IP6: Blockchain Transactionmanager

Bachelorthesis

Faustina Bruno, Jurij Maïkoff

Studiengang:
- iCompetence
- Informatik

Betreuer:
- Markus Knecht
- Daniel Kröni

Auftraggeber:
Fachhochschule Nordwestschweiz
FHNW Campus Brugg-Windisch
Bahnhofstrasse 6
5210 Windisch

2019-10-01

Abstract

Contrary to popular belief, Lorem Ipsum is not simply random text. It has roots in a piece of classical Latin literature from 45 BC, making it over 2000 years old. Richard McClintock, a Latin professor at Hampden-Sydney College in Virginia, looked up one of the more obscure Latin words, consectetur, from a Lorem Ipsum passage, and going through the cites of the word in classical literature, discovered the undoubtable source. Lorem Ipsum comes from sections 1.10.32 and 1.10.33 of "de Finibus Bonorum et Malorum" (The Extremes of Good and Evil) by Cicero, written in 45 BC. This book is a treatise on the theory of ethics, very popular during the Renaissance. The first line of Lorem Ipsum, "Lorem ipsum dolor sit amet..", comes from a line in section 1.10.32.

Inhaltsverzeichnis

1 Einleitung

Dieses Kapitel liefert eine ausführliche Zusammenfassung der Bachelorthesis.

1.1 Problemstellung

Die Aufgabe beinhaltet ein Blockchain Netzwerk [1] für die Fachhochschule Nordwest Schweiz[2] (FHNW) zur verfügung zu stellen, welches von den Studierenden zu test Zwecken genutzt werden kann. Blockchains verfügen über verschiedene Mechanismen, um sich gegen Attacken abzusichern. Eine davon ist eine Gebühr auf jeder Transaktion, der sogenannte Gas Price 2.2.7 [3]. Dadurch können Denial of Service (DoS) Attacken 2.2.8 [4], bei denen das Netzwerk mit unzähligen Transaktionen geflutet wird, effizient bekämpft werden. Der Angreifer kann die Attacke nicht aufrecht erhalten, da ihm die finanziellen Mittel zwangsläufig ausgehen. Obwohl dieser Schutzmechanismus auf einer öffentlichen Blockchain sehr effizient und elegant ist, eignet er sich nicht für eine Lernumgebung. Hier sollen Anwender die Möglichkeit haben, Transaktionen ohne anfallende Gebühren ausführen zu können. Dadurch wird jedoch die Blockchain anfällig für DoS Attacken.

1.2 Ziel

Das Ziel der Arbeit ist es ein Test Blockchain Netzwerk aufzubauen, welches für eine definierte Gruppe von Benutzern gratis Transaktionen erlaubt und trotzdem ein Schutzmechanismus gegen Dos Attacken hat.

1.3 Methodik

Hier wird beschrieben wie und was gemacht wurde

!!muss besprochen überarbeitet werden Wir haben zu Beginn Meilensteine und grössere Arbeitspakete definiert. Die kleineren Arbeitspakete wurden nach neugewonnen Wissen und Arbeitsstand definiert.

Strukturierung des Berichts

Der Bericht ist in einen theoretischen und praktischen Teil gegliedert. Gemachte Literaturstudiuen, geprüfte Tools und der aktuelle Stand der Ethereum Blockchain werden im theoretischen Teil behandelt. Im praktischen Teil wird beschrieben, wie das gewonnene Wissen umgesetzt wird. Es wird auf die Implementierte Lösung und deren Vor- und Nachteile eingegangen. Geprüfte Alternativen und deren Argumente sind ebenfalls enthalten. Das Fazit bildet den Abschluss des eigentlichen Berichts. Im Anhang ist eine Beschreibung der Entwicklungsumgebung und verwendeter Code zu finden.

2 Theoretische Grundlagen

Dieses Kapitel befasst sich nebst dem Kontext der Arbeit, mit den gemachteten Literaturrecherchen, welche für die Erarbeitung der Lösungsansätze nötig sind. Weiter wird der Anwendungsbereich der Lösung behandelt.

2.1 Anwendungsbereich

Die FHNW möchte zu Ausbildungszwecken eine eigene Ethereum Blockchain betreiben. Die Blockchain soll die selbe Funktionalität wie die öffentliche Ethereum Blockchain vorweisen. Sie soll den Studenten die Möglichkeit bieten, in einer sicheren Umgebung Erfahrungen zu sammeln und Wissen zu gewinnen. Obwohl eine öffentliche Blockchain für jedermann frei zugänglich ist, sind fast alle Aktionen mit Kosten verbunden. Die Kosten sind ein fixer Bestandteil einer Blockchain. So fallen zum Beispiel bei jeder Transaktionen Gebühren an. Diese ermöglichen nicht nur deren Verarbeitung, sondern garantieren auch Schutz vor Attacken.

Im Gegensatz zu einer öffentlichen Blockchain, sind Transaktionsgebühren in einer Lernumgebung nicht praktikabel. Die Studenten sollen gratis mit der Blockchain agieren können, ohne dass der Betrieb oder die Sicherheit der Blockchain kompromitiert werden.
Die FHNW bietet die kostenlose Verarbeitung von Transaktionen zu Verfügung. Damit sichert sie den Betrieb der Blockchain. Die Implementation von gratis Transaktionen und einem Schutzmechanismus wird in diesem Bericht behandelt.

2.2 Komponenten

Die folgenden Abschnitte behandeln die gemachten Literaturrecherchen. Für jedes Thema sind die gewonnen Erkenntnisse aufgeführt. Dabei ist nebst einem grundsätzlichen Verständnis für die Materie immer der Schutz vor einer Denial of Servie (DOS) Attacke im Fokus.

2.2.1 Ethereum Blockchain

Eine Blockchain ist eine kontinuierlich erweiterbare Liste von Datensätzen, „Blöcke“ genannt, die mittels kryptographischer Verfahren miteinander verkettet sind. Jeder Block enthält dabei typischerweise einen kryptographisch sicheren Hash (Streuwert) des vorhergehenden Blocks, einen Zeitstempel und Transaktionsdaten.[1] Im Gegensatz zur Bitcoin[5] kann bei Ethereum auch Code in der Chain gespeichert werden, sogenannte Smart Contracts /ref{sec_smartContracts} Die Ethereum Blockchain[6] hat eine eigene Kryptowährung, den Ether (ETH).

2.2.2 Smart Contracts

2.2.3 Identitäten

Um mit Ethereum interagieren zu können, wird eine Identität benötigt. Diese besteht aus einem öffentlichen und einem geheimen Schlüssel.

2.2.3.1 Geheimer Schlüssel

2.2.3.2 Öffentlicher Schlüssel

2.2.3.3 Adresse

2.2.4 Wallets und Smart Wallets

2.2.5 Transaktionen

2.2.6 Mining

2.2.7 Gas

2.2.8 Denial of Service (DOS) Attacken

2.2.9 Algorithmen

2.3 Lösungsansätze

//TODO Spellcheck über ganze Seite

2.3.1 Lösungsansatz 1: Super Smart Wallet

//TODO Bild

2.3.1.1 Hauptlösungsansätze

Es wird eine komplett eigene Smart Wallet erstellt, die sowohl die Whitelist wie auch den Schutzalgorithmus verwaltet. Jeder Benutzer erhält eine eigene Smart Wallet die von den Admins deployt wird. Auf einer Whitelist sind alle Benutzer aufgelistet die berechtigt sind gratis Transaktionen durchzuführen. Diese Liste wird von den Admins gepfleft. Der Sicherheitsalgorithmus prüft ob der Benutzer die Gratistransaktions Richtlinien verletzt. Falls der Benutzer die Sicherheitsrichtlinien verletzt, wird er vom Algorithmus aus der Whitelist gelöscht. Der Benutzer gelangt nur wieder in die Whitelist, wenn ein Admin ihn hinzufügt.

2.3.1.2 Pro

- Alles auf der Blockchain
- Dezentral
- Elegant
- Komplett eigene Lösung
- Ein System

2.3.1.3 Contra

- Machbarkeit unsicher
- Komplex
- Schutz Algorithmus kann nicht angepasst werden ohne Blockchain zu rebooten

2.3.1.4 Prozessworkflow

//TODO Bild

2.3.2 Lösungsansatz 2: Smart Wallet mit externen JavaProgramm nach Whitelist-Check

//TODO Bild

2.3.2.1 Hauptlösungsansätze

Es wird eine komplett eigenen Smart Wallet erstellt, die die Whitelist verwaltet. Zusätzlich wird ein Java Programm entwickelt, der den Schutzalgorithmus beinhaltet. Jeder Benutzer erhält eine eigene Smart Wallet die von den Admins deployt wird. Auf einer Whitelist sind alle Benutzer aufgelistet die berechtigt sind gratis Transaktionen durchzuführen. Diese Liste wird von den Admins gepfleft. Der externe Sicherheitsalgorithmus prüft nach dem Whitelist-Check ob der Benutzer die Gratistransaktion Richtlinien verletzt. Falls der Benutzer die Sicherheitsrichtlinien verletzt, wird er vom Algorithmus aus der Whitelist gelöscht. Der Benutzer gelangt nur wieder in die Whitelist, wenn ein Admin ihn hinzufügt.

2.3.2.2 Pro

- Sicher machbar
- Komplett eigene Lösung
- Algorithmus kann angepasst werden ohne Blockchain zu rebooten

2.3.2.3 Contra

- Mehrere Komponenten
- Zentrale Autorität

2.3.2.4 Prozessworkflow

//TODO Bild

2.3.3 Lösungsansatz 3: Smart Wallet mit externen JavaProgramm vor Whitelist-Check

//TODO Bild

2.3.3.1 Hauptlösungsansätze

Es wird eine komplett eigenen Smart Wallet erstellt, die die Whitelist verwaltet. Zusätzlich wird ein Java Programm entwickelt, der den Schutzalgorithmus beinhaltet. Jeder Benutzer erhält eine eigene Smart Wallet die von den Admins deployt wird. Auf einer Whitelist sind alle Benutzer aufgelistet die berechtigt sind gratis Transaktionen durchzuführen. Diese Liste wird von den Admins gepfleft. Der

externe Sicherheitsalgorithmus prüft vor dem Whitelist-Check ob der Benutzer die Gratistransaktion Richtlinien verletzt. Falls der Benutzer die Sicherheitsrichtlinien verletzt, wird er vom Algorithmus aus der Whitelist gelöscht. Der Benutzer gelangt nur wieder in die Whitelist, wenn ein Admin ihn hinzufügt.

2.3.3.2 Pro

- Komplett eigene Lösung
- Algorithmus kann angepasst werden ohne Blockchain zu rebooten
- DOS Algorithmus blockt bevor Transaktion auf SmartWallet trifft

2.3.3.3 Contra

- Mehrere Systeme
- Zentrale Autorität

2.3.3.4 Prozessworkflow

//TODO Bild

2.3.4 Lösungsansatz 4: Zentrale Smart Wallet für jeden Benutzer

//TODO Bild

2.3.4.1 Hauptlösungsansätze

Es wird eine komplett eigene Smart Wallet erstellt, die sowohl die Whitelist wie auch den Schutzalgorithmus verwaltet. Alle Benutzer erhalten eine zentrale Smart Wallet die von den Admins deployt wird. Auf einer Whitelist sind alle Benutzer aufgelistet die berechtigt sind gratis Transaktionen durchzuführen. Diese Liste wird von den Admins gepfleft. Der Sicherheitsalgorithmus prüft ob der Benutzer die Gratistransaktions Richtlinien verletzt. Falls der Benutzer die Sicherheitsrichtlinien verletzt, wird er vom Algorithmus aus der Whitelist gelöscht. Der Benutzer gelangt nur wieder in die Whitelist, wenn ein Admin ihn hinzufügt.

2.3.4.2 Pro

- Nur eine Smart Wallet muss deployed und betrieben werden
- Weniger administrativer Aufwand für Admins

2.3.4.3 Contra

-Schwierig Sender ID für Transaktion zu setzten (überhaupt möglich?)

2.3.4.4 Prozessworkflow

//TODO Bild

2.4 Evaluation der Lösungsansätze

2.4.1 Lösungsansatz 1: Super Smart Wallet

//TODO Evaluation

2.4.2 Lösungsansatz 2: Smart Wallet mit externen JavaProgramm nach Whitelist-Check

//TODO Evaluation

2.4.3 Lösungsansatz 3: Smart Wallet mit externen JavaProgramm vor Whitelist-Check

//TODO Evaluation

2.4.4 Lösungsansatz 4: Zentrale Smart Wallet für jeden Benutzer

//TODO Evaluation

3 Praktischer Teil

4 Fazit

5 Quellenverzeichnis

[1] „Blockchain - Wikipedia“, 2019. [Online]. Verfügbar unter: https://en.wikipedia.org/wiki/Blockchain.

[2] „University of Applied Sciences and Arts Northwestern Switzerland“, 2019. [Online]. Verfügbar unter: https://www.fhnw.ch/.

[3] M. Inc., „What is Gas | MyEtherWallet Knowledge Base“, 2018. [Online]. Verfügbar unter: https://kb.myetherwallet.com/en/transactions/what-is-gas/.

[4] „Denial-of-service attack - Wikipedia“, 2019. [Online]. Verfügbar unter: https://en.wikipedia.org/wiki/Denial-of-service_attack.

[5] „Bitcoin - Wikipedia“, 2019. [Online]. Verfügbar unter: https://en.wikipedia.org/wiki/Bitcoin.

[6] Ethereum, „Home | Ethereum“, 2019. [Online]. Verfügbar unter: https://www.ethereum.org/.

[7] go-ehereum, „Go Ethereum“, 2019. [Online]. Verfügbar unter: https://geth.ethereum.org/.

[8] P. Technologies, „Blockchain Infrastructure for the Decentralised Web | Parity Technologies“, 2019. [Online]. Verfügbar unter: https://www.parity.io.

[9] „https://github.com/ethereum/aleth“, 2019. [Online]. Verfügbar unter: https://github.com/ethereum/aleth.

[10] T. B. G. 2019, „Sweet Tools for Smart Contracts“, 2019. [Online]. Verfügbar unter: https://www.trufflesuite.com/.

[11] uPort, „uPort“, 2019. [Online]. Verfügbar unter: https://www.uport.me/.

[12] MetaMask, „MetaMask“, 2019. [Online]. Verfügbar unter: https://metamask.io/.

[13] A. Wallet, „Atomic Cryptocurrency Wallet“, 2019. [Online]. Verfügbar unter: https://atomicwallet.io/.

[14] E. M. Inc., „Crypte Wallet - Send, Receive & Exchange Cryptocurrency | Exodus“, 2019. [Online]. Verfügbar unter: https://www.exodus.io.

[15] MyEtherWallet, „MyEtherWallet | MEW“, 2019. [Online]. Verfügbar unter: https://www.myetherwallet.com/.

[16] Solidity, „Solidity - Solidity 0.5.11 documentation“, 2019. [Online]. Verfügbar unter: https://solidity.readthedocs.io/en/v0.5.11/.

[17] „Vyper–Vyper documentation“, 2019. [Online]. Verfügbar unter: https://vyper.readthedocs.io/en/v0.1.0-beta.13/#.

6 Anhang

6.1 Glossar

Begriff	Bedeutung

6.2 Entwicklungsumgebung

In diesem Abschnitt wird die geplante Testumgebung und deren Verwendung beschrieben.

6.2.1 Blockchain

Es wird eine Test-Blockchain aufgesetzt. Diese wird benötigt, um geschriebenen Code zu testen und analysieren.

Als Blockchain wird Ethereum[6] verwendet. In den nachfolgenden Absätzen werden mögliche Tools besprochen, die für den Aufbau von einer Testumgebung genutzt werden können.

6.2.1.1 Client

In der Arbeit wird evaluiert ob Geth[7] als Client den Ansprüchen genügt oder ob ein anderer Client (z.B. Parity[8], Aleth[9], etc.) zum Einsatz kommt.

Trufflesuite Trufflesuite[10] wird verwendet, um eine simulierte Blockchain aufzusetzen. Diese kann für die Einarbeitung in die Materie genutzt werden.

6.2.2 Wallet

Wallets werden für die Verwaltung von Benutzerkonten und deren Transaktionen benötigt. Zu den möglichen Wallets gehören z.B.:

- uPort[11]
- Metamask[12]
- Atomic Wallet [13]
- Exodus[14]

Es wird davon ausgegangen, dass keine Wallet alle Bedürfnisse abdecken kann, daher wird die gewählte Wallet im Zuge dieses Projekts erweitert. Für Ethereum existiert ein offizieller Service um eine eigene Wallet zu ertellen: MyEtherWallet[15]

6.2.3 Smart Contracts

Smart Contracts werden benötigt, um zu bestimmen, wer auf einer Blockchain gratis Transaktionen ausführen kann. Sobald eigene Smart Contracts entwickelt werden, kann die Testumgebung genutzt werden, um diese zu testen.

6.2.3.1 Programmiersprache

Für die Entwicklung von Smart Contracts werden folgende zwei Sprachen evaluiert:

- Solidity[16]
- Vyper[17]

7 Ehrlickeitserklärung

Die eingereichte Arbeit ist das Resultat unserer persönlichen, selbstständigen Beschäftigung mit dem Thema. Alle wörtlichen und sinngemässen Übernahmen aus anderen Werken sind als solche gekennzeichnet

Datum _____

Ort _____

Faustina Bruno _____

Serge Jurij Maïkoff _____