereum-Token gegen einen Bitcoin handeln.
2. A_{lice} sieht das Angebot von B_{ob} und akzeptiert dieses. Um zu beweisen, dass A_{lice} sich für den Kauf verpflichtet, muss A_{lice} eine Börsen-Gebühr von 0.15% des zu sendenden Betrages bezahlen. Dies sind dann 0.015 Bitcoin.
    a. Diese Gebühr ist eine Massnahme, um sicherzustellen, dass B_{ob} nicht mit Anfragen überflutet wird. B_{ob} muss keine Gebühr bezahlen.
    b. Nachdem die Gebühr bezahlt wurde, beginnt der «*Atomic Swap*».
3. B_{ob} Zahlt einen Betrag von 112% (11.2 Ethereum Token) des originalen Betrages (10 Ethereum-Token) zu einer sicheren «*Wallet-Adresse*» ein. Zu dieser Adresse hat niemand Zugriff, solange die Transaktion nicht storniert oder erfolgreich durchgeführt wurde. Abgesehen davon, ob der «*Atomic Swap*» erfolgreich war oder nicht, wird B_{ob} den einbezahlten Betrag wieder zurückerhalten.
4. A_{lice} sendet den Bitcoin an eine andere sichere «*Wallet-Adresse*». Solange der «*Atomic Swap*» nicht durchgeführt wurde, hat niemand Zugriff auf das Guthaben. Im Falle eines Abbruchs des Handels, erhalten beide Parteien ihre digitalen Werte, abzüglich der Gebühren, zurück.

[7] Miner sind Knoten eines Netzwerkes, welche die Korrektheit eines Blockes beweisen müssen. Dies kann das Lösen einer rechenintensiven Mathematikaufgabe sein. Der erste der es löst, wird in Form dieser Kryptowährung belohnt. Die Entlohnung führt zu einem Wachstum des Volumens.

5. B_{ob} sendet 10 Ethereum-Token zu A_{lice} und schliesst seinen Teil der Transaktion ab. A_{lice} ist die einzige Partei, welche die Zahlung verweigern kann.
6. A_{lice} akzeptiert die Zahlung von B_{ob} und gibt ihm die Möglichkeit ihren Bitcoins zu beantragen.
7. B_{ob} akzeptiert den Bitcoin von A_{lice} und beide Parteien haben nun die digitalen Werte erhalten.
8. B_{ob} und A_{lice} erhalten nun die im Vorfeld getätigten Anzahlungen zurück und der «*Atomic Swap*» ist beendet.

4.2.2 ResDEX Desktopapplikation

Für diese Analyse wurde die Software – Resistance Wallet 2.2.5 – für MacOS installiert. Bei der Installation werden der Pfad, Benutzername sowie das Passwort angegeben. Man hat die Möglichkeit ein neues «*Wallet*» zu erstellen oder ein vorhandenes wiederherzustellen. In diesem Versuch wurde ein neues «*Wallet*» erstellt. Das «*Wallet*» wiederum wird ebenfalls mit einem Passwort versehen und mit dem Passwort im zuvor definierten Schritt bestätigt. Es ist wichtig zu erwähnen, dass es sich bis dato (15.01.2020), um einen Prototyp handelt. Die wichtigsten Eigenschaften werden in den folgenden Unterkapiteln erläutert.

4.2.2.1 Buy Bitcoin with VISA

Diese Funktionalität ermöglicht dem Benutzer, durch eine Kreditkarte, den Kauf von Bitcoin oder Ethereum. Resistance bietet somit ein Fiat-Gateway an. Dieses Gateway wird vom Unternehmen «*Simplex Payment Services*» zur Verfügung gestellt. Um den Kauf zu tätigen, wird die Menge der gewünschten Kryptowährung sowie die entsprechende «*Wallet-Adresse*» angegeben. Anschliessend werden die Kreditkarteninformationen wie bspw. die Kreditkartennummer, der Name oder das Ablaufdatum angegeben. Der Kauf mittels Kreditkarte wurde nicht getestet.

4.2.2.2 ResDEX (Beta)

Gemäss Resistance ist diese Sektion die Kernkomponente. Damit der Benutzer die Funktionalitäten nutzen kann, benötigt es aber eine Verifizierung («*Know Your Customer (KYC)*»). Dabei ist es notwendig, dass persönliche Angaben wie Vor- und Nachname, Geburtsdatum und Telefonnummer übermittelt werden. Diese müssen wahrheitsgetreu sein, da man ebenfalls ein offizielles Dokument mitliefern muss, welches die vorher getätigten Angaben bestätigt. Bspw. hat man die Möglichkeit einen Pass oder eine Identitätskarte hochzuladen. Gemäss dem offiziellen Telegram-Chat von Resistance wird die Überprüfung von der externen Firma «*IdentityMind*» durchgeführt.

Nach erfolgreicher Verifizierung kann der Benutzer handeln. Es stehen bis dato (15.01.2020) 58 digitale Werte zur Verfügung. Diese können ausgewählt werden und es wird pro Token eine geheime «*Wallet-Adresse*» generiert. Gemäss Resistance wird dadurch die Privatsphäre und Anonymität der Benutzer garantiert. Bspw. sendet man 10 Bitcoins an diese verschleierte «*Wallet-Adresse*». Diese werden vom «*Resistance-Wallet*» empfangen und im Hintergrund dem Benutzer gutgeschrieben. Die Dokumente für den Versuch wurden am 30.10.2019 eingereicht, jedoch blieb eine Rückmeldung bezüglich des KYC-Prozesses aus. Aus diesem Grund kann im Forschungsbericht lediglich jenes, was im Whitepaper erläutert wird, wiedergeben werden. In der Subsektion «*Assets*» werden dem Benutzer seine Tokens und dessen ungefährer Wert in US-Dollar angezeigt. Ebenfalls sind in dieser Sektion die «*Wallet-Adressen*» der jeweiligen Tokens für Ein- und Auszahlung vorhanden. Die Subsektion «*Buy/Sell*» ermöglicht es dem Benutzer die gewünschten Währungen gegeneinander zu handeln. Man erstellt eine «*Market Order*», wo man eine Kryptowährung gegen eine andere handeln kann. Dies wird durch das «*Matching*» durchgeführt. Zu den anderen Subsektionen («*Advanced Trading und Order*»)

konnten keine Quellen ermittelt werden. In der letzten Subsektion – «*Accounts*» – können «*Wallet-Adressen*» für die entsprechenden Kryptowährungen erstellt werden.

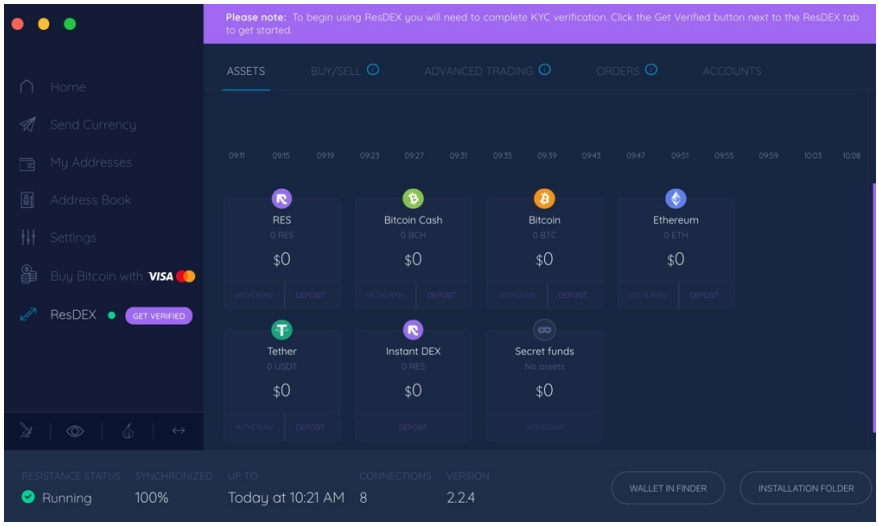

Abbildung 4. Resistance Desktopapplikation – ResDEX

4.2.3 Kritische Betrachtung ResDEX

Es sind zwei Punkte vorhanden, welche in Anbetracht der Privatsphäre kritisch zu beurteilen sind. Der erste Punkt ist das Fiat-Gateway, welches den Kauf von Bitcoin und Ethereum durch eine Kreditkarte ermöglicht. Beim Kauf von einer der beiden Kryptowährungen müssen die Kreditkarteninformationen des Benutzers oder irgendeines Benutzers angegeben werden. Da jede Kreditkarte in der Regel genau einem Besitzer zugeordnet ist, kann anhand dieses Kaufes die Transaktion einem Benutzer zugeordnet werden. Ebenfalls birgt diese Funktionalität ein Risiko, weil es für den Kauf mit der Kreditkarte keine KYC-Überprüfung des Benutzers benötigt. Ein Krimineller kann sich dadurch einen Vorteil verschaffen, indem er sich geklaute Kreditkarteninformationen besorgt. Mit diesen könnte er am Anfang des Monates Kryptowährungen auf Kosten anderer kaufen. Im ungünstigsten Fall merken die Betroffenen dies erst am Ende des Monates, wenn die Abrechnung der Kreditkarte ankommt.

Der zweite Punkt, welcher kritisch betrachtet wird, ist die der Handelsplatz (ResDEX). Im Whitepaper steht, dass man mit dieser Kernkomponente ohne jegliche Registrierung handeln kann. Dies ist nicht der Fall, denn ein Benutzer ist vollkommen eingeschränkt, solange er nicht den KYC-Prozess durchläuft. Dieses Vorgehen wurde in Kapitel 4.2.2.2 erläutert. Gemäss Angaben vom offiziellen Telegram-Chat, ist dies aus regulatorischen Gründen notwendig. Eine externe Firma übernimmt nimmt die Aufgabe der Verifizierung von Benutzern. Unter dieser Tatsache, dass Dokumente, welche die Eindeutigkeit einer Person beweisen, wird möglicherweise die Anonymität sowie die Privatsphäre des Benutzers gefährdet. Resistance unterhält ebenfalls einen eigenen Token (RES), welcher dazu verwendet wird, um einerseits die Anonymität zu erhöhen und anderseits, um damit einen Gewinn zu erzielen.

4.3 Auswertung von Bisq – ResDEX – diva.exchange

In der folgenden Tabelle werden die Eigenschaft von Bisq, Resistance und diva.exchange zusammengefasst. In der Spalte diva.exchange wird unterschieden zwischen bereits erreichter Eigenschaft (nicht kursive Schrift) und gewünschter Eigenschaft (kursive Schrift).

Eigenschaft	**Bisq**	**ResDEX**	**diva.exchange**
Plattform	Plattformunabhängig	Plattformunabhängig	Plattformunabhängig
Registrierung	Nein	Nein	Nein
Netzwerktechnologie	Tor	Tor	I2P
Order-Matching	Nein	Ja	Ja
KYC	Nein	Ja	Nein
Fiat-Gateway	Nein	Ja	Nein
Bankinformationen	Ja	Ja	Nein
Sicherheitsleistung	Ja	Ja	*Ja/Nein*
Eigene Gebühren	Ja	Ja	Nein
Eigener Token	Ja	Ja	Nein
Anonymität	Gebrochen	Gebrochen	Nicht gebrochen
Quelloffen	Ja	Ja	Ja

Tabelle 4. Vergleich Bisq, Resistance und diva.exchange.

Durch den Vergleich in der Tabelle 4 ist ersichtlich, wo die Gemeinsamkeiten und Unterschiede liegen. Nachfolgend werden die wichtigsten Punkte erläutert.

Registrierung
Es ist bei keiner der Applikationen eine Registrierung notwendig. Bei Resistance hingegen, werden die Handelsfunktionen erst dann freigeschaltet, wenn sich der Benutzer registriert, die Dokumente übermittelt und die Überprüfung seiner Dokumente erfolgreich war. Dies wurde in Kapitel 4.2.2.2 beschrieben.

Netzwerktechnologie
Bei der Netzwerktechnologie setzen Bisq und Resistance auf die Tor-Technologie. Bei Resistance ist diese Funktion standardmässig deaktiviert. Die I2P-Technologie, welche denselben Nutzen wie Tor generiert, wird vom diva.exchange verwendet.

Verifizierung der Benutzer
Bei Resistance müssen die Benutzer persönliche Informationen und Dokumente einreichen, damit sie die Handelfunktionen nutzen können. Bei Bisq und diva.exchange ist das nicht der Fall.

Fiat-Gateway
Resistance bietet den Benutzern die Möglichkeit, Bitcoin oder Ethereum mit einer Kreditkarte zu kaufen. Diese Funktionalität kann jeder Benutzer auch ohne Verifizierung nutzen. Bisq bietet keine integrierten Schnittstellen zu Finanzinstitutionen an. Bisq interagiert mit den Benutzern und leitet sie durch den Handel. Bei diva.exchange werden keine Schnittstellen zu Finanzinstitutionen integriert.

Anonymität und Privatsphäre
Bei Bisq (vgl. Kapitel 4.1.2) sowie Resistance (vgl. Kapitel 4.2.3) wurde aufgezeigt, dass die Anonymität und Privatsphäre gebrochen wird und Schlüsse auf die Identität der Benutzer gezogen werden können.

Der Unterschied zum jetzigen Projekt zeigt auf, dass ähnliche Ansätze vorhanden sind, jedoch sich diese in den wichtigsten Punkten zum jetzigen Projekt unterscheiden. Die Anonymität wird bei Bisq wegen dem Handel mit Fiat gebrochen, weil dadurch beiden Gegenparteien die Identität des anderen offengelegt wird. Bei Resistance hingegen wird eine Fiat-Schnittstelle durch eine Kreditkarte angeboten, welche ohne eine Verifizierung genutzt werden kann. Um aber nur mit Kryptowährungen zu handeln, benötigt es eine KYC-Überprüfung und dies ist ein klarer Bruch gegen die Privatsphäre. Weiter haben Bisq und Resistance eine eigene Kryptowährung, welche als Businessmodell verwendet wird. Bei diva.exchange wird dies nicht der Fall sein. Einen weiteren wesentlichen Unterschied bilden die handelbaren Paare. Bisq bietet nur den Handel gegen Bitcoin an. Bei diva.exchange werden die handelbaren Kryptopaare gegen keine Kryptowährung ausgerichtet sein.

5 Literaturrecherche

In den folgenden Unterkapiteln werden die Technologien I2P und Iroha, welche bei diva.exchange zum Einsatz kommen, erläutert.

5.1 I2P

I2P ist ein nachrichtenorientiertes, anonymes und «*peer-to-peer*» basiertes Kommunikationsnetzwerk. Der Hauptgrund für die Entwicklung dieses von I2P war die Gewährleistung einer vollständig anonymen Kommunikation zweier Parteien innerhalb eines Netzwerkes [13]. I2P wurde das erste Mal 2003 vorgeschlagen und hat seine Wurzeln im «*Invisible Internet Project*» [14]. Durch Erweiterungen mit Bibliotheken (bspw. «*I2P Streaming Library*») werden Datenübertragung sowie Streaming ebenfalls unterstützt [15]. Innerhalb des I2P-Netzwerkes werden einige Anwendungen angeboten, welche bspw. anonymes Webhosting oder «*Filesharing*» anbieten. I2P wurde so konzipiert, dass die Dienste innerhalb des Netzwerkes angeboten werden. Die Verwendung von Diensten, welche sich ausserhalb des I2P-Netzwerks befinden, benötigen einen Out-Proxy [13]. Sämtliche Datenübertragungen sind mehrfach verschlüsselt und werden zusätzlich durch die ständig wechselnden Routen (Tunnel) und Netzwerkteilnehmer (Peers) geschützt.

I2P ist ein sogenanntes «*Overlay-Netzwerk*», mit dem Benutzer anonym im Netzwerk interagieren können. Technisch gesehen handelt es sich dabei um ein Multiapplikations-Java-Framework, welches zur Bereitstellung anonymer «*peer-to-peer*» Netzwerke entwickelt wurde [16]. Jeder Benutzer betreibt einen sogenannten I2P-Router, welcher den Kern der I2P-Software bildet. Alle Nachrichten werden über Tunnel weitergeleitet, die von jedem I2P-Router mithilfe anderer I2P-Peers erstellt werden. Tunnel können nur in eine Richtung benutzt werden. Daher müssen Tunnel für ausgehenden und eingehenden (Inbound- und Outbound-Tunnel) Verkehr erstellt werden. Die Auswahl der Peers erfolgt über einen tierbasierten Peer-Auswahlalgorithmus («*tier-based-peer-selection*»), der auf jedem I2P-Router ausgeführt wird. Nach dem Einrichten von eingehenden und ausgehenden Tunneln, können Clients ihre Kontaktinformationen in einer globalen Datenbank namens «*netDB*» veröffentlichen. Die «*netDB*» enthält Kontaktinformationen für jeden I2P-Peer und jeden öffentlich ausgeführten Dienst innerhalb des I2P-Netzwerks. Nachrichten, die über das I2P-Netzwerk gesendet werden, wenden die Knoblauchverschlüsselung an («*garlic encryption*»).

5.1.1 I2P Router

Ein I2P-Router wird ebenfalls als Client oder Knoten («*Node*») bezeichnet. Der Aufbau des I2P-Netzwerks besteht aus Peers, welche die I2P-Software installiert haben und somit einen I2P-Router für das Netzwerk bereitstellen [13]. Die Aufgaben eines Routers besteht darin, eine Peer-Statistik zu unterhalten. Diese Statistik wird für die Peer-Auswahl, kryptographische Operationen, den Tunnelaufbau, das Anbieten von Services und für die Weiterleitung von Nachrichten verwendet. Die Anwendungen und die damit verbundenen Services hängen stark von den Tunneln ab, welche von den I2P-Routern erzeugt werden, weil dadurch die Anonymität beeinflusst wird [16]. Somit sind konstante Tunnelverbindungen als Netzwerkgerüst von grosser Bedeutung. Die Tunnelerstellung wird vom Router durch den «*tier-based-peer-selection*» Algorithmus durchgeführt.

5.1.2 Tunnel

Alle Nachrichten im I2P-Netzwerk werden durch sogenannte Tunnel übertragen. Ein Tunnel ist eine unidirektionale verschlüsselte virtuelle Verbindung, die Nachrichten nur in eine Richtung versendet [17]. Für den Tunnelaufbau werden in der Regel sechs I2P-Peers verwendet. Die Abbildung 5 zeigt einen Tunnelaufbau. Dieser wird durch sechs Peers initialisiert. Der «*Outbound-Tunnel*» besteht in diesem Beispiel aus drei Teilnehmern. Das «*Outbound-Gateway*» ist der Initiator der Tunnelverbindung und für dessen Kreierung zuständig. «*Outbound-Participant*» ist ein Peer, welcher für die Weiterleitung der Nachricht zuständig ist. Somit kann es *1..n* «*Outbound-Participants*» geben. Der «*Outbound-Endpoint*» ist der letzte Teilnehmer des «*Outbound-Tunnels*». Er ist dafür zuständig, dass die Nachricht an das korrekte «*Inbound-Gateway*» des «*Inbound-Tunnels*» der Destination («*Inbound-Endpoint*») weitergeleitet wird. Die Länge eines Tunnels ist ein Kompromiss zwischen der Leistung und der Anonymität. Längere Tunnel erhöhen die Anonymität, wobei dadurch die Leistung vermindert wird. Umgekehrt wird die Anonymität geschwächt und die Leistung erhöht [16] [18].

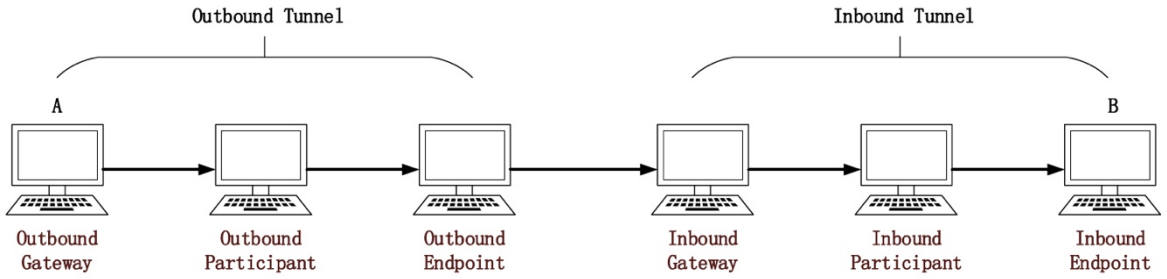

Abbildung 5. I2P-Tunnel [15].

Man unterscheidet in einem I2P-Netzwerk zwischen «*Exploratory-Tunnel*» und «*Client-Tunnel*» [14] [18].

Exploratory-Tunnel:
Diese Tunnelart hat eine geringe Bandbreite und wird für nicht private und sensitive Operationen verwendet. Ein I2P-Router verwendet diese Art von Tunneln, um Super-Peers zu kontaktieren und Abfragen an die «*NetDB*» zu senden. Diese Tunnelart wird auch dazu verwendet, um neue Tunnel zu erstellen, verwalten und um andere Tunnel zu zerstören [18].

Client-Tunnel:
Werden verwendet, um Applikationsnachrichten weiterzuleiten. Aus diesem Grund haben sie im Vergleich zum «*Exploratory-Tunnel*» eine höhere Bandbreite.

Tunnel haben eine Lebensdauer von zehn Minuten [17]. Im Hintergrund wird jeweils ein neuer Tunnel aufgebaut, damit man von der Zerstörung des aktuellen Tunnels nichts bemerkt. Die Lebensdauer ist eine präventive Massnahme für die Sicherstellung der Anonymität, damit der Netzwerkverkehr durch bspw. Mustererkennungsalgorithmen nicht oder erschwert analysiert werden kann [13].

5.1.3 Stufen-Basierte-Peer-Selektion (*tier-based-peer-selection*)

Dies beschreibt den Prozess für die Auswahl der Peers, welche für den Tunnelaufbau verwendet werden können. Diese Selektion der Peers ist abhängig von der Rangzuordnung («*tier*») der Peers («*peer-profiling*»).

5.1.4 Peer-Profile (*peer-profiling*)

Das Verfahren von «*peer-profiling*» wird dazu verwendet, um die I2P-Peers verschiedenen Rängen zuzuordnen. Peers, welche sich Leistungsmerkmale teilen, werden in denselben Rang eingestuft [18]. Jeder I2P-Router führt dieses Verfahren für andere I2P-Router durch. Damit wird sichergestellt, dass ein I2P-Router die eigenen Leistungsmerkmale nicht verfälscht und sich damit in gewünschte Ränge einordnen lässt. Jeder einzelne I2P-Router unterhält eine Datenbank mit diesen Statistiken. Die Datenbank wird «*profiles*» genannt. Die verschiedenen Ränge gliedern sich wie folgt:

Rang 1: Fällt nicht aus.
Rang 2: Gut integriert – Peers, die vermutlich viele andere Peers kennen.
Rang 3: Hohe Kapazität – Peers, die dafür bekannt sind Tunnelanfragen zu akzeptieren und zu erstellen.
Rang 4: Schnell – Das sind Peers von Rang 3, welche eine hohe Bandbreite aufweisen.

Um einen Client-Tunnel zu erstellen werden Peers mit dem Rang vier verwendet. Falls nicht genügend Peers mit dem Rang vier vorhanden sind, werden Peers in absteigender Reihenfolge (drei bis eins) verwendet. Für die Erstellung von «*Exploratory-Tunneln*» werden Peers mit dem Rang drei verwendet. Die Peers mit den Rängen eins und zwei sind sogenannte Backupoptionen, falls Peers mit den Rängen drei und vier nicht verfügbar sind.

5.1.5 Netzwerk Datenbank («*NetDB*»)

Die «*NetDB*» basiert auf einer verteilten Hash-Tabelle und enthält alle verfügbaren Informationen über das I2P-Netzwerk. Dies sind Informationen über die Peers des Netzwerkes («*routerInfo*») und Informationen über die angebotenen Services der jeweiligen Destinationen («*leaseSet*»). Für die Erstellung sowie den Unterhalt der «*NetDB*» werden Super-Peers, auch «*floodfill-peers*» genannt, benötigt. Jeder Super-Peer hat die Zuständigkeit über einen Teil des Netzwerkes. Durch die «*Kademlia XOR*» Distanzmetrik [19] und die Peer-ID [16] wird die Zuständigkeit des Netzwerkes für einen Super-Peer evaluiert.

«routerInfo»
Charakterisiert einen I2P-Peer und somit werden durch die «*routerInfo*» alle I2P-Peers identifiziert und können dadurch kontaktiert werden [20]. Das beinhaltet Informationen wie die IP-Adresse, den Port, die Peer-ID, die I2P-Stable-Versionsnummer, die Netzwerkversion, die Transport-Fähigkeit, einen öffentlichen Schlüssel und einen 256-Bit Hash-Identifier. Eine Liste von «*routerInfo*» kann von einem «*well-known*» Webserver erhalten werden. Das Erhalten der initialen «*routerInfo*» Liste wird als «*reseeding*» bezeichnet [16].

«leaseSet»
Ein «*leaseSet*» wird dazu verwendet, um Informationen zu speichern, wie ein interner I2P-Service erreicht werden kann (Destination). Ein «*leaseSet*» spezifiziert Zugangsknoten (Peers), welche als «*leases*» bezeichnet werden. Ein «*lease*» identifiziert einen Peer, welcher ein «*Inbound-Gateway»* eines «*Inbound-Tunnels*» und den damit verbundenen Services («*Inbound-Endpoint*») zur Verfügung stellt [16].

Um Informationen über «*routerInfo*» sowie «*leaseSet*» zu erhalten, kann der nächste «*Super-Peer*» kontaktiert werden. Im Falle des Speicherns sendet der Super-Peer im I2P-Netzwerk die «*routerInfo*» oder das «*leaseSet*» an die nächsten sieben «*Super-Peers*». Bei einer Anfrage werden zwei naheliegenden «*Super-Peers*» kontaktiert. Wenn keine Informationen vorhanden sind, geben die zwei «*Super-Peers*» eine Liste anderer «*Super-Peers*» zurück. Dieser Vorgang wird in einer Schleife solange weitergeführt, bis die Information erhalten wurde oder alle «*Super-Peers*» angefragt wurden und keine Information liefern konnten [16].

5.1.6 Garlic-Routing

Das Senden oder Empfangen von Nachrichten im I2P-Netzwerk benötigt mindestens eine Initialisierung von einem «*Inbound-*» sowie «*Outbound-Tunnel*». Damit ein Client von einem Service im I2P-Netzwerk Gebrauch machen kann, benötigt er vom «*Super-Peer*» die Destination [14]. Die Destination beinhaltet ein Set von «*Inbound-Gateways*» des «*Inbound-Endpoints*» (vgl. Abbildung 5).

«Garlic-Routing» verwendet «*Garlic-Messages*» (Nachrichten), welche mehrere «*Garlic-Cloves*» (Datennachrichten) enthalten können. «*Garlic-Cloves*» können unterschiedliche Destinationen haben und sind mit dem öffentlichen Schlüssel des Empfängers end-zu-end verschlüsselt. Die «*Garlic-Message*» selbst wird mit dem von den Tunnel-Peers ausgehandelten symmetrischen Schlüssel mehrfach verschlüsselt [14]. Beim Durchlaufen des Tunnels entfernt jeder I2P-Peer des Tunnels eine Verschlüsselungsebene, bis die «*Garlic-Message*» den «*Outbound-Endpoint*» des Tunnels erreicht. Der «*Outbound-Endpoint*» leitet dann jede «*Garlic-Clove*» an das «*Inbound-Gateway*» seines Ziels weiter, welches dann die «*Garlic-Message*» an den tatsächlichen Empfänger weiterleitet (der Empfänger erhält alle «*Garlic-Cloves*» in einer «*Garlic-Message*». Dabei wird von den am Tunnel beteiligten Peers eine Verschlüsselungsebene hinzufügt (ausgehandelter symmetrischer Schlüssel). Nur dem Empfänger ist es möglich alle Verschlüsselungsebenen der «*Garlic-Message*» sowie die-end-zu Verschlüsselung der «*Garlic-Cloves*» zu entschlüsseln [14].

5.2 Vergleich I2P und TOR

Beim Vergleich der beiden Technologien stützt sich der Autor auf den wissenschaftlichen Bericht «*A Survey on Tor and I2P*» von den Autoren Bernd Conrad und Fatemeh Shirazi [21].

I2P	**TOR**
I2P verwendet I2P-Router, um ein Netzwerk aufzubauen. Dabei fungieren sämtliche Router (wenn dies nicht deaktiviert wurde) als Netzwerkknoten, um einen Tunnel aufzubauen, damit Nachrichten transferiert werden können.	Tor verwendet sogenannte Onion-Router (OR), welche von freiwilligen zur Verfügung gestellt werden.
I2P wurde so konzipiert, um Services innerhalb des I2P-Netzwerkes zu verwenden. Um auf externe Services zuzugreifen, kann ein Out-Proxy konfiguriert werden.	Tor wurde so konzipiert, dass es einige Exit-Router hat, um auf externe Services zuzugreifen.
I2P ist eine Middleware (Message oriented Middleware – MoM), welche Applikationen ein Interface bietet, um innerhalb des Netzwerkes kommunizieren zu können. Für Kommunikation ausserhalb des Netzwerks werden Socks angeboten.	TOR verwendet das Socket Secure (Socks) Interface. Somit sind Applikationen, die nach Socks ausgelegt sind, vermutlich kompatibler mit der TOR-Software.
Unterstützt TCP und UDP. Theoretisch sollte I2P somit eine bessere Performanz aufweisen.	Socks Interface kann Nachrichten nur über TCP transferieren.
Verwendet TLS und zusätzliche Tunnelverschlüsselung	Verwendet TLS
Garlic Verschlüsselung	Onion Verschlüsselung
Innerhalb des Netzwerkes immer verschlüsselt.	Innerhalb des Netzwerkes immer verschlüsselt.
Nachrichten innerhalb des I2P-Netzwerkes sind end-zu-end verschlüsselt.	End-zu-end Verschlüsselung kann nicht garantiert werden, weil es abhängig vom Transport Layer Protokoll ist. Unsichere Protokolle sollten somit vermieden werden.
Der erste Peer, der die Nachricht erhält, weiss nicht ob diese vom Sender oder von einem anderen Peer kommt.	Nur der erste Onion-Router kennt die IP-Adresse des Senders. Alle anderen Onion-Router kennen nur die IP-Adresse des Vor -und Nachfolgers. Somit kennt nur der letzte Onion-Router die IP-Adresse des Empfängers. Diese Tatsache kann ein potenzielles Risiko sein. Korrupte Exit Nodes könnten somit die in Klartext übertragene Nachricht missbrauchen. Somit ist die Anonymität bei TOR stark abhängig vom «*node selecting*» Algorithmus und wie dieser vertrauenswürdige «*entry-guards*» auswählt.
Durch die steigende Anzahl an Peers wird vermutlich die Anzahl von Rang drei und vier Tiers erhöht. Das würde zu höheren Bandbreiten und tieferen Latenz führen.	Die steigende Anzahl an Clients bedeutet auch, dass der Netzwerkverkehr erhöht wird, was bei TOR zu einer Netzwerküberlastung führen könnte, weil es eine gewisse Anzahl

Die steigende Anzahl an Teilnehmern erhöht den Netzwerkverkehr, was vermutlich die Anonymität der Peers steigert. Die Anzahl an internen Services wird durch mehrere Peers erhöht.	von Onion-Servern gibt, welche von freiwilligen betrieben werden. Somit müsste die Anzahl an Servern erweitert werden.
Eine Zentralisierung gibt es hier nicht. Jeder Peer unterhält lokal eine Liste mit den für ihn bekannten Hosts.	Infos zu Relay-Nodes und den versteckten Services werden derzeit von autorisierten Verzeichnissen aus der US/EU bereitgestellt. Diese nehmen Änderungen im Netzwerk wahr und teilen diese Informationen. Wenn diese Verzeichnisse kollidieren ist somit die Anonymität gefährdet.
Jeder Peer in einem Tunnel kann der Sender, Tunnelteilnehmer oder der Empfänger sein.	TOR unterscheidet zwischen Eingangs- (Entry), Ausgangs- (Exit) und Intermediären Knoten.
I2P-Clients verlassen sich einzig und alleine auf die vorher aufgezeichneten Werte und auf den jetzigen Stand des Netzes. Der «*tier-based-peer-selection*» Algorithmus reagiert schnell falls gewisse Nodes ausfallen oder sich die Netzwerktopologie ändert. Das schnelle Ändern birgt auch ein Risikopotenzial. Peers der Ränge drei und vier können angegriffen werden (DDOS-Attacke) und somit ausfallen. Angreifer könnten versuchen ihre bösartigen Router zu injizieren damit diese in die Ränge drei und vier gelangen.	TOR's Verzeichnis Server untersuchen aktiv die Bandbreiten von den Onion-Routern, damit sie die Intermediären- sowie Ausgangsknoten auswählen können. Dies generiert «*non-data-message-traffic*». Falls über einen OR keine Daten bezüglich Bandbreite etc. vorhanden sind, muss sich TOR auf die vom OR zur Verfügung gestellten Daten verlassen. Dies kann zu einer Fehlklassifikation führen oder ein Angreifer kann seine Performanz-Informationen abändern, um als «*entry-guard*» ausgewählt zu werden.

Tabelle 5. Vergleich I2P und TOR.

5.2.1.1 Fazit – I2P – Tor

Zwischen den beiden Netzwerktechnologien sind einige Unterschiede vorhanden. Tor verlässt sich dabei auf Server, welche von freiwilligen [22] zur Verfügung gestellt werden, um Verbindungen herzustellen («*circuit*»). Bei I2P hingegen bilden die Peers, welche die Software installieren, das Netzwerk. Für die Kommunikation werden Tunnel aus verschiedenen Peers aufgebaut. Tor weist somit zentralisierte Aspekte auf. I2P ist vollkommen dezentral ausgerichtet. Aufgrund der zentralisierten Netzwerkarchitektur von Tor ist es vermutlich relativ einfach alle öffentlichen Server ausfindig zu machen und zu blockieren [21]. Diese Tatsache wiederspricht dem jetzigen.

Ein weiterer Unterschied zu Tor liegt darin, dass Tor für den Datenverkehr ins Internet konzipiert wurde. I2P hingegen wurde kreiert, um den Verkehr innerhalb des eigenen Netzwerkes zu regeln, was von Tor als versteckten Service bezeichnet wird. Das eigene Netzwerk besteht aus allen I2P-Routern, welche von den Teilnehmern betrieben werden.

Was die Sicherheit anbelangt ist I2P vermutlich robuster. Tor sowie I2P bieten eine end-zu-end Verschlüsselung innerhalb des jeweiligen Netzwerkes an. Bei I2P hat es den Vorteil, dass es ein vollkommen dezentrales Netzwerk ist, während Tor einen zentralen Directory Server zur Verfügung stellt und diesen pflegen muss. Ebenfalls weiss der Empfänger von einem Tunnel nicht, ob die Nachricht vom Sender oder von einem anderen Tunnel Peer gesendet wurde.

Attacken wie bspw. «*Timing-Angriffe*» können die Nutzer des Tornetzwerkes entlarven [22]. Der Mechanismus des I2P-Netzwerkes schützt sich vor solchen Angriffen, indem die Datenpakete in unidirektionalen Tunneln versendet werden. Somit werden Nachrichten durch unterschiedliche und immer wechselnde Tunnelrouten versendet oder empfangen [16]. Dies erschwert es den Angreifern eine Korrelation zwischen ein- und ausgehenden Daten herzustellen. Dabei müssen Daten mehrere Nodes, abhängig von der Konfiguration, traversieren. Dies steigert vermutlich die Wahrscheinlichkeit, dass einer davon einem Angreifer gehören könnte. Dies führt dazu, dass ein Angriff über Korrelation auch hier nicht ganz ausgeschlossen werden kann. Das «*Garlic-Routing*» hingegen erschwert diesen Angriff.

Es kann festgehalten werden, dass auf Grund der Merkmale von I2P, diese Technologie die Bedingungen dieses Projektes erfüllt. Wenn angenommen beide Netzwerke anfangen zu wachsen, wird Tor sich vermutlich vor Probleme stellen. Die betriebenen Server, welche die «*circuits*» aufbauen, könnten ausgelastet werden und somit würde die Bandbreite des Netzwerkes sinken. Bei I2P hätte dies vermutlich eine positive Auswirkung, da sämtliche Peers das Netzwerk aufbauen. Somit ist vorstellbar, dass die Bandbreite des Netzwerkes steigen würde. Ein Weiter Ansatz, der dem jetzigen Vorhaben wiederspricht, ist der Aspekt der Zentralisierung bei Tor. Dieses Projekt verfolgt das Ziel, dass sämtliche Daten und Prozesse des Prototyps vollständig dezentralisiert sin.

5.3 Hyperledger Iroha

Dieses Kapitel stützt sich vollumfänglich auf die offizielle Dokumentation von Hyperledger Iroha [23]. Falls andere Quellen verwendet werden, wird dies angegeben. Da es sich bei Hyperledger Iroha um ein neues Produkt handelt, sind Quellen noch rar. Die Zusammenfassung und Erläuterung dieses Kapitels werden dem Auftraggeber und potenziellen neuen Parteien als Informationsquelle dienen.

Hyperledger Iroha ist eine in C++ entwickelte private Blockchain, welche als Open-Source Projekt klassifiziert ist. Das Projekt ist das vierte in diesem Bereich und wird von der Linux Foundation betrieben. Iroha ermöglicht das Persistieren von sämtlichen Transaktionen und derer Änderungen auf der Blockchain. Das Design und die Konzeptionierung von Hyperledger-Iroha ermöglichen eine einfache Integration in neue oder bestehende Anwendungen.

5.3.1 Unterschied zu anderen Blockchains

Die Blockchains von Bitcoin sowie Ethereum sind öffentlich und dementsprechend kann jeder teilnehmen und alle Daten einsehen. Beide Blockchain-Technologien haben native Kryptowährungen, welche dazu verwendet werden, um mit dem System zu interagieren. Bei Hyperledger Iroha gibt es keine nativen Kryptowährungen. Stattdessen, um dem Nachtrag von Unternehmen gerecht zu werden, wird die Systeminteraktion nur bestimmten Personen (Rollen) zugeteilt, welche untereinander intragieren können. Das Abfragen der Daten von der Blockchain kann je nach Konfiguration variieren. Es ist möglich, dass jedermann die Daten einsehen kann oder nur die Teilnehmer («*permissioned*» oder «*permissionless*»)

Ein wesentlicher Unterschied zu Ethereum besteht darin, dass Benutzern von Iroha allgemeine Funktionen zur Verfügung gestellt werden. Dies ist bspw. das Übertragen von digitalen Werten oder die Erstellung eines Accounts. Auf diese Weise müssen keine «*Smart Contracts*» programmiert und integriert werden.

In den nachfolgenden Kapiteln werden die verschiedenen Komponenten von Hyperledger Iroha erläutert.

5.3.2 Account

Dies ist eine Iroha-Entität, welche ein spezifiziertes Set von Aktionen durchführen kann. Jeder Account gehört zu einer von *N* existierenden Domänen. Ein Account kann *0* bis *N* Rollen enthalten. Bei Iroha wird zwischen zwei Kategorien von Berechtigungen unterschieden. Dabei handelt es sich um rollenbasierte und erteilbare («*grantable*») Berechtigungen. Rollen sind ein Set von Berechtigungen, wobei nur «*grantable*» Berechtigungen einem Account direkt zugewiesen werden können.

5.3.3 Asset

Ein Asset ist jede Ware oder jeder Wert der zählbar ist. Jedes Asset ist mit einer vorhandenen Domäne verknüpft. Ein Asset kann eine beliebige Art von Einheiten darstellen. Bspw. eine Währungseinheit, einen Goldbarren oder eine Immobilieneinheit. Somit ist ein Asset eine abstrakte Darstellung einer realen Werteinheit. Bei diva.exchange werden das vermutlich die Kryptowährungen Bitcoin, Monero, Ethereum und Ripple sein.

5.3.4 Block

Ein Block weist eine Container-Datenstruktur auf, welcher aus einer oder mehreren Transaktionen besteht. Die Transaktionsdaten sind unveränderlich und werden dauerhaft in einem Block gespeichert. Die Anordnung der Blöcke und der sich darin befindenden Transaktionsdaten, geschieht linear über die Zeit. Blöcke werden mit einer kryptographischen Signatur von den Iroha-Peers signiert, welche für diesen Block während des Konsensmechanismus abstimmen. Der signierte Inhalt wird als Payload bezeichnet. Die Struktur eines Blocks lässt sich wie folgt unterteilen.

Ausserhalb vom Payload:
Dabei handelt es sich um Signaturen von Peers, welche für die Blockabstimmung während des Konsensmechanismus verwendet wurden. Ebenfalls beinhaltet der äussere Payload den Hash des vorherigen Blockes.

Innerhalb vom Payload:
- *«Height»:* Anzahl der Blöcke in einer Kette (Startblock hat die Höhe 1).
- *«Timestamp»:* Unix Zeit in Millisekunden, welche für die Erstellung durch einen Peer verwendet wurde.
- *«Array of Transactions»:* Erfolgreich durchgeführte Validierungs- sowie Konsensschritte.
- *«Hash of previous block»:* Ist der Hash des vorherigen Blockes.
- *«Rejected transaction hashes (Optional)»:* Array von Transaktions-Hashs, welche den Validierungsschritt nicht bestanden haben.

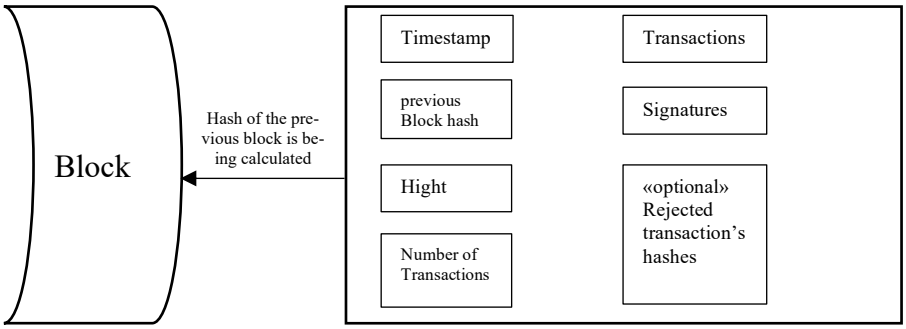

Abbildung 6. Blockaufbau Iroha.

5.3.5 Client

Jede Applikation, die Hyperledger Iroha verwendet, wird als Client bezeichnet. Ein distinktives Feature von Iroha ist, dass alle Clients im Peer-Netzwerk eine Abstraktion von der Client-Server Architektur nutzen, um untereinander zu interagieren. Bspw. müssen Bitcoin-Clients Blöcke validieren («*mining*») oder bei Hyperledger Fabric müssen mehrere Peers abgefragt werden, um sicherzustellen, dass eine Transaktion in einen Block geschrieben wurde. Bei Iroha hingegen interagiert der Client mit beliebigen Peers, ähnlich wie bei einem einzigen Server.

5.3.6 Befehl und Abfrage

Die Manipulation von Accounts und Assets wird durch eine Reihe von Befehlen («*command*») und Abfragen («*queries*») unterstützt. Befehle sind Aktionen, die den atomaren Status ändern und in eine einzelne Transaktion eingefügt werden können, während Abfragen es Iroha-Clients ermöglichen, einen Schnappschuss («*snapshot*») des Peer-Status («*World State View*») zu erstellen. Abfragen ändern somit nicht den Status des Netzwerkes.

5.3.7 Konsensmechanismus

Ein Konsensmechanismus ist in der Informatik ein Prozess, welcher dazu verwendet wird, um eine Einigung über einen einzelnen Wert in mehreren verteilten Prozessen oder Systemen zu erhalten. Konsensalgorithmen wurden entwickelt, um Zuverlässigkeit in einem Netzwerk mit mehreren unzuverlässigen Knoten zu erhalten. Es wird zwischen zwei verschiedenen Arten von Konsens unterschieden [24].

Ein *Konsensalgorithmus* besteht aus einer Reihe von Schritten, mit denen eine Einigung über einen einzelnen Datenwert zwischen verteilten Prozessen oder Systemen erzielt wird. Konsensalgorithmen wurden entwickelt, um Zuverlässigkeit in einem Netzwerk mit mehreren unzuverlässigen Knoten zu erreichen. Das Problem der Erzielung einer Übereinstimmung zwischen mehreren unzuverlässigen Knoten ist als Konsensproblem bekannt und ist in verteilten Rechen- und Multiagentensystemen wichtig. Durch die Verwendung eines Konsensalgorithmus, als grundlegender Bestandteil eines Systems, wird die Zuverlässigkeit erhöht.

Konsens als Komponente erhält einen konsistenten Zustand unter den Peers innerhalb eines Peer-Netzwerks. Iroha verwendet seinen eigenen Konsensalgorithmus – «*yet another consensus (YAC)*». Kennzeichnende Merkmale dieses Algorithmus sind Skalierbarkeit, Leistung und Byzantinische Fehlertoleranz [25]. Wenn Blöcke fehlen, werden sie vom «*Synchronizer*» von einem anderen Peer heruntergeladen. Festgeschriebene Blöcke werden im «*Ametsuchi*» Blockspeicher persistiert.

5.3.7.1 Ablauf vom Konsensalgorithmus - YAC

Der Konsensmechanismus dient für einen konsistenten Zustand unter den Clients innerhalb eines Peer-Netzwerks. Iroha verwendet einen eigenen Konsensalgorithmus namens «*yet-another-consensus – YAC*» [25]. Die Funktionsweise vom Konsensalgorithmus wird anhand eines Beispiels aufgezeigt.

In diesem Beispiel sind vier Peers (A_{lice}, B_{ob}, C_{lara} & D_{anilo}) vorhanden, welche die Erreichung der Einigung anstreben. Jeder Peer sendet seine Transaktion zum eigenen «*Ordering Service*». Es ist die Aufgabe des «*Ordering Services*» die Transaktionen zu sammeln, diese zu sortieren und ein «*Block Proposoal*» P_1 zu erstellen. Der «*Ordering Service*» teilt dann P_1 mit allen anderen Peers im Netzwerk.

Alice fungiert als Prüfstelle welche P_1 validiert und versucht jede Transaktion vom «*Proposal*» auf ihren lokalen Stand zu übertragen. Eine Transaktion wird als valide erachtet, wenn sie die Validierungsregeln nicht verletzt oder nicht gegen den globalen Zustand verstösst (bspw. kann ein Account nicht eine negative Balance aufweisen).

Anschliessend erstellt *Alice* aus allen validen Transaktionen einen Block und berechnet vom Block den Hash H_1. *Alice* kennt nun den «*Proposal Hash*» H_1 und die initiale Reihenfolge der Peers – [A_{lice}, B_{ob}, C_{lara}, D_{anilo}]. Als Parameter für die «*Ordering Funktion*» wird nun der Hash H_1 verwendet. Die «*Ordering Funktion*» errechnet nun eine Permutation als Ergebnis für die

jetzige Rund – [C_{lara}, D_{anilo}, A_{lice}, B_{ob}]. Der erste Peer in der Liste dieser Permutationsrunde ist *Clara*. *Alice* erstellt ein «*vote*» und sendet dieses an *Clara*. Nach dem Teilen des «*votes*» wechselt *Alice* ihren lokalen Zustand zu «*waiting for a commit message*» bis zu einer gewissen Zeitspanne. Wenn von *Clara* nach einer gewissen Zeit keine «*Commit Message*» kommt, sendet *Alice* den «*vote*» an *Danilo*.

Clara erhält den «*vote*» von *Alice*. Angenommen, dass *Clara* im Validierungsprozess denselben Hash H_1 vom Proposal P_1 errechnet, dann gibt *Clara* sich selber den «*vote*», da sie durch die «*Ordering Funktion*» dieselbe Permutationsreihe erhält. *Clara* hat nun zwei «*votes*», nämlich die von *Alice* und von sich selbst. Zum jetzigen Zeitpunkt sind jedoch keine «*votes*» von *Bob* und *Danilo* vorhanden. *Danilo* errechnet ebenfalls den Hash H_1 und sendet seinen «*vote*» an *Clara*. *Clara* hat nun die absolute Mehrheit («*supermajority*») der «Votes». Die absolute Mehrheit bildet sich aus $(\frac{Teilnehmer}{2}) + 1$. *Clara* sendet nun allen Netzwerkteilnehmern die «*Commit Message*», welche alle «*votes*» der Peers (*Alice, Clara und Danilo*) beinhaltet. *Bob* hat zur Zeit Probleme mit dem Netzwerk und erhält den Broadcast nicht. Jeder Peer, der die «*Commit Message*» erhalten hat, fügt Signaturen zum Block mit dem Hash H_1 und aktualisiert den lokalen Zustand.

Angenommen wird, dass *Bob* Internetprobleme hatte und somit die aktuelle Runde, wie die bisherigen, verpasst hat. Die «*Commit Message*» von *Clara* hat er ebenfalls nicht erhalten. *Bob* hat ausserdem seinen «*vote*» mit dem Hash H_1 nicht an *Clara* übermittelt, da er einen inkonsistenten Stand hat. *Bob* errechnet den Hash H_2 und erhält eine andere Peer Anordnung – [A_{lice}, B_{ob}, D_{anilo}, C_{lara}] basierend auf den von ihm errechneten Hash H_2. Anschliessend sende *Bob* seinen «*vote*» an *Alice*, weil *Alice* der erste Peer der Liste ist. *Alice* erhält den «*vote*». Da sie aber bereits eine «*Commit Message*» von *Clara* erhalten hat, sendet sie diese direkt an *Bob*. Er verifiziert die «*Commit Message*» von *Alice*, akzeptiert sie und wendet sie an. Nun hat *Bob* denselben Stand nach der Konsensrunde wie die anderen Peers .

5.3.8 Domäne

Ist ein abstrakter Name, um Accounts sowie Assets zu gruppieren. Bspw. kann eine Domäne eine Organisation in einer Gruppe, welche mit Iroha arbeiten, repräsentieren. Beliebig viele Assets sowie Accounts können derselben Domäne zugeordnet werden. Konten und Vermögenswerte, die in einer anderen Domäne mit demselben Namen, wie in der ersten Domäne erstellt wurden, sind separate und unabhängige Konten und Vermögenswerte.

5.3.9 Peer

Ein Knoten der am Iroha Netzwerk teilnimmt. Er beteiligt sich am Konsensprozess.

5.3.10 Berechtigung

Stellt eine benannte Regel dar, welche die Erlaubnis erteilt einen Befehl oder eine Abfrage auszuführen. Berechtigungen können nicht einem Account direkt zugewiesen werden. Stattdessen verfügt der Account über Rollen, bei denen es sich um Sammlungen von Berechtigungen handelt. Ausnahmen gibt es bei «*grantable*» Berechtigungen.

5.3.11 «Grantable» Berechtigung

Diese Berechtigungen können direkt einem Account zugewiesen werden. Ein Account der «*grantable*» Berechtigungen hat, kann bestimmte Aktionen für einen anderen Account durchführen. Wenn der Account *a@domain1* dem Account *b@domain2* Berechtigungen erteilt, damit dieser digitale Werte transferieren kann, dann kann der Account *b@domain2* die Assets von Account *a@domain2* an jede beliebige Person übertragen.

5.3.12 Proposal

Das ist ein Set von Transaktionen, die nur die «*stateless*» Validation durchlaufen haben. Die «*stateless*» Validierung wird für jede eingehende Transaktion vor der Angebotserstellung durchgeführt. Ein «*verified*» Proposal, ist ein Set von Transaktionen, welche die «*stateful*» sowie «*stateless*» Validierung durchliefen, jedoch noch nicht bestätigt wurden.

5.3.13 Quorum

Ein Quorum ist eine Zahl, welche die Mindestanzahl von Signaturen darstellt, die erforderlich sind, damit eine signierte Transaktion berücksichtigt werden kann. Der Standardquorumwert bei Iroha liegt bei eins. Jedes Konto kann zusätzliche öffentliche Schlüssel hinzufügen, um das Quorum zu erhöhen.

5.3.14 Rolle

Eine Rolle ist eine abstrakte Benennung, welche eine Reihe von Berechtigungen oder Regeln enthält, die einem Konto das Recht verleihen, eine Reihe von Befehlen oder Abfragen auszuführen.

5.3.15 Transaktion

Bildet ein geordnetes Set von Befehlen dar, welche atomar auf einem Hauptbuch angewendet werden. Jeder inkorrekte Befehl innerhalb einer Transaktion führt dazu, dass die gesamte Transaktion während des Validierungsprozesses abgelehnt wird.

5.3.16 Transaktionsstatus

Iroha unterstützt «*push & pull*» Interaktionen mit einem Client. Ein Client, der den «Pull Mode» nutzt, fragt den Transaktionsstatus vom Iroha-Peer ab indem er Transaktions-Hashs sendet und auf die Antwort wartet. Push Interaktion wird durch lauschen auf einen Event von Streams für jede Transaktionen durchgeführt. Die Abbildung 7 illustriert den Transaktionsablauf und deren Schritte eins bis neun. Die einzelnen Schritte werden in der Tabelle 6 erläutert. Die Nummerierung der Schritte definiert nicht den Ablauf, sondern dient als Erläuterung jedes einzelnen Schrittes.

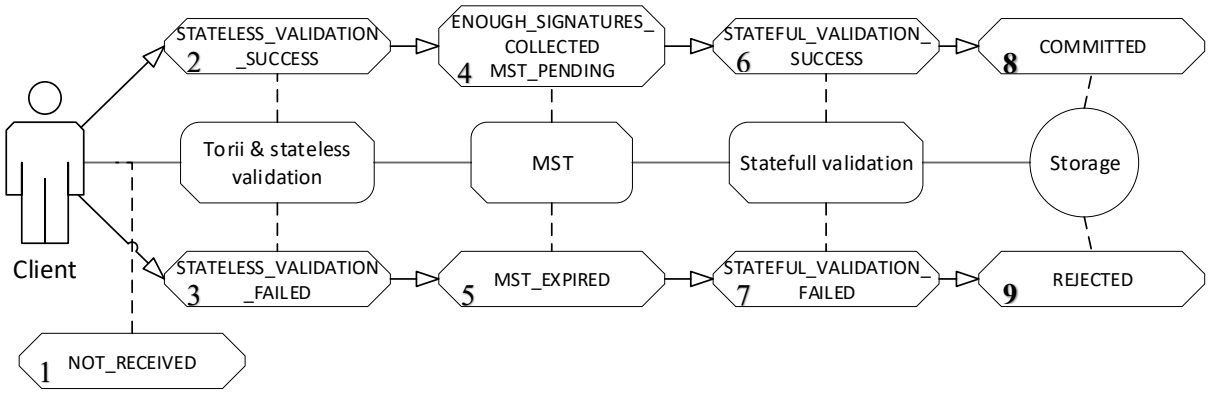

Abbildung 7. Transaktionsablauf bei Iroha.

Schritt	Erklärung
1	Der angefragte Peer hat die Transaktion nicht.
2	Die Transaktion hat die «*stateless*» Validierung erfolgreich bestanden. Dieser Status wird dem Client zurückgegeben, der die Transaktion weitergeleitet oder selbst initiiert hat.
3	Die Transaktion wurde mit Feldern kreiert, welche die «*stateless*» Validierungs-Bedingungen nicht erfüllen. Dem Client wird eine Antwort mit den verletzten Details gesendet.
4	Ist eine Multisignierte Transaktion (MST), welche genügend Signaturen hat und von einem Peer validiert wurde.
5	Diese MST ist nicht mehr gültig und wird deswegen vom Peer überschrieben und somit gelöscht.
6	Die Transaktion hat die «*stateful*» Validierung erfolgreich bestanden.
7	Diese Transaktion enthält Befehle, welche gegen die «*stateful*» Validierungs-Bedingungen verstossen. Ein Grund dafür können bspw. fehlenden Berechtigungen sein. Die Antwort erhält die verletzte Regel.
8	Die Transaktion ist ein Teil vom Block, welche genügend «*Votes*» erhalten hat und befindet sich im Moment im Block-Speicher.
9	Die Validierung wurde nicht bestanden. Die Transaktion wird als Hash im Block-Speicher abgelegt. So können Replay-Attacken präventiv vermieden werden.

Tabelle 6. Schritte des Transaktionsablaufes.

5.3.17 Transaktions-Batch

Der Transaktions-Stapel («*batch of transactions*») ist eine Funktion, mit welcher mehrere Transaktionen gleichzeitig an Iroha gesendet werden können. Dabei wird die Reihenfolge beibehalten. Jede Transaktion innerhalb eines Stapels enthält Batch-Metainformationen. Batch-Meta enthält eine Batch-Typ-ID (atomar oder geordnet) und eine Liste reduzierter Hashs aller Transaktionen innerhalb eines Stapels. Die Reihenfolge der Hashs definiert die Transaktionsreihenfolge.

Stapel können Transaktionen enthalten, die von verschiedenen Accounts erstellt wurden. Jede Transaktion innerhalb eines Stapels kann eine oder mehrere Signaturen erfordern. Jenes ist abhängig vom Quorum, welches für ein Konto des Erstellers der Transaktion festgelegt wurde. Eine Transaktion innerhalb eines Stapels sollte mindestens eine Signatur aufweisen, damit der Stapel die «*stateless*» Validierung bestehen kann.

5.3.18 Multisignierte Transaktionen

Eine Transaktion, deren Quorum größer als eins ist, wird als Multisignatur Transaktion «MST» betrachtet. Um die «*stateful*» Validation zu bestehen, ist eine Bestätigung der Signaturen vom Ursprungskonto notwendig. Diese Teilnehmer müssen die gleiche Transaktion mit ihrer Unterschrift senden.

5.3.19 Torii

Bildet den Einstiegspunkt für Clients. Torii verwendet als Transport «*gRPC*». Zur Interaktion mit Iroha kann jeder «*gRPC*» Endpunkte verwenden.

5.3.20 MST Processor

Ist ein interner «*gRPC*» Service, welcher Nachrichten von anderen Peers senden oder empfangen kann. Das Senden und Empfangen wird durch das «*Gossip*» Protokoll ermöglicht. Seine

Aufgabe ist es multisignierte Nachrichten, welche nicht genügend Signaturen für das Erreichen des Quorums haben, weiter zu versenden, damit diese das Quorum erreichen.

5.3.21 Peer Communication Service (PCS)

Interne Komponente von Iroha («*gRPC-Client*»), welche eine Intermediäre Stelle repräsentiert und Transaktionen von Torii durch den «*MST Prozessor*» zum «*Ordering-Gate*» weiterleitet. Ein Vorteil am PCS ist die Verminderung der Komplexität von den Interaktionen durch die Konsensimplementierung.

5.3.22 Ordering Gate

Bildet die interne Komponente von Iroha («*gRPC-Client*») ab, welche Transaktionen vom «*peer communication service*» zum «*Ordering Service*» weiterleitet. «*Order Gate*» erhält «*Proposal*» (potenzielle Blöcke in der Kette) vom «*Ordering Service*» und sendet diese zum «*Simulator*» für die «*stateful*» Validierung. Es fragt ebenfalls «*Proposals*» vom «*Ordering Service*» der Konsensrunde an.

5.3.23 Ordering Service

Ist eine interne Iroha Komponente («*gRPC-Server*»), welche Nachrichten von anderen Peers und mehrere Transaktionen enthält, welche die «*stateless*» Validierung durchlaufen haben. Jeder Knoten hat seinen eigenen «*Ordering Service*». Die Erstellung eines «*Proposals*» kann durch folgende Ereignisse ausgelöst werden:

- Die Frist für die Erfassung des «*Proposals*» ist abgelaufen.
- Der «*Ordering Service*» hat die maximale Anzahl an Transaktionen erhalten, welche für ein einziges «*Proposal*» zulässig sind.

Beide Parameter («*timeout*» und «*maximum size of proposal*») können angepasst werden. Eine Vorbedingung für beide Ereignisse ist, dass mindestens eine Transaktion den «*Ordering Service*» erreichen sollte. Ansonsten wird kein «*Proposal*» kreiert. Der «*Ordering Service*» führt auch eine vorläufige Validierung der «*Proposals*» durch.

5.3.24 Verified Proposal Creator

Diese interne Iroha Komponente dient für die «*stateful*» Validierung für die im «*Proposal*» enthaltenen Transaktionen, welche vom «*Ordering Service*» erhalten werden. Auf der Basis der Transaktionen, welche die «*stateful*» Validierung bestanden haben, wird ein «*verified Proposal*» erstellt und an den «*Block Creator*» weitergegeben. Alle Transaktionen, welche die «*stateful*» Validierung nicht bestanden haben, werden nicht berücksichtigt und sind somit nicht im «*verified Proposal*» enthalten.

5.3.25 Block Creator

Systemkomponente, die einen Block aus einem Set von Transaktionen bildet, welche «*stateless*» und «*stateful*» Validierung für die Weitergabe an den Konsens bestanden haben. Der «*Block-»* und «*vefified Proposal Creator*» bilden zusammen eine Komponente, die Simulator genannt wird.

5.3.26 Permutationsfunktion

Wenn ein Peer für einen Block-Hash abstimmt («*vote*»), generiert er eine Reihenfolge der zu validierenden Peers. Die Reihenfolge ist eine Permutationsliste von Peers, welche für die Abstimmung im Netzwerk benötigt wird. Die Reihenfolge wird von einer Funktion erstellt, welche als Parameter den Block-Hash sowie die initiale Liste der Peers verwendet.

5.3.27 Abstimmungsschritt – «Vote-Step»

Abstimmungsnachrichten für ein «*Proposal*» werden jedem Peer in der Reihenfolge, der vorher durch die Permutationsfunktion ergebenen Liste, übermittelt. Zwischen jeder Ausarbeitung besteht eine Verzögerung. Dieser Prozess wiederholt sich solange, bis ein «*commit*» oder eine «*Reject Message*» vom Netzwerk empfangen wird. Eine Iteration dieses Abstimmungsprozesses wird Abstimmungsschritt («*vote step*») genannt. Der Prozess der Weitergabe («*Propagation*») beginnt beim ersten Peer und endet beim letzten Peer gemäss der Liste (Vgl. Kapitel 5.3.26).

5.3.28 Commit

Eine «*Commit Message*» ist ein Set von «*votes*» für einen Block-Hash, signiert von der absoluten Mehrheit der Peers. Bspw. reicht es aus, wenn von insgesamt drei Netzwerk-Peers, zwei davon die absolute Mehrheit bilden. Dann reichen diese beiden Signaturen für eine «*Commit Message*» aus. Wenn ein Peer die absolute Mehrheit an «*votes*» gesammelt hat, sendet er diese allen Netzwerkteilnehmern als «*Commit Message*». Per Definition ist die absolute Mehrheit eine Zahl, die grösser ist als die Hälfte der gesamten Anzahl an Peers im Netzwerk $((\frac{Teilnehmer}{2}) + 1)$.

5.3.29 Reject

Eine «*Reject Message*» ist ein Set von «*votes*», welches darauf hinweist, dass Peers nicht die absolute Mehrheit an «*votes*» für den Block-Hash erreichen werden. Die «*Reject Message*» wird auf gleiche Art und Weise wie die «*Commit Message*» den Netzwerkteilnehmern übermittelt. Es besteht die Möglichkeit, dass ein Peer für ein «*Proposal*» stimmen kann, welches bereits «*commitet*» wurde. In diesem Fall wird er die «*Commit Message*» von dem Peer erhalten, welcher den «*vote*» erhalten hat. Dieser Prozess wird «*Commit Forwarding*» genannt.

6 Testing

Die Falsifikation der in Kapitel 1.3 definierten Hypothese *[4]* wird durch durch das Testen durchgeführt. Dabei wird die Open-Source Software Apache JMeter verwendet. JMeter ist eine Java Anwendung, welche das Testen von Funktionalitäten einer Applikation erlaubt und dabei bspw. die Performanz misst und die Ergebnisse virtualisiert. Bei der Entscheidungsfindung standen sich ApacheBench sowie Apache JMeter als Test-Software gegenüber. Apache JMeter wurde ausgewählt, da es über eine bedienbare Graphikoberfläche verfügt und mitgelieferte Visualisierungsmethoden anbietet.

Das Wiederlegen der Hypothesen *[1], [2] und [3]* wird durch das Herleiten von Informationen versucht. Dies wurde bereits in der Literaturrecherche in Kapitel 5.1 verfasst.

6.1 Laborumgebung

Für die Testdurchführung wird eine Laborumgebung mithilfe von zwei Rechnern (*R1* und *R2*) aufgebaut. *R1* (Vgl. Tabelle 7) agiert als B_{Light} Instanz. *R2* (Vgl. Tabelle 8) hingegen betreibt drei B_{Full} Instanzen. Die Laborumgebung besteht aus insgesamt vier Instanzen, welche als Benutzer der diva.exchange Applikation klassifiziert sind. Drei Instanzen (B_{Full}) werden mithilfe der Docker-Technologie erstellt und beinhalten den vollen Stack (Vgl. Kapitel 3.1). Der lokale Benutzer (B_{Light}) wird mit dem Paketmanager «*npm*» installiert. Die Kommunikation zwischen *R1* und *R2* wird mittels Portweiterleitung realisiert. Eine Funktionsausführung wird stets durch B_{Light} (*R1*) initiiert. Seine Anfrage wird an eine der drei Docker-Instanzen auf *R2* weitergeleitet

Eigenschaft	**Definition**
Systemversion	macOS 10.15 (19A583)
Prozessortyp	Dual-Core Intel Core i7
Prozessorgeschwindigkeit	3.5 GHz
Anzahl der Prozessoren	1
Gesamtanzahl der Kerne	2
Kernel Version	Darwin 19.0.0
L2-Cache (pro Kern)	256 KB
L3-Cache	4 MB
Speicher (RAM)	16 GB 2133 MHz LPDDR3
Speicher (SSD)	499.96 GB

Tabelle 7. Spezifikation des Rechners (B_{Light}) für Testumgebung.

Eigenschaft	Definition
Systemversion	Ubuntu 7.4.0-1ubuntu1~18.04.1
Prozessortyp	Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz
Prozessorgeschwindigkeit	3.4 GHz
Anzahl der Prozessoren	1 CPU & 4 Cores
Gesamtanzahl der Kerne	8
Kernel Version	4.15.0-64-generic
L2-Cache (pro Kern)	256 KB
L3-Cache	8 MB
Speicher (RAM)	64 GB 4 x 2133 MHz DIM DDR4
Speicher (SSD)	4.25 TB

Tabelle 8. Spezifikation des Rechners (B_{Full}) für Testumgebung.

In der Abbildung 8 wird der Aufbau des Labors illustriert. Die drei Benutzer B_{Full0}, B_{Full1} und B_{Full2} bilden die Docker-Instanzen auf *R2* ab. Die «*light*» Instanz befindet sich auf *R1* und ist zuständig für die Ausführung der Funktionen, welche dann von den Instanzen von *R2* abgearbeitet werden. Es gilt zu erwähnen, dass dieser Aufbau ohne die I2P -sowie Hangout-Komponente eingerichtet wurde. Die I2P-Komponente funktioniert einwandfrei und wurde von den Auftraggebern bereits getestet. Deswegen wird die I2P-Komponente, welche die Tests verzögern würde, nicht berücksichtigt. Die Hangout-Komponente hingegen ist zum jetzigen Stand (15.01.2020) noch nicht funktionsfähig.

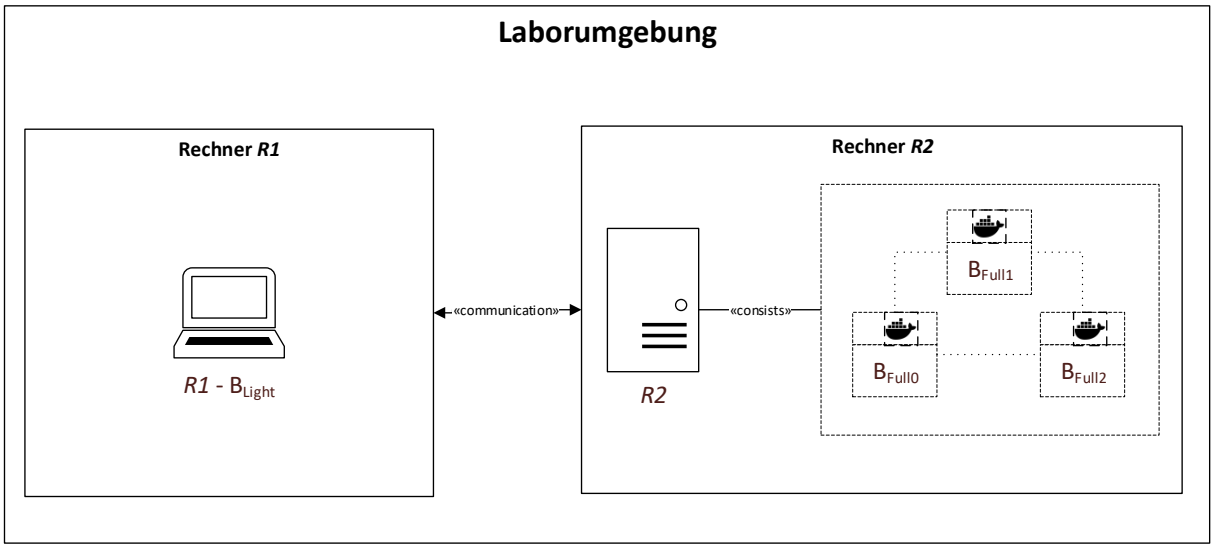

Abbildung 8. Aufbau Laborumgebung.

6.2 Teststrategie

Die Hypothese *[4]* bezieht sich auf die Performanz des Systems, welche bei unterschiedlichen Systemzuständen, für gleiche Aufgaben die ungefähr selbe Zeit benötigt. Dabei handelt es sich um Transaktionen, welche von einem oder mehreren Benutzern initiiert und dann auf der Iroha-Blockchain persistiert werden. Solche Transaktionen sind bspw. Registrierungen von mehreren Benutzern. Dabei agieren mehrere Systemkomponenten miteinander, um die Registrierung durchzuführen. Den ersten Einstiegspunkt für einen Benutzer bildet Registrationsoberfläche. Dabei muss der Benutzer ein Passwort festlegen. Anschliessend wird für den Benutzer ein einzigartiger Benutzername generiert. Diese Transaktionsdaten und noch weitere werden auf der Iroha-Blockchain in einen Block geschrieben. Die Blockhöhe wird durch die Anzahl Blöcke definiert. Ein Block kann maximal zehn Einträge beinhalten. Im Fall der Registrierung sind das zehn Einträge von neuen Benutzern. Der erste Block einer Blockchain wird als Genesis-Block bezeichnet und beinhaltet spezifische Daten über die Blockchain selbst. Die Systemkomponenten, welche bei diesem Beispiel verwendet werden, sind hauptsächlich die Diva- sowie Iroha-Komponente.

Bei diesem Ansatz handelt es sich gemäss dem International Software Testing Qualifications Board um den «*hardest first*» Ansatz. Bei dieser Test-Art werden zuerst die wichtigsten Funktionen getestet. Die Registration bildet den Einstiegspunkt von diva.exchange und ist notwendig, damit die Funktionalitäten verwendet werden könne.

6.3 Testkonfiguration

Wie in Kapitel 6.1 bereits erläutert, wurde für die Testdurchführung eine Laborumgebung eingerichtet. Dabei werden mithilfe der Software JMeter, welche sich auf *R1* befindet, Benutzerregistrierungen durchgeführt und von den «*full*» Clients auf *R2* abgewickelt und persistiert. Durch die Schnittstelle (API) ist es möglich mit einem *GET* Befehl einen Benutzer zu kreieren. Dieser Befehl sowie die Adresse des Servers können dann bei JMeter mitgegeben werden. Die Antwort, welche man nach diesem Befehl erhält, ist die «*job id*». Durch einen weiteren Befehl mit der «*job id*» könnte man überprüfen, wie der Status («*job sate - js*») dieses «*jobs*» ist.

JMeter bietet verschiedene Konfigurationsparameter an, welche es erlauben, den Test den Bedürfnissen entsprechend anzupassen. Die korrekte Konfiguration wurde mit dem Auftraggeber abgestimmt und festgesetzt. Die wichtigsten Parameter für diese Testdurchführung sind die Anzahl Threads («*number of threads*»), die Anlaufzeit in Sekunden («*ramp-up period*») sowie die Anzahl Wiederholungen («*loop count*»). Die Anzahl Threads simulieren unterschiedliche Benutzer, welche die Registrierung vornehmen. Bei der Anlaufzeit wird die Wartezeit in Sekunden zwischen dem Thread oder der Threadgruppen definiert. Soll ein Test wiederholt werden, so kann die Anzahl Wiederholungen auf eine Ganzzahl gesetzt werden. Angenommen man hat fünf Benutzer mit einer Anlaufzeit von fünf Sekunden, welche die Registrierung fünf Mal durchführen, dann ist eine ungefähre Testdauer von 25 Sekunden zu erwarten. Die totale Ausführung der Funktion ergibt sich aus der Multiplikation der Anzahl Threads mit der Anzahl an Wiederholungen. In Diesem Beispiel sollte die Funktion demnach 25 Mal ausgeführt worden sein.

Die Konfiguration von JMeter für den durchgeführten Test kann der Tabelle 9 entnommen werden.

Threads	Anlaufzeit (s)	Wiederholungen	Erwartete Testdauer	Total
100	5	1’000	83 Minuten	100’000

Tabelle 9. Konfiguration des Tests.

Der Test wird zweimal durchgeführt und beinhaltet je sechs Testschritte. Während der Testdurchführung werden folgende Parameter festgehalten.

- jsOK
- Anzahl erfolgreich abgearbeiteter Befehle.
- jsPending
- Anzahl sich in Bearbeitung befindender Befehle.
- jsError
- Anzahl fehlerbehafteter Befehle.
- Zeitstempel des ersten geschriebenen Blocks.
- Wird von der Blockchain ausgelesen. Dies ist jeweils der erste Zeitstempel des ersten Blocks, welcher in diesem Testschritt kreiert wurde.
- Zeitstempel des letzten geschriebenen Blocks
- Wird von der Blockchain ausgelesen. Dies ist jeweils der letzte Zeitstempel des Blocks, welcher in diesem Testschritt kreiert wurde.
- Differenz der beiden Zeitstempel
- Ergibt sich durch die Subtraktion der Zeitstempel vom letzten und ersten Block pro Testdurchlauf.
- Anzahl Benutzer
- Dies sind die total persistierten Benutzer pro Testdurchlauf, welche sich auf der Blockchain befinden.
- Blockhöhe pro Testschritt
- Dies ist die Höhe des letzten Blocks pro Testschritt.
- Anzahl Blöcke pro Testschritt
- Diese Angabe ergibst sich aus Differenz der Blockhöhe des vorherigen Testschrittes und der aktuellen Blockhöhe nach dem ausgeführten Testschritt.

Ein weiterer Parameter wurde für Docker-Container der Iroha Instanzen angepasst. Dies ist die «*shared memory size*». Diese wurde auf vier Gigabyte gesetzt und erlaubt den Containern diese maximale Nutzung des RAM-Speichers.

6.4 Testergebnisse

Dieses Kapitel fasst die Daten zusammen, welche bei den Testdurchläufen entstanden sind. Anschliessend wird eine Interpretation dieser Daten durchgeführt (Kapitel 7.2). Es werden pro Testdurchlauf 100’000 Benutzer kreiert. Es wird erwartet, dass jeweils pro Testdurchlauf alle Benutzer auf der Blockchain persistiert werden. Es wird ebenfalls erwartet, dass die steigende Blockhöhe die Performanz des Systems nicht beeinträchtigt und somit keine Degradierung entsteht. Das bedeutet, dass bei jedem Testdurchlauf 100'000 Benutzer in ungefähr gleicher Zeit auf der Blockchain persistiert werden.

Die Durchführung der der sechs Testschritte wurde je zweimal durchgeführt, um eine verlässlich Datenbasis für die Auswertung zu erhalten. Die «*shared memory size*» wurde im Docker-Compose Konfigurationsfile auf vier Gigabyte gesetzt. Standardgemäss ist die «*shared memory size*» auf ein Gigabyte gesetzt. Beim Testen mit dieser Grösse (1GB) konnte das System mit dieser Anzahl an Anfragen nicht nachkommen und stürzte ab.

In der Abbildung 9 sind die Daten vom ersten Test ersichtlich. Bei allen Testschritten wurden sämtliche Benutzer erstellt und auf der Blockchain persistiert, obwohl bei jedem der sechs Testschritte Fehlermeldungen («*jsError*») generiert wurden. Bei den Fehlermeldungen handelt es sich immer um die gleiche («*Error: Command response error: expected=COMMITTED, actual=NOT_RECEIVED*»). In Kapitel 5.3.16 wurde dieser Transaktionsstatus («*NOT_RECEIVED*») erläutert. Dabei fragt der Peer nach dieser Transaktion und weil diese Transaktion nicht vorhanden ist, wird diese Fehlermeldung geworfen. Dies bedeutet für diesen Testfall, dass dieser Mechanismus zu früh ausgeführt wird und dass zu diesem Zeitpunkt die nachgefragte Transaktion noch nicht von einem Peer entgegengenommen wurde. Vermutlich handelt es sich dabei um einen unterliegenden Implementationsfehler von Iroha, weil die Transaktion trotzdem erfolgreich auf die Blockchain geschrieben wird. Eine Änderung dieser Funktionalität wurde der Iroha Community vom Auftraggeber vorgeschlagen.

Test	jsOK	jsPENDING	jsERROR	Total	First Block	Last Block	Duration	Users	Blockheight	Blocks written
1	99920	0	80	100000	17:59:09	19:08:04	01:08:55	100000	14359	14359
2	99896	0	104	100000	12:19:16	13:27:21	01:08:05	200000	28699	14340
3	66984	43	32973	100000	13:52:23	14:52:29	01:00:06	300000	41884	13185
4	6235	0	93765	100000	15:12:15	18:52:50	03:40:35	400000	54074	12190
5	5840	0	94160	100000	15:18:22	17:37:27	02:19:05	500000	66361	24477
6	2561	0	97439	100000	17:54:13	21:11:57	03:17:44	600000	79024	24950

Abbildung 9. Ergebnisse der sechs Testdurchläufe – Erste Durchführung.

Der Systemzustand sowie die Testkonfigurationen waren für den zweiten genau die gleichen wie beim ersten Testdurchlauf. In der Abbildung 20 sind die Testresultate zusammengefasst.

Test	jsOK	jsPENDING	jsERROR	Total	First Block	Last Block	Duration	Users	Blockheight	Blocks written
1	99782	0	218	100000	13:18:12	14:34:44	01:16:32	100000	12413	14359
2	99850	0	150	100000	14:58:03	16:02:32	01:04:29	200000	26641	14228
3	68321	56	31623	100000	16:09:44	17:20:02	01:10:18	300000	41212	14571
4	6199	0	93801	100000	17:32:58	20:59:57	03:26:59	400000	54312	13100
5	4362	0	95638	100000	21:40:24	23:59:33	02:19:09	500000	65422	24210
6	2286	0	97714	100000	00:17:27	03:55:18	03:37:51	600000	80033	25721

Abbildung 10. Ergebnisse der sechs Testdurchläufe – Zweite Durchführung.

Die Durchführung beider Testdurchläufe zeigt auf, dass alle Benutzer auf die Blockchain persistiert wurden. Die benötigte Zeit für die Speicherung der Benutzer auf die Blockchain war bei beiden Testdurchläufen bis zum dritten Testschritt ungefähr gleich. Vom vierten bis zum letzten Testschritt benötigte das Persistieren der Benutzer deutlich mehr Zeit, obwohl die Menge der Benutzer wie in den vorherigen Testschritten dieselbe war. Dies war bei beiden Testdurchläufen der Fall.

In der Abbildung 11 werden die beiden Testdurchläufe und deren sechs Testschritte visualisiert. Auf der *X-Achse* ist die Anzahl Benutzer definiert, welche nach einem Testschritt auf der Blockchain persistiert wurden. Maximal kann es 600'000 Benutzer haben, da bei jedem der sechs Testschritte 100'000 Benutzer erstellt werden. Die *Y-Achse* beinhaltet die Dauer in Stunden, Minuten und Sekunden, welche für die Speicherung der Benutzer auf die Blockchain benötigt wurde. Die blauen Linien zeigen den ersten (Abbildung 9) und die orangen den zweiten Testdurchlauf (Abbildung 10) auf.

Die beiden Symbole (Rhomboid und Quadrat), welche die Linien verbinden, bilden die einzelnen Testschritte jedes Testes ab. Ein Symbol im Graphen zeigt dementsprechend auf, wie lang das Persistieren von jeweils 100'000 Benutzern bei einem gewissen Stand an Benutzern benötigt hat. Jeder erste Testschritt hat bei der Blockhöhe eins angefangen. Für die Auswertung wurde dieser Block nicht berücksichtigt, weil es sich um den Genesis-Block handelt.

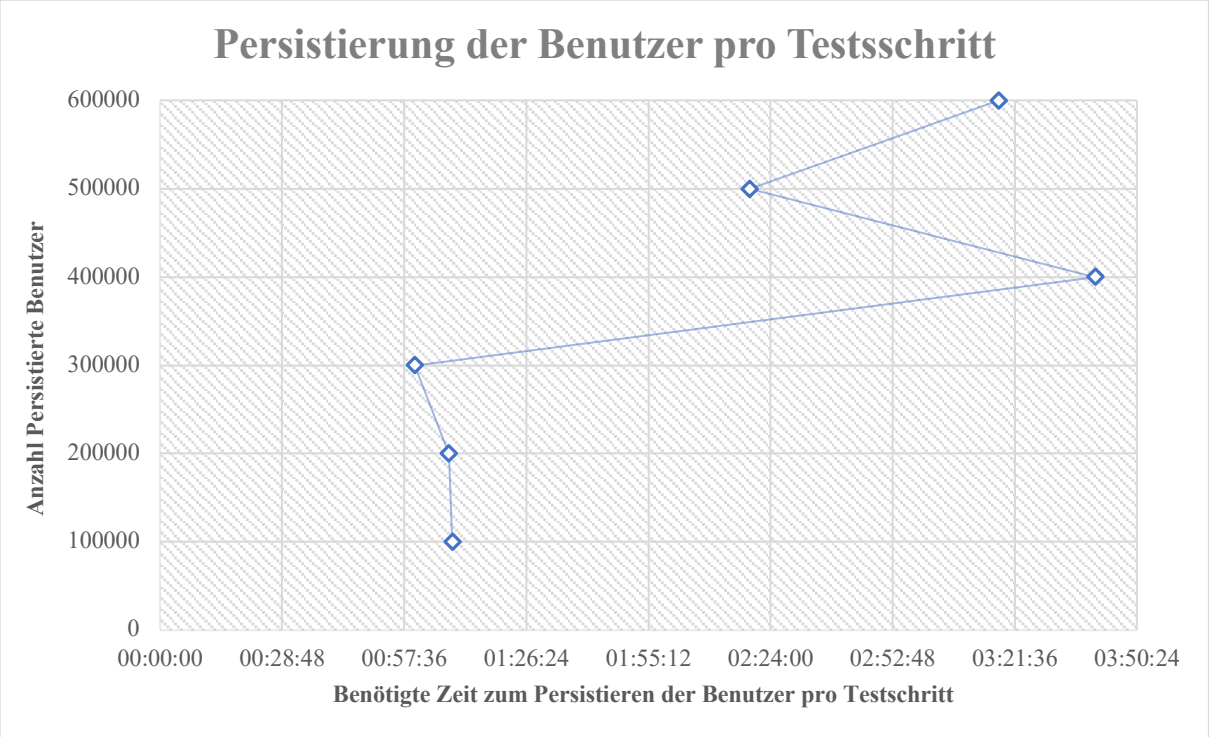

Abbildung 11. Benutzererstellung.

7 Falsifikation

In diesem Abschnitt wird der Bezug auf die in Kapitel 1.3 formulierten Hypothesen vorgenommen.

7.1 Hypothesen *[H1]*, *[H2]* und *[H3]*

Die Hypothese *[H1]* konnte nicht wiederlegt werden. Die Literaturrecherche ergab, dass eine steigende Anzahl an Netzwerkknoten im I2P-Netzwerk die Ausfallwahrscheinlichkeit reduziert und zugleich die Transaktionskapazität erhöht [21]. Es wird angenommen, dass eine steigende Anzahl an neuen I2P-Knoten mit einer gewissen wahrscheinlich auch Knoten mit dem Rang vier (Vgl. Kapitel 5.1.4) beinhalten wird. Solche Knoten können für den Tunnelaufbau verwendet werden und erhöhen damit die Bandbreite und vermindern zugleich die Latenz im I2P-Netzwerk. Durch diese Tatsache kann die Hypothese *[H2]* ebenfalls nicht falsifiziert werden.

Eine steigende Anzahl an von neuen I2P-Knoten wird vermutlich den Netzwerkverkehr und die Anzahl möglicher unterschiedlicher Routen erhöhen. Bei einer Anzahl von elf verschiedenen I2P-Netzwerkknoten, welche alle den Rang vier aufweisen, gibt es für einen Sender 210 ($\frac{10!}{6!(10-6)!}$) Kombinationen einen Tunnel mit der Länge sechs für die Transferierung der Nachrichten zu verwenden. Hinzuzufügen ist, dass ein Sender nicht an einem Tunnel beteiligt sein kann. Die kombinatorischen Möglichkeiten ergeben sich ohne Wiederholungen bzw. ohne die Berücksichtigung der Reihenfolgen. Angenommen, dass nun zehn weitere I2P Netzwerkknoten mit den Rängen vier dazukommen, sind es dann insgesamt 21 Netzwerkknoten. Gemäss dieser Annahme ergeben sich für jeden Netzwerkknoten nun 38’760 ($\frac{20!}{6!(20-6)!}$) Kombinationen, um einen Tunnel mit der Länge sechs zu verwenden. Die Tabelle 10 zeigt die verschiedenen Möglichkeiten für die Tunnelerstellung von Netzwerknoten, welche dem Rang vier zugeordnet sind.

Rang 4 Knoten	**Anzahl kombinatorische Möglichkeiten ohne Wiederholung**
10	210
14	3’003
18	18’564
22	74’613
26	230’230
30	593’775
40	3'838’380
45	8'145’060
50	15'890’700

Tabelle 10. Kombination möglicher Tunnelverbindungen.

Die Routen für gesendete Nachrichten, welche durch die Tunnel bereitgestellt werden, ändern sich alle zehn Minuten. Bei einer hohen Möglichkeit an Kombinationen für Tunnel, wird es Angreifern erschwert, Muster zu erkennen und somit auf die Identität eines Senders zu schliessen. Somit steigt mit der Anzahl von Rang vier I2P-Netzwerknoten auch die Anonymität und somit die Privatsphäre der Benutzer. Die Abbildung 12 zeigt einen Grafen, wobei die Anzahl Rang vier Netzwerkknoten (*X-Achse*) in Relation mit den möglichen Kombinationen für die Tunnel steht (*Y-Achse*). Der steigende Graph kann als steigende Komplexität für Angreifer betrachtet werden. Ebenfalls wiederspiegelt er einen Anonymitätsgrad für Benutzer. Diesen Tatsachen entsprechend kann die Hypothese *[3]* nicht wiederlegt werden.

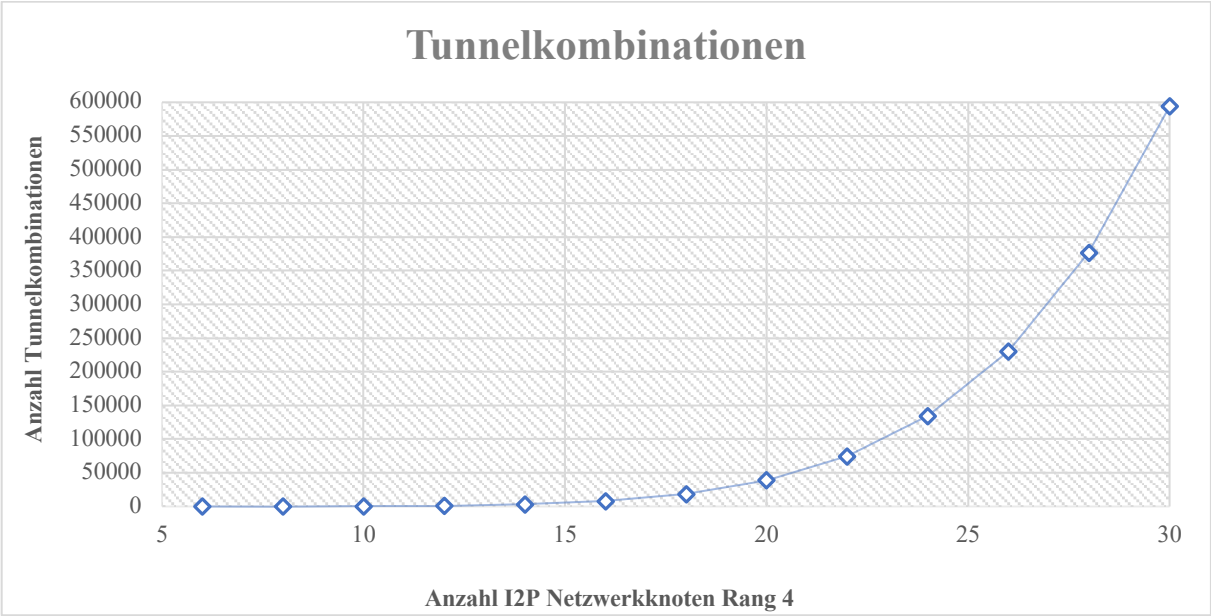

Abbildung 12. Anzahl Rang vier Knoten vs. Anzahl Tunnelkombinationen.

7.2 Hypothese *[4]*

Die Erwartung für die beiden Tests lag darin, dass alle Benutzer auf die Blockchain persistiert werden. Diese Erwartung wurde erfüllt. Ein weiteres Kriterium war, dass alle Benutzer in ungefähr derselben Zeitdauer auf die Blockchain geschrieben werden. In der Abbildung 13 visualisiert die rote Linie das ungefähre Verhalten, welches erwartet wurde. Die beiden anderen Linien visualisieren den tatsächlichen Testverlauf. Somit wurde für das Speichern der Benutzer ab dem dritten Testschritt beider Testdurchläufe deutlich mehr Zeit benötigt. Damit kann angenommen werden, dass ab einer gewissen Blockhöhe oder der Anzahl persistierter Transaktionen das System degradiert. Die Hypothese *[4]* wurde somit falsifiziert.

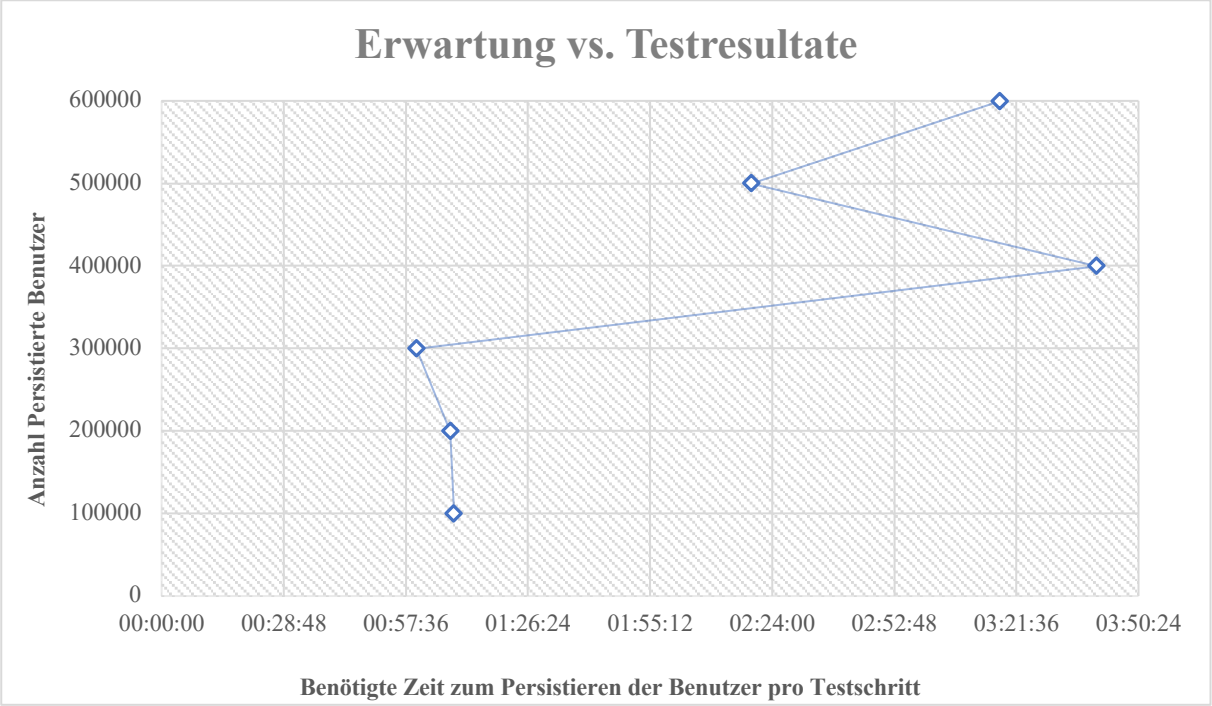

Abbildung 13. Erwartung vs. Testresultate.

8 Fazit und Ausblick

Mit dem vorliegenden Forschungsbericht wurde versucht, in einem noch sehr unbekannten und neuen Bereich eine Informationsbasis aufzubauen, um die gestellten Hypothesen durch Literaturrecherche und Tests zu falsifizieren. Dabei konnten die Hypothesen *[1]*, *[2]* und *[3]* nicht wiederlegt werden. Die Hypothese *[4]* hingegen konnte durch Tests falsifiziert werden. Es wurde ebenfalls eine Marktrecherche durchgeführt, welche zwei ähnliche Projekte mit dem jetzigen vergleicht. Diese beiden Projekte (Bisq und Resistance) unterscheiden sich in den wichtigsten Punkten zum diva.exchange.

Für die Hypothesen *[1]*, *[2]* und *[3]* wurde die Methodik der Literaturrecherche angwendet, um diese durch das Herleiten von Schlussfolgerungen zu falsifizieren. Keine dieser Hypothesen konnte wiederlegt werden. Dabei handelt es sich stets um die Frage, was eine steigende Anzahl an neuen I2P-Knoten für Folgen hat. Durch neue Zugänge kann angenommen werden, dass die Ausfallwahrscheinlichkeit des Netzwerkes sinkt, da das Netzwerk durch alle Teilnehmer gebildet wird und dadurch mehr I2P-Router im Netzwerk verfügbar sind. Ein weiterer Effekt der steigenden Anzahl an I2P-Knoten wirkt sich auch auf die Transaktionskapazität aus. Dies bedeutet, dass die Bandbreite des Netzwerkes erhöht und die Latenz vermindert wird. Die Anonymität und somit der Schutz der Privatsphäre werden durch die steigende Anzahl der I2P-Knoten erhöht. Die Anonymität der Benutzer kann in Relation zu der Anzahl unterschiedlicher Tunnelverbindungen gesetzt werden. Somit wird durch jeden neuen I2P-Knoten die Anzahl möglicher Tunnelverbindungen erhöht, was gleichzeitig die Anonymität jedes I2P-Knoten erhöht. Die Wiederlegung konnte somit nicht hergeleitet werden, was für den weiteren Projektverlauf als positiv eingestuft werden kann.

Die Hypothese *[4]* konnte durch die Durchführung von Tests wiederlegt werden. Dabei wurden jeweils zwei Tests mit je sechs Testschritten durchgeführt. Jeder Testschritt beinhaltet die Registrierung von 100'000 Benutzern. Es wurde erwartet, dass alle Benutzer in ungefähr gleicher Zeitdauer auf der Blockchain persistiert werden und das unabhängig von der Grösse der Blockchain (Blockhöhe). Für die ersten drei Testschritte beider Tests benötigte das System ungefähr dieselbe Zeit für das Speichern aller Benutzer. Ab dem vierten Testschritt war zu erkennen, dass das System alle Benutzer auf die Blockchain persistiert, jedoch dafür deutlich mehr Zeit verwendet hat. Dieses Verhalten konnte bis zum letzten Testschritt beider Tests beobachtet werden. Somit kann behauptet werden, dass das System ab einer gewissen Blockhöhe degradiert.

Die Marktanalyse, bei welcher Resistance und Bisq dem jetzigen Projekt gegenübergestellt wurden, zeigte auf, dass sich diva.exchange von ihnen unterscheidet. Bei Resistance benötigt es die Überprüfung von jeder Person, welche handeln möchten. Dokumente, welche die Identität eindeutig bestimmen, müssen eingereicht werden. Somit liegt hier der Unterschied bei der Anonymität, die bei Resistance gebrochen wird. Bei Bisq hingegen ist nur der Handel gegen Bitcoin möglich. Bei Fiat-Transaktionen wird beiden Parteien die Identität des anderen offengelegt und somit die Anonymität verletzt. Ein weiterer Unterschied ist, dass Resistance sowie Bisq eine eigene Kryptowährung haben. Bei diesem Projekt wird/darf dies nicht der Fall sein.

Die Testdurchführung hat mehr Zeit beansprucht als eingeplant wurde, weil während der Testdurchführung Fehler entdeckt wurden. Deswegen bestand eine ständige Wechselwirkung zwischen dem entwickeln des Software-Prototypen und der Testdurchführung. Für den Auftraggeber waren diese Fehler jedoch nützliche Informationen, weil dadurch die Applikation verbessert wurde. Aus diesem Grund wird diese Fehlerfindung ebenfalls als Mehrwert betrachtet, welcher durch diesen Forschungsbericht generiert wurde.

Die Schwierigkeit bei diesem Forschungsprojekt bestand hauptsächlich in der Einzigartigkeit des Vorhabens und deswegen stellte es sich als schwierig heraus, Informationen bzw. geeignete Quellen dafür zu finden. Die Iroha Blockchain ist ein noch eher neues Projekt und wurde am 06. Mai vorgestellt [26]. Quellen und wissenschaftliche Artikel waren zum Zeitpunkt des Schreibens nicht vorhanden. Bei I2P sind Quellen vorhanden, jedoch handelt es sich mehrheitlich dabei um ältere Quellen.

Das Projekt befindet sich nach wie vor in der Anfangsphase. Trotzdem sind sich die Auftraggeber und der Verfasser dieses Forschungsberichtes einig, dass durch dieses Projekt ein erheblicher Mehrwert entstanden ist. Während der dreimonatigen Projektdauer wurde täglich miteinander gearbeitet und dies ermöglichte den Austausch von wichtigen Erkenntnissen. Die nächsten Abschnitte erklären den Ausblick für das weitere Projektvorgehen.

Der Ausblick beinhaltet einerseits die Problematik bezüglich der Auslösung einer Transaktion auf einer Blockchain und die damit verletzte Anonymität gemäss [Marx 2001 [4]] und andererseits das mögliche Angriffsszenario mit Replay-Attacken. Jede Transaktion auf einer Blockchain muss ausgelöst werden, bevor ein Konsensmechanismus über die Korrektheit der Transaktion entscheiden kann. Diese Auslösung entsteht auf einem bekannten, nicht-anonymen Knoten. Wenn also ein Knoten eigene Transaktionen initiiert, so sind diese Transaktionen mit dem Knoten assoziiert und damit nicht mehr anonym. Um dieses Problem zu lösen wurden während dieses Forschungsberichtes zwei Lösungsansätze untersucht. Der erste Lösungsansatz bestand darin, die gesamte Blockchain im Darknet (I2P) zu betreiben. Die Durchführung von Tests zeigte auf, dass die Performanz des Systems dadurch sehr geschwächt wird. Der zweite und implementierte Lösungsansatz bestand darin, dass die Blockchain im Clearnet betrieben wird. Deswegen wurde eine Stellvertreterfunktion (API) implementiert, welche im Auftrag eines Urhebers (Sender), dessen Transaktion und die damit verbundenen Daten auf die Blockchain schreibt. Somit wird verhindert, dass der Urheber identifiziert werden kann. Umgekehrt muss es beweisbar sein, dass der Stellvertreter, welcher die Transaktion auf die Blockchain schreibt, nicht der Urheber (tatsächlicher Sender) ist. Die Teilnahme am I2P-Netzwerk beweist, dass derjenige der auf die Blockchain schreibt, nicht der Urheber ist [15]. Diese Implementierung des Stellvertreters muss überprüfen können, ob eine Nachricht vom tatsächlichen Sender (Ursprung) stammt. Da der Sender die Nachricht mit seinem privaten Schlüssel verschlüsselt, kann der Stellvertreter mit dem öffentlichen Schlüssel, welcher sich auf der Blockchain befindet, überprüfen, ob es sich dabei um den tatsächlichen Sender handelt. Dieser Vorgang beschreibt das asymmetrische Schlüsselverfahren.

Nun besteht die Gefahr, dass ein Mittelsmann die komplette Nachricht des Senders abfängt, diese kopiert und dadurch eine Replay-Attacke durchführen kann. Der Angreifer könnte eine gewisse Zeit warten und dann die Nachricht wieder versenden. Der Stellvertreter, welcher die Nachricht auf den Ursprung überprüft, stellt keine Unstimmigkeiten fest und schreibt die Daten erneut auf die Blockchain. Für dieses Problem wurden zwei Mechanismen eingeführt, welche Replay-Attacken vermutlich komplett verhindern. Der erste Lösungssatz sieht vor, dass der Sender einen Zeitstempel der Nachricht hinzufügt. Dieser ist nur 20 Sekunden gültig. Alle Nachrichten, die nicht in 20 Sekunden beim Stellvertreter ankommen, werden als ungültig betrachtet und nicht behandelt. Es besteht jedoch weiterhin die Gefahr, dass ein Angreifer die Nachricht innerhalb der erlaubten 20 Sekunden kopiert und an einen Stellvertreter übermittelt. Dann würden die Daten erneut auf die Blockchain geschrieben werden. Um das zu unterbinden wurde der Deltamechanismus (Zustandsmechanismus) implementiert. Dabei nutzt der Stellvertreter den Vorteil, dass er den korrekten Datenzustand vom tatsächlichen Sender (Ursprung) kennt. Den Datenzustand kennt er, weil dieser auf der öffentlichen Blockchain vorhanden ist. Der Sender kennt seinen eigenen Zustand auch. Beide Parteien kennen somit den Ist-Zustand vor der Änderung. Dieser Ist-Zustand wird ebenfalls in die Nachricht mit dem Zeitstempel integriert und der Sender signiert diese Nachricht. Damit können vermutlich Replay-Attacken verhindert werden.

Dieser Schutzmechanismus wurde noch nicht getestet und somit kann keine Effektivität gewährleistet werden. Im weiteren Projektverlauf müssen die Gegenmassnahmen, welche implementiert wurden, überprüft werden.

9 Abkürzungsverzeichnis

AHV	Alters- und Hinterlassenenversicherung
bspw.	beispielsweise
BTC	Bitcoin
bzw.	beziehungsweise
CHF	Schweizer Franken
d.h.	das heisst
etc.	et cetera
ETH	Ethereum
GB	Giga Byte
gRPC	Remote procedure call (Google)
KB	Kilo Byte
MB	Mega Byte
Mhz	Megaherz
MST	multisignature
OR	Onion Router
PCS	Peer Communication Service
u.a.	unter anderem
usw.	und so weiter
vgl.	Vergleiche
YAC	Yet Another Consensous
z. B.	zum Beispiel

10 Abbildungsverzeichnis

11 Tabellenverzeichnis

12 Literaturverzeichnis

[1] A. König, „Nau,“ 8 Mai 2019. [Online]. Available: https://www.nau.ch/news/wirtschaft/binance-wurden-bitcoins-im-wert-von-42-millionen-franken-gestohlen-65519892. [Zugriff am 15 Januar 2020].

[2] J. Young, „Cointelegraph,“ 18 Juni 2019. [Online]. Available: https://cointelegraph.com/news/round-up-of-crypto-exchanges-hack-so-far-in-2019-how-can-it-be-stopped. [Zugriff am 03 Januar 2020].

[3] Dudenredaktion, „Duden online,“ o.j.. [Online]. Available: https://www.duden.de/node/6772/revision/6799. [Zugriff am 15 Januar 2020].

[4] G. T. Marx, „Identity and Anonymity: Some Conceptual Distinctions and Issues for Research,“ 16 Februar 2001. [Online]. Available: https://web.mit.edu/gtmarx/www/identity.html. [Zugriff am 15 Januar 2020].

[5] Diva, „GitLab,“ 30 Juli 2019. [Online]. Available: https://gitlab.com/diva.exchange/diva. [Zugriff am 15 Januar 2020].

[6] Broadbandnow, „Broadbandnow,“ 15 Januar 2020. [Online]. Available: https://broadbandnow.com/All-Providers. [Zugriff am 15 Januar 2020].

[7] I2P, „i2pmetrics,“ 18 Dezember 2019. [Online]. Available: http://i2pmetrics.i2p/network-size . [Zugriff am 19 Dezember 2019].

[8] C. Beams, „34C3 ChaosWest - Bisq - A decentralized bitcoin exchange,“ 18 März 2018. [Online]. Available: https://www.youtube.com/watch?time_continue=90&v=Fv-eCchzBZA. [Zugriff am 15 Januar 2020].

[9] C. Beams, S. Jain, D. Apostolou und A. David, „Bisq - The peer-to-peer bitcoin exchange,“ 12 Dezemer 2019. [Online]. Available: https://docs.bisq.network/exchange/whitepaper.html. [Zugriff am 15 Januar 2020].

[10] Whitepaper, „Resistance. The First Privacy-Focused Decentralized Exchange & Blockchain,“ Juni 2019. [Online]. Available: https://docs.resistance.io/Resistance_Whitepaper_v1.5.pdf. [Zugriff am 15 Januar 2020].

[11] C. Paar und J. Pelzl, „Einführung in Die Kryptographie und Datensicherheit.,“ in *Kryptografie verständlich: Ein Lehrbuch für Studierende und Anwender*, Springler, 2016, pp. 24-25.

[12] S. Schulte, M. Sigwart, P. Frauenthaler und M. Borkowski, „Towards Blockchain Interoperability,“ Distributed Systems Group, TU Wien, Wien, 2019.

[13] J. P. Timpanaro, I. Chrisment und O. Festor, „I2P’s usage characterization,” Traffic Monitoring and Analysis,„ Henri Poincar´e University, Nancy-Grand Est, 2011.

[14] M. Ehlert, „I2P Usability vs. Tor Usability A Bandwidth and Latency Comparison,“ Humboldt University of Berlin, Berlin, 2011.

[15] I2P-Documentation, „I2P,“ Juli 2019. [Online]. Available: https://geti2p.net/en/docs/how/tech-intro. [Zugriff am 15 Januar 2020].

[16] M. Herrmann und C. Grothoff, „Privacy-implications of performancebased peer selection by onion-routers:,“ in *Privacy-Implications of Performance-Based Peer Selection by Onion-Routers: A Real-World Case Study Using I2P*, Waterloo, Springer, 2011, pp. 155-174.

[17] J. Abawajy, A. Kelarev und M. Chowdhury, „Automatic Generation of Meta Classifiers with Large Levels for Distributed Computing and Networking,“ School of Information Technology, Deakin University, Burwood, 2014.

[18] L. Schimmer und z. Pseudonym, „Peer Profiling and Selection in the I2P Anonymous Network,“ k. A., k. A., 2009.

[19] P. Maymounkov und M. David , „Kademlia: A peer-to-peer information system based on the xor metric,“ New York University, New York, 2002.

[20] B. Zantout und R. A. Haraty, „I2P Data Communication System,“ Lebanese American University, Beirut,, 2011.

[21] B. Conrad und F. Shirazi, „A Survey on Tor and I2P,“ Department of Computer Science, Darmstadt, 2014.

[22] J. Ren und J. Wu , „Survey on anonymous communications in computer networks,“ in *Computer Communications*, Elsevier Science Publishers B. V., 2010, pp. 420-431.

[23] Hyperledger, „Hyperledger Iroha Documentation,“ Hyperledger, 25 Dezember 2019. [Online]. Available: https://iroha.readthedocs.io/en/latest/. [Zugriff am 15 Januar 2020].

[24] P. Egloff und E. Turnes, „Synchronisierung durch Konsensmechanismus,“ in *Blockchain für die Praxis: Kryptowährungen, Smart Contracts, ICOs und Tokens*, Muttenz, Schwabe AG, 2019, p. 52.

[25] F. Muratov, A. Lebedev, N. Iushkevich, B. Nasrulin und T. Makoto, „YAC: BFT Consensus Algorithm for Blockchain,“ Soramitsu, k. A., 2018.

[26] L. Foundation, „THELINUXFOUNDATION,“ Linux Foundation, 6 Mai 2019. [Online]. Available: https://www.linuxfoundation.org/blockchain-data-analytics/2019/05/hyperledger-launches-hyperledger-iroha-1-0/. [Zugriff am 25 Dezember 2019].