TALLINN UNIVERSITY OF TECHNOLOGY
Faculty of Information Technology
Department of Computer Engineering

ITC70LT

Christopher David Raastad

SECURITY ANALYSIS OF THE EURO 2.0 DIGITAL CURRENCY PROTOTYPE

Master thesis

Alexander Horst Norta
PhD
Associated Professor

Tallinn, Estonia 2017

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Arvutitehnika instituut

ITC70LT

Christopher David Raastad

EURO 2.0 DIGITAALSE VALUUTA PROTOTÜÜBI TURVALISUSANALÜÜS

Magister

Alexander Horst Norta
PhD
Dotsent

Tallinn, Estonia 2017

Author's declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Christopher David Raastad

May 18th 2017

Abstract

Digital money in the form of cards, bank accounts, and online payments has been around for the last few decades. The cryptocurrency Bitcoin emerged as a completely new paradigm of storing and transferring value outside of the regulations and governments of the traditional financial system. The blockchain distributed ledger consensus technology powering Bitcoin inspired similar cryptocurrency clones, specialized use cases, and the creation of a general purpose distributed application computing platform Ethereum. Despite the innovations in value transfer, cryptocurrencies have a very niche use in daily transactions outside the circle of enthusiasts. The problem is the lack of incentive for users and merchants alike to hold all of their assets in a cryptocurrency. Significant cost and friction is associated with moving funds in and out of cryptocurrencies through exchanges and payment processors as well as the risk of wildly fluctuating exchange rates. The Euro 2.0 Foundation presents a fiat backed, fully regulated, government backed digital currency to alleviate these shortcomings. This thesis presents a security analysis of the proposed Euro 2.0 digital currency prototype. We present the features and architecture of the Euro 2.0 digital currency, a fusion of traditional client server components, Estonia ID card authentication, and Ethereum smart contract decentralized distributed computing platform. Then we present a change analysis of the impact of the Euro 2.0 digital on the stakeholders including users, merchants, banks, and government. Finally we present an information system security risk analysis based on the OCTAVE Allegro methodology of key information assets user money, Ethereum admin keys, and user identity. From the security analysis we present suggestions and mitigation strategies of the biggest risks of a hybrid centralized and decentralized Euro 2.0 digital currency system.

The thesis is in English and contains 112 pages of text, 6 chapters, 3 figures, 75 tables.

Annotatsioon

Digitaalne raha on kaartide, pangakontode ja internetimaksete kujul olemas olnud juba aastakümneid. Krüptoraha Bitcoin kujundas täiesti uue paradigma väärtuste hoidmisele ja ülekannete tegemisele väljaspool traditsiooniliste finantssüsteemide regulatsioone ning valitsust. Plokiahela abil loodi ühtse tehnoloogiaga avalik raamatupidamisregister, olles inspireeritud sarnaste krüptorahade kloonidest, erilistest kasutamisjuhtudest ja üldotstarbelise rakenduste arvutamise platvormi Ethereum loomisest. Vaatamata väärtuste ülekannete protsesside innovatsioonile on krüptoraha väljaspool entusiastide ringe igapäevakasutuses siiski niši staatuses. Probleem tuleneb sellest, et kasutajatel ning kaupmeestel puuduvad stiimulid kõiki oma varasid krüptovaluutas hoida. Lisaks valuutakursside suurtele kõikumistele seostatakse krüptoraha märkimisväärseid kulusid ja huvide vastuolusid raha liigutamisega krüptovaluutasse ja tagasi valuutavahetuste ja maksete töötlemise protsesside käigus. Euro 2.0 Sihtasutus esitleb täielikult reguleeritud, ametlike volitustega ja valitsuse poolt toetatud digitaalvaluutat, millega soovitakse kõiki neid puudusi leevendada. Käesolev magistritöö esitab kavandatava Euro 2.0 digitaalvaluuta prototüübi turvalisusanalüüsi. Töös on välja toodud Euro 2.0 digitaalvaluuta omadused ja ülesehitus, traditsiooniliste kliendiserverite lülide, Eesti ID-kaardi autentimise ning Ethereum targa lepingu detsentraliseeritud hajusandmetöötluse platvormi integratsioon. Järgmisena on esitatud analüüs Euro 2.0 digitaalvaluuta mõjust huvigruppidele, hõlmates kasutajaid, kaupmehi, panku ja valitsust. Lõpetuseks tuuakse välja infosüsteemi turvariskid toetudes OCTAVE Allegro kasutajate varade olulise teabe metoodikale, Ethereumi haldajate võtmete ja kasutajate identideedi analüüsile. Turvalisusanalüüsi põhjal esitatakse soovitused ja leevendamise strateegiad hübriid-tsentraliseeritud ja detsentraliseeritud Euro 2.0 digitaalvaluutasüsteemi suurimatele riskidele.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 112 leheküljel, 6 peatükki, 3 joonist, 75 tabelit.

Table of abbreviations and terms

AML	*Anti Money Laundering*, processes implemented by financial institutions to hinder money laundering activities and comply with regulations
API	*Application Programming Interface*, instructions for programmatically communicating with servers
BTC	*Bitcoin* cryptocurrency abbreviation
CMB	*Citizenship and Migration Board* of Estonia
CTF	*Counter Terrorist Financing*, processes implemented by financial institutions to hinder terrorist financing and comply with regulations
ETC	*Ether*, Ethereum cryptocurrency abbreviation
ISSRM	*Information Systems Security Risk Management*
JSON	*JavaScript Object Notation*, a serialization format for REST API communication
KYC	*Know Your Customer*, proccesses implemented by financial institution to identify and verify customer identities to aid in AML and CTF as well as comply with regulation
LDAP	*Lightweight Directory Access Protocol*, particular that hosted by SK for Estonian PICs
PIC	*Personal Identity Code*, issued by the Estonian CMB associated with an Estonian ID card
PKI	*Public Key Infrastructure*, used in cryptocurrency cryptography
REST	*REpresentational State Transfer*, a web server communication architecture
RPC	*Remote Precedure Call*, external function call from application
RSA	widely used public-key crypto-system for secure data transmission
SK	*Sertifitseerimiskeskus* (Certification Center), SK ID Solutions AS, the only Certificate Authority (CA) in Estonia for the Estonian ID card system
TLS	*Transport Layer Security*, cryptographic protocol to provide secure communication over a computer network
UX	*User Experience*

Contents

List of Figures

List of Tables

1. Introduction

1.1. The Road to Digital Currency

Payments today are the laggard of the information age. While emails can be sent instantly for free anywhere in the world, money is either slow and/or expensive to move digitally. Bank transfers can take days, only work during business hours, and have high fees across borders. Card payments are instant, but subject merchants to high fees and chargeback risks. Paypal brought convenient payments to the web, but at a high cost to accept payments and move funds internationally. Fintech companies like Venmo and Square can create the illusion of fast payments, but still take days to settle in the background with the potential chargeback risks. This cost comes from the highly regulated centralized financial payments systems with little economic incentive to reduce fees. A usable digital currency would greatly alleviate this friction in transferring value that theoretically costs the economy an estimated 1% of GDP annually[43].

Bitcoin was the first successful implementation of a decentralized digital currency. Nakamoto's peer to peer digital cash system completely sidestepped the existing financial system[51]. Its clever proof of work consensus protocol, economic incentives for nodes to maintain the network, pseudo anonymity, and irreversible transactions implemented on the public ledger blockchain technology has been called "the next technological revolution"[61]. Hundreds of Altcoins followed Bitcoin's lead borrowing from the blockchain technology to implement other flavors of cryptocurrency and digital asset management[25]. Despite the rapid growth of the cryptocurrency ecosystem, everyday payments are a niche use case for digital currencies, which are more commonly used for investment and financial speculation[41]. Merchants accepting Bitcoin payments are in fact immediately converting funds to a fiat currency for a non negligible fee. Bitcoin exchanges are high cost gatekeepers between the cryptocurrency and financial systems.

What is missing today are incentives for parties on both sides of transactions to hold assets end to end in digital currency. The cost and friction of moving between two financial systems, crypto and fiat, eliminates the perceived benefits of cryptocurrencies. The Euro 2.0 project aims to implement fiat currency on digital currency technology in order to reduce payment friction by eliminating unnecessary financial intermediaries. The unanimous adoption of Estonian ID card in Estonia and the rise of the Ethereum distributed application platform make a usable digital currency prototype possibly.

Can a regulated, government backed, digital currency system be securely built? So far, no academic literature has explored the security of real money moving through cryptocurrency pipes. This thesis introduces the Euro 2.0 digital currency prototype, its

impact on stakeholders in the finance, and assesses the security risks of the proposed implementation to determine whether this prototype can in fact reasonably support a national digital currency.

1.2. Research Questions

This thesis analyzes the feasibility of the proposed Euro 2.0 digital currency by exploring the following research question:

> **RQ:** *How do we assure that the proposed Euro 2.0 digital currency system does not have any major security flaws compromising its usefulness as a monetary system?*

In which we explore in sub research questions:

> **RQ1:** *How do we describe the features and components of the Euro 2.0 digital currency system for a security analysis?*
>
> **RQ2:** *How does adoption of Euro 2.0 digital currency impact the relationship with money for key stakeholders: users, merchants, banks, and governments?*
>
> **RQ3:** *How do we assess the security risks of Euro 2.0 digital currency system and potential impact on stakeholders?*

The result of exploring **RQ** will result in one of the following outcomes:

- The Euro 2.0 system is free from significant risks in functioning as a digital monetary system.
- The Euro 2.0 system has one or more serious risks that need to be addressed before significant adoption.

In order to do a risk analysis, the key features of the Euro 2.0 system need to be identified and implementation details described by answering **RQ1**. Following a clear description of the system, **RQ2** explores the current and future relationship with money of key stakeholders after adoption of Euro 2.0. This exploration of the stakeholders' relationship with money guides the security analysis to asses the impact of realized risks discovered in **RQ3**. The severity and likelihood of realized risks to the monetary system lead to an answer to the original research question **RQ**.

1.3. Research Methods

The thesis utilizes methods from design science and Information Systems Security Risk Management (ISSRM) to approach the research questions. The main contributions are:

- Presentation of the features and components of the proposed Euro 2.0 digital currency prototype
- Change analysis of stakeholders adopting the Euro 2.0 digital currency
- Security Risk analysis of the Euro 2.0 prototype
- Design recommendations for Euro 2.0

The features and components are presented by a comprehensive review of the Euro 2.0 codebase (Appendix A.1). The change analysis is based on an interpretation of the Change Formula[22] described in Section 4.2. The security risk analysis employs the OCTAVE Allegro risk analysis framework presented in Section 2.6 in which design recommendations follow from the analysis.

1.4. Structure of the Thesis

Chapter 2 - Bridge of Knowledge will give an overview of the existing body of knowledge needed to understand the remaining chapters including traditional finance, Bitcoin, Tether, Ethereum, Estonia ID card, and ISSRM frameworks. Chapter 3 - Euro 2.0 Digital Currency Prototype will describe the motivation, features, smart contracts, and implementation architecture of the Euro 2.0 digital currency prototype needed for a security analysis. Chapter 4 - Euro 2.0 Change Analysis will perform a change analysis to better understand the socio-technical impact of stakeholders adopting the Euro 2.0 digital currency. In Chapter 5 - Euro 2.0 Security Risk Analysis we will perform a OCTAVE Allegro information security risk analysis of the Euro 2.0 system resulting in mitigations proposals and design recommendations. Finally Chapter 6 - Summary will summarize the thesis, answer the research question, and suggest future avenues of research.

2. Bridge of Knowledge

2.1. Introduction

In this chapter we present background research needed to understand the rest of the thesis in Chapters 3, 4, and 5. Section 2.2 presents the costs associated with the currency and payments. Section 2.3 presents Bitcoin, Tether, and research analyzing trust in payments needed to understand the motivation behind Euro 2.0. Section 2.4 presents an overview of Ethereum including accounts, messages, execution model, and the high level programming language Solidity. Section 2.5 presents background on the Estonian ID card system. Then in Section 2.6 we present findings on ISSRM frameworks, security risk patterns of distributed systems, and our decision to choose OCTAVE Allegro as a risk analysis framework for this thesis.

2.2. Currency and Payments

Money and payments haven been part of human society since the dawn of civilization. Currency grew out of the need to transport value of goods without transferring the goods themselves. First came gold, silver, and other precious metals to trade in exchange for goods and services. Later in the 1600s, notes were issued by the Bank of England backed by silver and the practice spread to every country in the Western world. Banks became the de facto institutions to store and transfer value in government issued currencies. By the 1950s the USA was the first government to eliminate physical backing (silver) to create a fiat currency only backed by the trust of the US Government and Federal Reserve. Today there are no countries left physically backing government issued notes; money is entirely trust that society accepts the government backed notes for exchange of goods and service.

As early as the 1960s banks were some of the first adopters of information technology, using computers and databases to overhaul paper processes. The first electronic access of money was the Credit Card, its earliest usage was a credit exchange between private companies in the USA in the 1920s, Diners' Club, Inc. for consumers in 1950, American Express Travel and Entertainment card in 1958, and finally the founding of VISA in 1976 to spread the concept globally[30]. Debit cards directly debiting bank accounts hit market as early as 1966 in the USA, long before mass consumer adoption of technology, gaining popularity in the 80s and 90s with the rise of ATM network and merchant acceptance[26]. With all of these developments, the Financial institutions of banks and payment processors are the heart of system and source of high fees. Credit Cards cost merchants 1-3% transaction fees and carry the risk of costly chargebacks at the benefit of

consumer convenience. Debit cards are slightly better, charging about €0.45 per swipe [44].

With consumer adoption of the internet came personal windows into our finances via online banking. Bank transfers can take days to settle and only work during working hours of weekdays. International bank transfers can take even longer and with a heavy 3-5% fee. Many transactions in financial systems, such as stock trades, can happen "instantly" but in fact take days to settle on the backend due to legacy paper based processes. The financial industry and has only in this time created pretty facades to their inefficient processes. Paypal succeeding to bring payments to the internet, but is still atrociously expensive to send and receive card payments, charging 2.9% + $0.30 USD per transaction and adding a 2.5% fee on top of a bank dictated currency conversion spread for international conversion of funds[52].

Why all the friction? In France, the economic cost of payments is an estimated at 1% of GDP annually [43]. Elsewhere an estimation was not found, but we can assume this tax on the economy is non-trivial since financial institutions designed themselves to profit from friction in payments. International money transfer moves between four or more levels of banks, each taking a small slice of the fee, until funds reach the destination bank. Consumer banking creates the illusion of free domestic transfers though subsidization, but makes the meat of their profit from transactions data to sell customers credit and loan. Institutional investment banks gamble with our money every day on Wall Street, when things get out of hand, we have the Financial meltdown of 2008. Even post financial crisis, too big to fail banks have barely changed there ways. With increased regulatory costs and greedy shareholders comes even less economical incentive to reduce the cost of payments.

2.3. Cryptocurrencies and Trust

2.3.1. Bitcoin

Following the 2008 crisis, Satoshi Nakamoto quietly released *Bitcoin: A Peer-to-Peer Electronic Cash System*[51] aiming to solve the business problem *"How do I create a system where nobody can stop me spending my own money?"*[18]. A decentralized system *"based on cryptographic proof instead of trust"* allows parties to transact with each other directly without a trusted third party. Double spending is prevented by a *"peer-to-peer distributed timestamp server to generate computational proof of the chronological order of transactions"*[51]. This computational proof is distributed in the amongst participant nodes in the network in a voting scheme governed by a cryptographic puzzle known as *proof-of-work*. Nodes compete to publish a block of transactions in the system for a

reward of Bitcoins and transaction fees. The proof of work mechanism probabilistically limits the network to publish a new block of transactions approximately every ten minutes. Blocks reference previous blocks and once published are irrevocable unless another node were to rewrite history by solving the cryptographic puzzle multiple times in a short timespan, which is computationally infeasible. Hence transactions on the Bitcoin network are irreversible. This chain of blocks of transactions cryptographically tied in a historical record is the *blockchain*. Nodes are incentivized to act in their best interest and maintain the network to claim their reward, currently worth 12.5 BTC per block and more than 21,000 USD at the time of this writing[47], a process commonly known as *mining*. A published block has to be valid in history, i.e. no double spent bitcoins and a valid proof-of-work solution, to be accepted by the network. With this consensus mechanism, the Bitcoin peer-to-peer network maintains itself without a centralized entity[51].

Users of the bitcoin network enjoy the possibility of pseudo-anonymity in using the system. Transactions are a chain of digital signatures. Public keys are the addresses which can be thought of accounts. Private keys are the proof of ownership of a public key which allow the right to claim ownership and spend all coins previously sent to that address. A user only requires a private and public key tied to some unspent tokens in order to use the system[51]. All transaction hashes and public keys addresses are published in blocks on the blockchain *public ledger*. Anyone can view produced blocks and transactions by downloading a Bitcoin client or browsing an online tool like Blockchain Info[57]. Pseudo-anonymity comes from the fact once the a Bitcoin address is linked to a real identity, all present, past, and future transactions are visible on the public ledger. Graph analysis of the the entire Bitcoin blockchain can reveal clues of transaction patterns, owners of addresses, and networks of miners[56].

Bitcoins counteracts inflation of traditional monetary systems with a limited supply, designed to emulate a growth curve similar to gold. This supply is algorithmically limited to 21,000,000 BTC, which will be all mined into circulation around year 2140. In Theory, Bitcoin will become more and more valuable overtime because of its rarity, hence an attractive mechanism for long term savings.

Bitcoin transactions are geographically independent. Since the entire global network runs distributively on the internet and not controlled by a central party, transactions can happen anywhere a client has enough internet bandwidth to broadcast to the network. Theoretically Bitcoin is a completely governmental independent store and growth of value, but its price does fluctuate with geopolitical events in China, USA, and UK (Brexit).

Currently Bitcoin has a major benefit low cost transactions, since miners are incentivized enough by Bitcoin rewards mining new blocks of transactions. When Bitcoin supply approaches its upper bound, already 99% of Bitcoins will be mined by 2036, then

transaction fees must take over as incentive for miners to maintain the network.

2.3.2. Tether

Tether is the project most resembling the Euro 2.0 digital currency system by supporting *"fiat currencies on the Bitcoin blockchain."*[46]. The Tether coin is built on top of the Omni Protocol, which coins to be created by embedding data in Bitcoin transactions[36]. The Hong Kong based company Tether Limited holds a reserve of Fiat currency, currently USD and EUR, that can be converted to and from Tethered TUSD and TEUR on Bitcoin blockchain. TUSD and TEUR can be transferred, stored, and spent like Bitcoin. Tether Limited maintains a proof of reserve that at any given time the balance of fiat currency held in the reserves will be equal to (or greater than) the number of tethers in circulation. Tethers in circulation exist as a decentralized digital currency but reserve assets must be administered by centralized Tether Limited. Tether transacts just like Bitcoin, hence, is subject to the same trust and regulatory issues outlined in the next section.

2.3.3. Anonymity, Regulation, and Trust

Bitcoin gained early attention for the infamous Silk Road, the "Ebay for Drugs"[14]. The website amassed over 1000 vendors, tens of millions of dollars of revenue, and triple digit yearly growth until it was shutdown by the FBI in 2013 after two and a half years operation[4]. The buyers were protected in escrow Bitcoin transaction, only releasing payment to the seller if the buyer was completely satisfied. Bitcoin allowed payments amongst parties that can never trust each other in real life. *"Anonymity and lack of regulation which is meant to free users from central authorities also empowers drug dealers and money launderers"*[5].

Sas and Khairuddin interviewed 20 Bitcoin users to explore the challenges and opportunities of Bitcoin users with respect to technological, social, and institutional trust[58]. The economic rationale of users holding Bitcoin came from distrust in governments and banks directly having access to bank account funds, fear of unfavorable fiat currency exchange rates due to inflation, economic downturn, or political events, and speculation that Bitcoin will become more and more valuable over time. Bitcoin users are not discouraged by the early reputation of the currency for drugs and money laundering. Ironically, they found *"spending bitcoins as a currency appears as an exception rather than a norm"*. Bitcoin behaves more like a commodity than a currency. Websites and stores accepting Bitcoin do so more for marketing then actual value of the currency. Merchants pay a 1% transaction fee to accept Bitcoin payment for fiat currency priced goods with payment provider to directly convert the Bitcoin to a fiat currency bank

account, showing little incentive to hold their Bitcoin. The short term fluctuations of value and difficult to grasp price with regards fiat currency makes it unattractive for end to end commerce in Bitcoin.

From the findings of Ali, Clarke, and McCorry *"Bitcoin's strengths and weaknesses both derive from the same essential ideological and architectural design choices"*[5]. Sas and Khairuddin support this statement with their findings of Bitcoin user experiences with regards to the blockchain's characteristics and their impact on trust [58]. The decentralized blockchain gives users a sense of honestly and credibility contrasted to sometimes dishonest central financial institutions. Users value transparency in the public ledger as well as easy, quick, and low cost transactions. Unregulated Bitcoin gives users a sense of empowerment, setting no limits to transaction frequency and size, and comfort in the impossibility of governmental intervening with financial matters in the system.

On the other hand, surveyed users showed concern over insecure transactions and risks associated with literally managing their own finances which were categorized by the authors as:

- *Risks Due to Users' Challenges of Handling Passwords* (or private keys)
- *Risks Due to Hackers' Malicious Attacks* to steal coins
- *Risks Due to Failure to Recover from Human Error or Malice*
- *Risks Related to Dishonest Partner of Transaction*

These risks are heightened by hackers benefiting from irreversibility of transactions, lack of regulation, and pseudo-anonymity. Also highlighted by Ali, Clarke, and McCorry, *"Denoting money as virtual assets to remove reliance on banks also opens the doors to hackers and malware."*[5]. The same factors increase the severity of losses due to user error, such as a forgotten password, wrong receiver account address, or incorrect amount in a transaction. Off chain components of transactions, such as the exchange of fiat currency for Bitcoin or buying goods and services, have no component on the blockchain and hence have possibility for fraud and deception.

The authors propose strategies for mitigating the risks of dishonest traders, the main aspect of user distrust in transacting with Bitcoin:

- *Trade with Authorized Exchanges*
- *Trade with Socially Authorized Traders*
- *Trade with Reputable Individual Traders*
- *Trade with De-anonymised Individual Traders*

- *Regulating Bitcoin*

These propositions to mitigate risks paradoxically nullify some key privacy and decentralized features of Bitcoin. Users are more likely to trust the counter-party when they are socially validated and/or identified and no longer anonymous, a type of user and community driven KYC to build trust. Exchanges add an interface to the traditional financial institution, forcing users to undergo traditional AML and KYC processes. Regulating Bitcoin completely goes against the original vision of Nakamoto, but could help users deal with dishonest traders and bring the security and institutional trust of regulated financial institutions, which is currently best served by exchanges.

The authors propose three general design implications for Bitcoin to *"address the trust challenges of dishonest traders while respecting blockchain's main characteristics"*[58]:

- *Support Transparency of Two-way Transactions* (on and off blockchain)
- *Create Tools for Materializing Trust in Blockchain*
- *Create Tools to Support Reversible Transactions*

The study highlights the trust needs of usable digital currency and motivates the creation of Euro 2.0 system.

2.4. Ethereum

Ethereum generalizes blockchain technology as a decentralized computing platform for building distributed applications. Ethereum is used to implement the decentralized component of Euro 2.0, providing the main account balance and state data and functionality described in Section 3.4. Section 2.4.1 gives an overview of the necessary details of the Ethereum platform and Section 2.4.2 outlines the key points of the Solidity Smart Contracts programming language used later in the thesis.

2.4.1. Ethereum Platform

The core of Ethereum is the *Ethereum Virtual Machine* (EVM), which functions as a decentralized general purpose computing platform with Turning complete scripting functionality. Ethereum is formally specified in Wood's *Ethereum: A Secure Decentralised Generalised Transaction Ledger*[63], also know as the "Yellowpaper" functioning as a "technical bible" of the Ethereum platform. The Yellowpaper, although very detailed and thorough, is too low level of a formal specification to be digestible as an introduction to the platform. Buterin's Ethereum Whitepaper, *A Next-Generation Smart Contract and Decentralized Application Platform*[19], is a high level introduction to the

platform suitable for understanding the key decentralized concepts of Euro 2.0. Anderson et al. compare Namecoin, Peercoin, and Ethereum to Bitcoin, looking at factors such as blockchain disk size and client bootstrapping security, and present the results of a crawler on the Ethereum blockchain looking at contract usages and Zombie contracts[6].

Accounts

Ethereum state is held in objects called *accounts*. An account is composed of the following fields:

- *nonce* - a counter used to make sure each transaction can only be processed once
- *balance* - amount of Ether in the account
- *contract code* - the code associated with this account, if present
- *storage* - storage space for this account

An Ethereum account is named with a 20 byte address. This address is the `Keccak-256` hash of a public key of a private generated from an Elliptic Curve Digital Signature Algorithm. Accounts come in two flavors: *Externally Controlled Accounts* and *Contract Accounts*. Externally controlled accounts are controlled by private keys of users outside of Ethereum and can be used to send messages by creating and signing transactions. Contract account code is executed upon a received messages to the address. This executed code can read and write to local storage, send other messages, and also create contracts. Both types of accounts store a balance of *Ether*, the currency of Ethereum, which can be used as a store of value and as fee for miners to execute EVM code. Ether is produced by miners as an output of producing blocks in a Proof-of-Work mechanism. Unlike bitcoin, Ether doesn't have a capped total supply, but is anticipated to be produced at the same rate of Ether being lost to unaccessible accounts where the private key is lost or unknown.

Messages and Transactions

Messages and transactions in Ethereum are the same construct. A *Transaction* refers to a signed message from an externally owned Ethereum account with the following fields:

- Recipient Ethereum address of the message
- Signature identifying the sender
- Amount of Ether transferred from the sender to the recipient
- Optional data field accessible by smart contract

- `STARTGAS` value in Ether, the maximum amount of fee a given transaction is allowed to execute
- `GASPRICE` value in Ether, the fee the sender pays per computational step

Transactions are persisted on the blockchain. Account code will be executed upon receipt of a message. Gas allows for a Turing complete scripting language by creating economical disincentive for accidental or intentional denial of service attacks through infinite loops or expensive computations. Each computational step has a fee in a multiple of `GASPRICE` and when greater or equal to transaction specified `STARTGAS` has been executed, the computation fails with all execution changes rollback. Blocks of transactions also have a dynamic gas limit to the total computational expense of an entire block of transactions.

Messages refer to those originating from a contract `CALL` opcode and not by an external actor. These are not persisted on the blockchain and are only virtual objects in the Ethereum execution environment of nodes on the network. Messages contain the following fields:

- Sender Ethereum address of the message
- Recipient Ethereum address of the message
- Amount of Ether to transfer with the message
- Optional data field
- `STARTGAS` value in Ether delegated to the message from calling contract original gas

Limited top level `STARTGAS` prevents contracts from calling in infinite recursive loops.

Ethereum State Transition Function

Unlike Bitcoin's unspent transaction output (*UTXO*) mechanism, Ethereum employs an account model of system state. The Ethereum state S represents the state of all addresses, balances, contracts, and data in the Ethereum. Given a transaction TX the new Ethereum state S' is computed by the state transaction function $APPLY(S, TX) -> S'$ and does the following according to the Ethereum Whitepaper[19]:

- Check if transaction is valid, signature is valid, and nonce matches sender account nonce; if not throw an error.
- Calculate transaction fee as `STARTGAS * GASPRICE`, determine the sending address from the signature, subtract the fee from the sender's account balance, and increments the sender's nonce. If there is not enough balance, throw an error.

- Initialize `GAS = STARTGAS`, and take off a certain quantity of gas per byte to pay for the bytes in the transaction.
- Transfer the transaction value from the sender's account to the receiving account. If the receiving account does not yet exist, it is created. If the receiving account is a contract, run the contract's code either to completion or until the execution runs out of gas.
- If the value transfer failed because the sender had insufficient balance or the code execution ran out of gas, revert all state changes except the payment of the fees, and add the fees to the miner's account.
- Otherwise, refund the fees for all remaining gas to the sender, and send the gas fees consumed to the miner.

State is calculated and stored identically on every node in the network for all transactions.

Ethereum Code

Ethereum code is fundamentally a low-level, stack-based, byte code language referred to as the "EVM code". Distributed applications in Ethereum are usually written in high level languages such as python based Serpent[21] or javascript based Solidity, which is described in Section 2.4.2, and then compiled to EVM bytecode. Code executes until an error is thrown, runs out of gas, or a `STOP` or `RETURN` statement is executed. Persistence capabilities include a last-in-first-out (LIFO) *stack* in which values can be pushed and popped, an infinitely expandable byte array *memory*, or contract *storage*, a key value store which persists after computation has finished (unlike the memory or stack). Executing code also has access to the value, sender, and data of an incoming message and can return a byte array of data as an output.

Events

Events are a space optimized, searchable, storage of predefined conditions in contracts described formally in the Yellowpaper[63]. This is analogous to logs in the traditional computing. These logs are not usable like contract storage in future Ethereum computations, but provide searchable records of the past usable in Ethereum applications.

Blockchain and Mining

Ethereum has a proof-of-work mining mechanism, like Bitcoin, to economically incentivize miner nodes to maintain the global state of Ethereum network for Ether

rewards. Miners compete to solve a cryptographic puzzle to produce valid blocks of transactions. The hash of the previous block is an input to a new block, hence forming a blockchain record of transactions. In the process of mining, the block validation process executes the EVM code for all contracts involved in transactions. This implies that all EVM code is being executed simultaneously by all nodes in the network and all contract data is public and shared on the network. Blocks are mined in Ethereum at an average rate of ten to fifteen seconds unlike, the rate of ten minutes of Bitcoin. The security of the network lies in the difficulty of producing valid blocks, with valid cryptographic proof-of-work puzzle, and valid state transition faster than new blocks of honest nodes.

2.4.2. Solidity Smart Contracts

Solidity is a high level programming languages to define Smart Contracts for the Ethereum Virtual Machines[32]. It has syntax is similar to Javascript, statically typed, supports inheritance, libraries, and many other features. A high level overview of a very small subset of features of Solidity needed to understand the smart contracts discussed in this thesis are described below.

Contract Structure

A minimal viable Solidity Smart contract is defined below in the `contract.sol` file below:

```
pragma solidity ^0.4.10;

contract ContractName {
        ...
}
```

`pragma` declares the version of solidity. `contract` indicates a contract with ContractName. Contract is synonymous to class in most object oriented programming languages. Inside the contract definition can be any number of *data* fields, *functions*, *modifiers*, and *events* described below.

Data

Data in Solidity smart contracts are like fields in other object oriented languages. They have the basic structure of `<type> <visibility> <name>`. Example types are listed below:

- `address` - a 20 byte value of an Ethereum address
- `uint256` - an unsigned 256 bit integer
- `bool` - boolean
- `bytes32` - fixed sized byte array of 32 bytes
- `_ArrayType[]` - represents a dynamically sized array of `_ArrayType`
- `mapping(_KeyType => _ValueType)` - a dynamically sized mapping (hash table) with quite a few Ethereum specific intricacies not listed below. There is no size function, nested mappings are prohibited, and mappings cannot be iterated.

Visibility choices are listed below:

- `public` - automatic getter function is generated
- `internal` - can only be accessed internally to current contract or contracts inheriting from the current
- `private` - only visible to current contract and no inherited contracts

Because smart contracts are deployed on the blockchain **all data is stored on every node of the Ethereum network** whether it's "public" or "private". Hence secrets can only be saved with encrypted or hashed data.

Functions

Functions headers are defined in a contract with the following structure:

```
function name(<parameters>) {internal|external} [modifier] [returns (<return types>)]
```

`name` is the name of the function. `<parameters>` are the parameters of the function such as `(address source, uint256 amount)`. `internal` function can only be used in the current contract while `external` function can be called from messages directly to the deployed contract. `returns` is optional and specifies the function's single or multiple `(<return types>)`. `modifier` is described in the next section.

A function body can have some implicit values accessible in functions, such as `msg.sender` which is the sender address of the message or transaction calling the function or `msg.value`, the amount of Ether sent with the message.

Modifiers

A modifier is a special block of code to add validation or assertion logic to a function. For example,

```
address public masterAccount;
modifier onlyMasterAccount {
    if(msg.sender != masterAccount) throw;
    _ ;
}
```

can be used in a function like,

```
function appointMasterAccount(address next) onlyMasterAccount { masterAccount = next; }
```

to restrict `appointMasterAccount` to only a caller transactions whose sender is an address `masterAccount` (meaning the transaction was signed by master account's private key) otherwise the function executes `throw`. `throw` is a very important command in the Solidity programming language, throwing an exception, reverting all execution, and forfeiting all gas to the miner. Contract functions typically have many many modifiers and/or validation logic to throw an exception for unexpected input.

Events

Events are the last major construct used in this thesis. An event is a sort of structured log message that is later searchable in the Ethereum blockchain. For example, inside a function could have the event defined:

```
event Transfer(address indexed source, address indexed destination, uint256 amount)
```

which can be executed in a function by calling `Transfer(source, destination, amount)`. This creates a log entry in the Ethereum blockchain that could be searched for a particular `source` and `destination` address.

2.5. Estonia ID Card

Estonian ID Card and Mobile ID is used to identity users in the the Euro 2.0 digital currency system. The detailed architecture of the ID card system is not considered in the security analysis presented in this thesis. Martens summarizes the development of the national electronic Identity Management System (eIDMS), describing its technical features, its historical development, and the contributions from the public and private sector[49]. Springall et al. perform a security analysis of Estonian internet voting

scheme and in their introduction provide an overview of the Estonian national ID card infrastructure [60].

The central agency of the Estonia ID card system is the Estonia Citizenship and Migration Board (CMB), under the authority of the Ministry of Interior, with the responsibility of maintaining the population register, administering national Personal Identification Codes (PIC), and issuing identity documents. The 11 digit PIC has the following structure:

- digit for gender and century of the birth (one digit for two attributes)
- date of birth digits (YY+MM+DD)
- three random digits
- one checksum digit

The Estonian ID card contains two electronic RSA key pair certificates, one for authentication and one for legally binding signatures regulated under the Estonian Digital Signatures Act and administered by SK Certification Center[8]. The card also has an unprotected data file containing the personal information displayed on the physical card. Public key certificates are stored in a public LDAP database administered by SK[10]. Certificates require a pin code for use. Both certificates and card have a validity of five years for citizens or length of validity of legal residence if less than five years. Mobile ID is offered by carriers through a special PKI-capable SIM card and allows the same functionality as the ID card through a mobile interface. Estonia ID card or Mobile ID TLS authentication is widely used in online banking in the country. A user sends an authentication requests, enters his pin code, and if the certificate is still valid, is authenticated to the website.

2.6. Information Security Risk Analysis Frameworks

Information System Security Risk Management (ISSRM) is a broad field with *"over 200 practitioner-oriented risk management methods and several academic security modeling frameworks available"*[29]. From these, we need to choose an appropriate framework to guide our security risk analysis of the Euro 2.0 digital currency prototype in Chapter 5 - Euro 2.0 Security Risk Analysis. Duboi et al. build a unified domain model of ISSRM by surveying and categorizing risk management standards, information security and information technology standards, risk management methodologies, and security frameworks, as well as the state of the art of security modeling languages. This serves as a useful source of representative security risk analysis frameworks to choose from.

Ahmed and Matulevičius explore eliciting security requirements earlier in business

process modeling by introducing security risk oriented patterns[1]. The goal is to *"align business processes and security requirements elicited using security risk-oriented patterns"* to mitigate security risks. This gives a good contextual framework for thinking through a security analysis on business processes.

Uzunov and Fernandez give a deeply technical overview of security patterns of distributed systems[62]. They first survey threat patterns, pattern-based threat taxonomies, and architectural contexts for the threat patterns. Then they provide a list first level security threat patterns and second level meta-security threat patterns to see further analysis. Then they construct a specialized taxonomy of security threats for distributed and peer to peer systems. The ideas presented are useful input for the Euro 2.0. security risk analysis in Section 5.

From the review of various security risk analysis frameworks, *OCTAVE Allegro* was chosen as the simplest framework allowing a strait forward risk analysis of the Euro 2.0 proccesses using a seven step methodology[23]:

- *Step 1 - Establish Risk Measurement Criteria*
- *Step 2 - Develop an Information Asset Profile*
- *Step 3 - Identify Information Asset Containers*
- *Step 4 - Identify Areas of Concern*
- *Step 5 - Identify Threat Scenarios*
- *Step 6 - Identify Risks*
- *Step 7 - Analyze Risks*
- *Step 8 - Select Mitigation Approach*

We use inspiration from security patterns of from the other two papers as input for the OCTAVE Allegro risk analysis. We chose not to undergo formal modeling or complicated modeling software, as complexity explodes when applied to a complicated system like Euro 2.0, without any benefits of the time invested.

3. Euro 2.0 Digital Currency Prototype

3.1. Introduction

This chapter introduces the Euro 2.0 digital currency system. The contents of this section are based from a prototype developed by the Euro 2.0 Foundation as a non profit initiative. The code is accessible in Github under the MIT License as indicated in Appendix A.1. The chapter explores the following research question:

> **RQ1:** *How do we describe the features and components of the Euro 2.0 digital currency system for a security analysis?*

Which is broken down into the following sub-questions:

> **RQ1.1:** *What are the features of the Euro 2.0 digital currency?*
>
> **RQ1.2:** *What are the decentralized components?*
>
> **RQ1.3:** *What are the centralized components?*
>
> **RQ1.4:** *How do the decentralized and centralized work together to fulfill the features?*

Answering *RQ1* lays the groundwork for further analysis in Euro 2.0 Change Analysis and Euro 2.0 Security Risk Analysis. Section 3.2 will briefly go through the motivation behind the Euro 2.0 digital currency system. Section 3.3 will give an overview of the system architecture and features currently implemented answering *RQ1.1*. Section 3.4 will give an overview of the decentralized components built on the Ethereum platform answering *RQ1.2*. Section 3.5 will give an overview of centralized components built with traditional software development architecture answering *RQ1.3*. Finally Section 3.6 will briefly describe how the centralized and decentralized components work together to fulfill the features answering *RQ1.4*.

3.2. Euro 2.0 Motivation

The idea of Euro 2.0 originated in 2014 from Kristo Käärmann's blogpost *Government Backed Bitcoin*[44]. Following the success of Bitcoin, he saw an opportunity for a payments settled on a Government backed cryptocurrency, *EuroCoin*, pegged one to one to real world Euro counterpart. Paypal cofounder Max Levchin supports the same idea, *"I am confident that there will be a form of settlement that will be a crypto-currency"*[24]. Bitcoin's short term fluctuations, functioning both as a commodity and a currency, in

value make it unreasonable for end to end holdings in a fiat based economy. Removing financial intermediaries in payments reduces the costs of transactions.

Even some governments have expressed interests in digital currency. The Swedish central bank debated whether to become the first to issue digital currency as a response to the country's move away from cash[50]. The UK government pushed for a "Call for Information" on digital currencies in order to assess risks[37] that was later responded by Citi bank's global Treasury and Trade Services (TTS) Technology and Innovation Team with a suggestion for government to create its own digital currency[59]:

> *"The greatest benefits of digital currencies can be realized through the government issuing a digital form of legal tender. This currency would be less expensive, more efficient and provide greater transparency than current physical legal tender or electronic methods."*

Following the rise in popularity of Ethereum, in mid 2016, another blogpost by Käärmann *The future of money may be in the ether*[45] described the rough idea of the system. We can describe Euro 2.0 as attempting to solve the business problem:

> *How do I build a system to digitally store and transact fiat currency without a centralized for-profit financial intermediary conveniently, trustfully, and securely while satisfying government regulations?*

From this business problem derives the key pillars of Euro 2.0:

1. Fiat backed
2. Decentralized accounts and transactions
3. Identities of all users
4. Convenient conversion between fiat and digital assets
5. Government can control monetary supply
6. Features for law enforcement and AML

Fiat backed (1) provides monetary stability to the system and builds trust with users and merchants to keep assets in the digital format. This eliminates the issue of constantly fluctuating daily value between real world and digital assets seen in crypto currencies today.

Decentralized accounts and transactions (2) comes from the need of an impeccable record and process of transacting value that no centralised party can modify in a database. This is the key benefit and usecase of distributed ledger technology.

Identities of all users (3) builds trust in the system for users, merchants, and governments.

If everyone is identified, any transaction can be legally ramified (assuming legislation arises supporting distributed ledger records in the court). The strong identity of Estonian ID card is a perfect fit for a digitally identifying mechanism and the main driver of the first prototype.

Convenient conversion between fiat and digital assets (4) makes it easy for users to gradually start using the digital currency from existing financial infrastructure. At first this exchange gateway can be a private financial institution, similar to Tether[46], with a proof of reserve of the one to one Euro backing. Ideally this conversion service is hosted by the central bank itself to limit the number of financial intermediaries.

Government can control monetary supply (5) follows from the ideal scenario of (4), the central bank can issue new money directly into the digital monetary system without a real world Euro counterpart.

Features for law enforcement and AML (6) is the most controversial aspect of the system. Bitcoin was made to sidestep the direct control of any third party, including government[51]. Euro 2.0 needs to be regulated from the start with full support of governments, which requires necessary intervention that typically happen today via cooperation with regulated financial institutions.

3.3. Euro 2.0 Features

The following section describes the main features of the Euro 2.0 system. Features are enumerated with the label **XY#** where:

- **X** is an abbreviation of the domain of the feature, described in the Domains section
- **Y** is client, admin, or server. Client is a user of the system, admin is some superuser with special permissions, and server is some centralized or decentralized server infrastructure
- **#** is the number of a feature for a given **XY**

Features are implemented with a combination of decentralized and centralized components described in Section 3.4 and Section 3.5 respectively.

3.3.1. Domains

Table 1 gives an overview of the domains of Euro 2.0 used to group features described in later sections.

Domain	Abbr.	Description
Account	A	Mechanism of storage of value in the system
Identity	I	Features linking real life human identity to users
Transaction	T	Mechanism of transfer of value in the system
Reserve	R	Features regarding the exchange of value between traditional Euro (EUR) in the financial system and digital Euro (EUR2)
Enforcement	E	Features for government regulators and law enforcement to police the system for misuse and AML
Convenience	C	Features for improving user experience and expanding use cases

Table 1. Domains of Euro 2.0

3.3.2. Entities

Table 2 clarifies key data entities used throughout the description of features.

Entity	Description
Address	Ethereum externally controlled address (i.e. public key) holding EUR2 and Ether
Key	Ethereum address private key needed to make transactions (EUR2 or Ether) for an address
Account	All verified Ethereum addresses for an ID
Account Balance	Sum of total balance of EUR2 in an account
ID	Estonian ID code of an individual (citizen, resident, e-resident)
ID Name	Name of person associated with an Estonian ID Code
EUR	Euro currency in an Estonian bank account
EUR2	Euro currency in the Euro 2.0 digital form, balance of an address in the Euro 2.0 Ethereum contract
Wallet	Client program (app or mobile web) used as an interface to Euro 2.0 system, accessible keys to addresses
Wallet Password	Password used to encrypt and decrypt local wallet
Backup Password	Password used to encrypt and decrypt keys and address to send to backup server
Designator account	Account collecting seized funds

Table 2. Euro 2.0 Entities

3.3.3. Roles

Role	Description
CryptoFiat Master	Ethereum address (public / private key) to master contract
Approver	Ethereum address (public / private key) entitled to approver operations
Reserve Bank	Ethereum address (public / private key) entitled to reserve operations and funds
Enforcer	Ethereum address (public / private key) entitled to to enforcer operations
Designator	Ethereum address (public / private key) entitled to designator operations

Table 3. Euro 2.0 Roles

3.3.4. Processes

Table 4 clarifies different processes used throughout the description of features.

Process	Description
Approval	Link Ethereum address with the an Estonian ID number via ID card certificate verification. Allow Ethereum address to send EUR2.
Freeze	Stop address from sending EUR2.
Close	Stop address from receiving EUR2.
Seize	Remove EUR2 from an address and send them to the designator account.
AML checks	AML database lookup based on real name of person.

Table 4. Euro 2.0 Processes

3.3.5. Account Features

Client

- **AC1** - Create address
- **AC2** - View account balance

Server

- **AS1** - Aggregate account balance

The account features allow users of store balance of EUR2. The requirement is an Ethereum public and private key pair created in **AC1**. Viewing account balance is taking the sum of all addresses under ownership of the user's private keys. A server aggregation is needed for retrieving the balances and resolving delegated balances described later.

3.3.6. Identity Features

Client

- **IC1** - Approve address

Server

- **IS1** - Check external source for valid ID
- **IS2** - Approve address
- **IS3** - Save ID number to address mapping
- **IS4** - Look up Ethereum address by ID

In order for an address to transact EUR2, it needs to be approved, that is linked with an identity. The server here plays a critical role in approving the address by checking a client approval request (**IC1**) with an external source (**IS1**) then both approving this address on the blockchain (**IS2**) and saving this mapping (**IS3**) for use in other components (**IS4**).

3.3.7. Transaction Features

Client

- **TC1** - Send EUR2 to an address
- **TC2** - Send EUR2 to an ID
- **TC3** - Receive EUR2 to an address
- **TC4** - Receive EUR2 to an ID
- **TC5** - Send EUR2 to a EUR bank account

Server

- **TS1** - Resolve ID to an address
- **TS2** - Escrow EUR2 to an ID
- **TS3** - Release escrow EUR2 to an address for an ID
- **TS4** - Execute valid EUR transaction
- **TS5** - Execute valid EUR2 transaction

The transaction features are the heart of the EUR2 system. What differentiates Euro 2.0 from other cryptocurrencies is the ability to easily send funds amongst digital (EUR2) and real (EUR) Euros. **TC1** and **TC3** allow a user to transact EUR2 directly with the Ethereum decentralized system using Ethereum addresses. **TC2** and **TC4** allow a user to conveniently send EUR2 to a human being via their ID code using the centralized identity infrastructure (**IS4**). Funds sent to an ID that does not yet have an Ethereum address will have EUR2 held to a new Ethereum address in Escrow (**TS2**) and then later released and transferred to an address created and approved by the receiver (**TS3**). **TS5** provides the infrastructure to execute EUR2 transactions for **TC1** through **TC4**. A centralized party needs to provide the mediation gateway (**TS4**) allowing a user to send EUR2 to a EUR bank account (**TC5**).

3.3.8. Reserve Features

Client

- **RC1** - Convert EUR to EUR2
- **RC2** - Convert EUR2 to EUR
- **RC3** - Proof of reserve

Admin

- **RA1** - Assign reserve bank address
- **RA2** - Manage reserve bank account

Server

- **RS1** - Resolve address from ID of received EUR transaction
- **RS2** - Create EUR2
- **RS3** - Destroy EUR2

The reserve features allow convenient conversion between EUR and EUR2. The reserve process for converting from EUR to EUR2 is to to receive a EUR bank transaction with an individuals ID (**RS1**), create EUR2 (**RS2**) of the equivalent amount, resolve the ID to an Ethereum address (**TS1**), and send this to the address for the ID (**TS5**). Likewise the process from converting EUR2 to EUR is receiving a EUR2 transaction (**RC1**), destroy the EUR2 (**RS3**), and send a bank transfer of the equivalent amount in EUR (**TS4**). Finally there is admin feature to assign the reserve EUR2 address to make conversions (**RA1**) and actually perform manage the banking operations on the bank account (**RA2**).

3.3.9. Enforcement Features

Admin

- **EA1** - Freeze account
- **EA2** - Seize account funds to designator's address, closing the account
- **EA3** - Assign enforcement
- **EA4** - Assign designator
- **EA5** - Set designator address
- **EA6** - View name check and analysis data

Server

- **ES1** - AML checks on ID names
- **ES2** - Create database of transactions with ID

The enforcement features allow law enforcement to do their job that would be traditionally be done via financial institutions. Law enforcement can freeze (**EA1**) and seize (**EA2**) funds to the designator address (**EA5**). Admin controls exist to assign the role of enforcer (**EA3**) and designator (**EA4**). Finally automatic processes in the server need to perform name checks on all users of the system (**ES1**) and create a database of identities and transactions (**ES2**) to be usable by some analysts (**EA6**).

3.3.10. Convenience Features

The convenience features are not necessarily central to a functioning and compliant fiat and digital monetary system, but are quite useful to increase the usefulness and usability barrier. They are quite important to consider in a security analysis, hence listed here.

Wallet Security

Client

- **CC1** - Encrypt wallet
- **CC2** - Decrypt wallet

Since private keys are stored locally on a user's device, they should be encrypted (**CC1**) with some password when not in use and decrypted (**CC2**) when needed for signing transactions.

Key Backup

Client

- **CC3** - Encrypt keys for addresses
- **CC4** - Create a cryptographic challenge
- **CC5** - Send addresses, keys, and challenge to backup server for an ID
- **CC6** - Retrieve addresses and keys from backup server for an ID with challenge answer
- **CC7** - Decrypt keys for addresses

Server

- **CS1** - Save encrypted addresses and keys for an ID
- **CS2** - Check challenge answer
- **CS3** - Load encrypted addresses and keys for an ID

Since keys are stored locally, if not backed up, they can be easily lost with the device. Euro 2.0 provides a centralized convenience feature to back up these keys for an ID and secured by a challenge question. The user can encrypt their keys with a password (**CC3**), create a cryptographic challenge (**CC4**), send the addresses, encrypted keys, and challenge to the server (**CC5**) to be saved (**CS1**). The user can then retrieve the addresses and encrypted keys for an ID from the server (**CC6**) by providing the correct challenge answer checked by the server (**CS2**) (**CS3**), and finally the keys can be decrypted by the user (**CC7**).

Transfer Details

Client

- **CC8** - Encrypt copies of transfer details for sender and receiver
- **CC9** - Save transfer details for a transaction hash

- **CC10** - Load transfer details for a transaction hash
- **CC11** - Decrypt transfer details for sender or recipient

Server

- **CS4** - Save transfer details for a transaction hash
- **CS5** - Load transfer details for a transaction hash

Transfer details features provide extra information to transfers only visible to sender and receiver. First a sender of a transaction encrypts two copies of the transfer details, one with the sender's public key and the other with the recipient's private key (**CC8**). Then these can be saved to the transfer details server with the transaction's hash (**CC9** and **CS4**). Then these details can be loaded from the server (**CS5**) by the sender or receiver from the server (**CC10**) and decrypted with their respective private key (**CC11**).

Account Recovery

Client

- **CC12** - Set address recovery address
- **CC13** - Recover address funds

Server

- **CS6** - Execute address funds recovery transaction and close account

Account recovery features offer some protection from accidentally losing keys to an address. A user can set a very trusted address, a friend or close relative, with the able to recover funds (**CC12**). This trusted party at any time can perform the funds recovery and withdraw all of the account's funds to their account with their key (**CC13**) executed by the server (**CS6**).

Delegated Transfers - Send without Ether

Client

- **CC13** - Execute EUR2 transaction with a EUR2 fee instead of Ether for gas
- **CS7** - Fulfill delegated EUR2 transaction and claim EUR2 fee

Finally, this convenience feature eliminates the side effect of using Ethereum, consuming gas (ether) in transactions. A trusted third party, the wallet server in this case, can execute transactions on behalf of the sender (**CC13**) and take the fee in EUR2 cents instead of Ether (**CS7**). This requires only the trusted third party to have Ether and the user can only worry about EUR2.

3.4. Decentralized Components

This section walks through the decentralized components of the Euro 2.0 system on the Ethereum distributed application platform. The main container of code deployment is the

Contract. Once deployed, the contract is on the blockchain forever processing received transactions and messages, unless an optional destroy function is implemented and called. All function calls and data modifications are irreversible and publicly visible while being executed on the thousands of computers on the Ethereum network maintaining the Ethereum blockchain. Thus we describe the implementation of the Euro 2.0 contracts in detail, since this non-revertible code is important for security considerations.
The decentralized contracts are comprised of of the main administrator contract (*CryptoFiat*), shared contracts (*Constants*, *Relay*, *Data*, *InternalData*), and the upgradable subcontracts providing functionality to the system (*Accounts*, *Approving*, *Reserve*, *Enforcement*, *AccountRecovery*, *Delegation*). The contract code can be found in Appendix A:2. These decentralized contracts function as gateways to the shared decentralized data of the Euro 2.0 system used together with centralized components of Section 3.5 to fulfill the features of the system in Section 3.6.

3.4.1. CryptoFiat Contract

The heart of decentralized Euro 2.0 is the *CryptoFiat* contract. This contract functions as an administrator, managing the master address and the references to all deployed contracts comprising the main functionality of the Euro 2.0 system.

Name	Type	Description
`masterAccount`	`address`	Master Ethereum account whose allowed to do operations on the CryptoFiat contract. Set in the constructor and changeable by master account owner.
`contractAddress`	`uint256 => address`	Stores mapping from subcontract id to deployed address of active subcontract.
`contractId`	`address => uint256`	Stores mapping from active subcontract address to id.
`contracts`	`address[]`	Array with the address of all contracts ever added to the Euro 2.0 system. It's contents is never cleared.

Table 5. CryptoFiat contract data

Function	Args	Description
`contractActive`	`address addr`	Returns `bool` whether the subcontract at address `addr` is active.
`contractsLength`		Returns the length of `contracts`, i.e. the number of all contracts ever deployed.
`appointMasterAccount`	`address next`	Sets `masterAccount` to `next` and hence gives up control of the CryptoFiat contract. Can only be called by master account owner.
`upgrade`	`uint256 id` `address next`	Upgrades the active contract with `id` to the address `next`. Only owner of master account can call this. `prev` cannot be the same contract as `next`. `next` cannot be an already active contract. `contractAddress`, `contractId`, and `contracts` are updated appropriately.

Table 6. CryptoFiat contract functions

Event	Args	Description

`ContractUpgraded`	`uint256 indexed id`
	`address previous`
	`address next`
		Event describing upgrading the contract of index `id` from `previous` to `next`

Table 7. CryptoFiat contract events

3.4.2. Shared Contracts: Constants, Relay, Data, InternalData

The code in the shared contracts in Euro 2.0 are exposed ultimately in the *InternalData* contract interface that heavily calls *Data contract*. *InternalData* contract acts as an abstract Contract inherited by all the sub contracts providing functionality. Only the *Data* contract is actually constructed and deployed on the blockchain, the other shared contracts are simply organizing code. *InternalData* inherits from *Constants* and *Relay*. While *Relay* inherits from *Constants* and references *Data* for use by *InternalData*. *Data* inherits from *Relay*.

3.4.2.1 Constants Contract

The *Constants* contract has no functionality, but holds convenient values accessible to all subcontracts. The contract ids in Table 8 are used in contract deployment mapping to access particular subcontracts. Bucket identifiers in Table 9 are used in data storage key computation to distinguish types of data. Account states listed in Table 10 are boolean flags used as permissions for flow of transactions. Finally events listed in Table 11 are a convenient record of when some action occurred in the system.

Contract Name	Id	Description
DATA	1	Contains data storage for all Euro 2.0 contracts
ACCOUNTS	2	Contract regarding account operations
APPROVING	3	Contract approving accounts based on verified ID
RESERVE	4	Contract able to increase, decrease, and transfer supply
ENFORCEMENT	5	Contract with law enforcement operations
ACCOUNT_RECOVERY	6	Contract assigning account recovery options
DELEGATION	7	Contract providing delegate transfers

Table 8. Constant contract subcontract ids

Contract Name	Id	Description
STATUS	1	Account states
BALANCE	2	Account balance
DELEGATED_TRANSFER_NONCE	3	Delegate transfer nonce for original sender address
RECOVERY_ACCOUNT	4	Recovery account assignments
TOTAL_SUPPLY	5	Total supply issued by reserve

Table 9. Constant contract bucket identifiers

State	Id	Description
APPROVED	1	Account has a linked ID approved and can send funds.
CLOSED	2	Account is closed (by owner or law enforcement) and cannot receive funds.
FROZEN	4	Account is frozen by law enforcement and cannot send funds.

Table 10. Constant contract account state

Event	Args	Description

`Transfer`	`address indexed source`
	`address indexed destination`
	`uint256 amount`
		Event when `amount` was transferred from `source` to `value`.

`AccountApproved`	`address indexed source`	Event signifying address `source` was approved.
`AccountClosed`	`address indexed source`	Event signifying address `source` was closed.
`AccountFreeze`	`address indexed source` `bool frozen`	Event signifying address `source` was frozen.
`SupplyChanged`	`uint256 totalSupply`	Event signifying total supply changed to `totalSupply`.

Table 11. Constant contract events

3.4.2.2 Relay Contract

The *Relay* contract holds convenience methods for subcontracts to access each other. The functions in Table 14 use the reference to *CryptoFiat* in Table 12 to provide a reference to all deployed subcontracts. There is also convenient restriction modifiers listed in Table 13.

Name	Type	Description
`cryptoFiat`	`address`	Reference to main *CryptoFiat* contract deployment used to resolve address to all other contracts

Table 12. Relay contract data

Modifier	Description
`onlyMasterAccount`	Restricts function to only be usable by master account by checking `msg.sender`.
`onlyContracts`	Restricts function to only be usable by active contracts.

Table 13. Relay contract modifiers

Function	Args	Description
`switchCryptoFiat`	`address next`	Sets the address of *CryptoFiat* contract. Restricted to `onlyMasterAccount`.
`contractAddress`	`uint256 id`	Returns the address of subcontract of `id` listed in Table 8.
`accounts`		Returns reference to `ACCOUNTS` contract.
`data`		Returns reference to `DATA` contract.
`approving`		Returns reference to `APPROVING` contract.
`reserve`		Returns reference to `RESERVE` contract.
`accountRecovery`		Returns reference to `ACCOUNT_RECOVERY` contract.
`delegation`		Returns reference to `DELEGATION` contract.

Table 14. Relay contract functions

3.4.2.3 Data Contract

The *Data* contract provides the main data access layer for all subcontracts which is accessed by convenience methods in the *InternalData* contract. No subcontracts work with the *Data* contract directly. The main data object is defined in Table 15 and functions on this data in Table 16.

Name	Type	Description
`_data`	`bytes32 => bytes32`	The shared data storage structure for all subcontracts

Table 15. Data contract data

Function	Args	Description
set	uint256 bucket bytes32 key bytes32 value	Saves the data value with the hash key sha3(bucket, key), with bucket referring to the bucket id list in Table 9 and key being an arbitrary value. Restricted to onlyContracts callers.
get	uint256 id	Returns byte32 value stored in hash key sha3(bucket, key), with bucket id being from the list in Table 9.

Table 16. Data contract functions

3.4.2.4 InternalData Contract

InternalData is the main interface for accessing data storage used by all subcontracts. All functions here are marked internal, meaning they can only be accessed from within a contract and not called by external transactions or messages. All subcontracts extend *InternalData*. Data access functions are listed in Table 17, account status functions are listed in Table 18, modifiers in Table 19, and account functions in Table 20. The data dependency of functional subcontracts on *InternalData* and *CryptoFiat* are depicted in Figure 1.

Function	Args	Description
_balanceOf	address addr	Returns uint256 BALANCE stored for address addr.
_setBalanceOf	address addr uint256 value	Stores BALANCE of value for addr.
_statusOf	address addr	Returns uint256 account STATUS of address addr as listed in status codes of Table 10.
_setStatusOf	address addr uint256 value	Stores account STATUS of value for addr.
_delegatedTransferNonceOf	address addr	Returns uint256 of last nonce used in delegatedTransfer for original sender address addr.
_setDelegatedTransferNonceOf	address addr uint256 value	Stores last nonce used value in delegatedTransfer for original sender address addr.
_recoveryAccountOf	address addr	Returns the address recovery account set for address addr.
_setRecoveryAccountOf	address addr address value	Stores recovery account value for address addr.
_totalSupply		Returns total supply uint256 of tokens in circulation.
_setTotalSupply	uint256 value	Stores value of total tokens in circulation.

Table 17. InternalData contract account status functions

Function	Args	Description
_isApproved	address account	Returns bool whether account is APPROVED.
_isClosed	address account	Returns bool whether account is CLOSED.
_isFrozen	address account	Returns bool whether account is FROZEN.

Table 18. InternalData account status functions

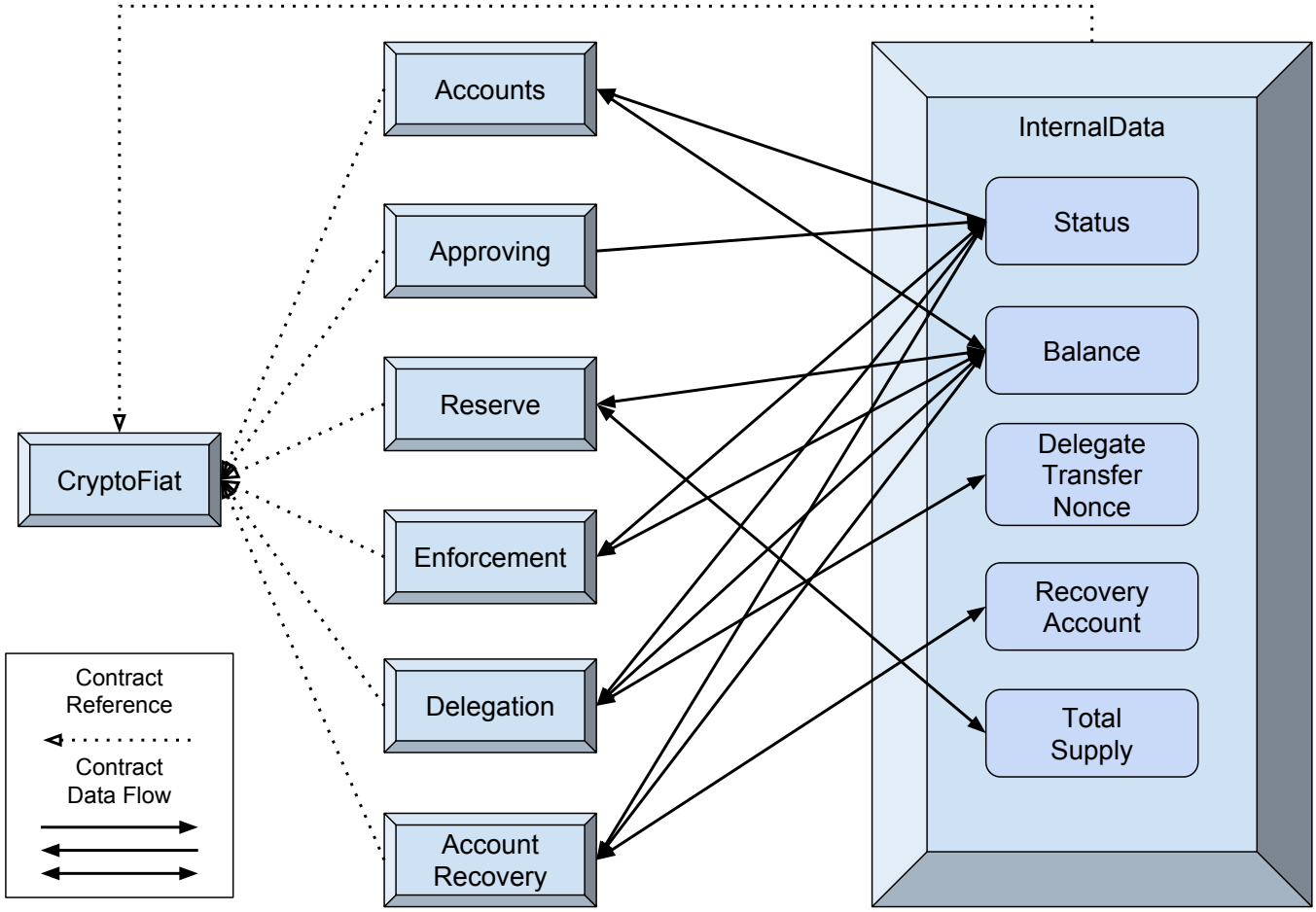

Figure 1. Contract Data Dependencies

Function	Args	Description
`canSend`	`address account`	Validates if `account` can send by throwing an exception if it is not approved, frozen, or the address 0.
`assertSend`	`address account`	Function form of `canSend`.
`canReceive`	`address account`	Validates if `account` can receive by throwing an exception if it is closed or the address 0.
`assertReceive`	`address account`	Function form of `canReceive`.

Table 19. InternalData modifiers

Function	Args	Description
`_withdraw`	`address account` `uint256 amount`	Withdraws `amount` from balance of `account`. Throws exception if `amount` is more than balance.
`_deposit`	`address account` `uint256 amount`	Deposits `amount` into balance for `account`. Throws exception if amount to withdraw plus balance is less than balance (overflow).

Table 20. InternalData account functions

3.4.3. Accounts Contract

The *Accounts* contract exposes all the functions, listed in Table 21, necessary from *InternalData* to work with accounts and balances.

Function	Args	Description
`balanceOf`	`address account`	Returns `uint256` balance of `account`.
`statusOf`	`address account`	Returns `uint256` of boolean flags with the status of `account` as defined in table Table 10.
`isApproved`	`address account`	Returns `bool` whether `account` is approved (can send funds).
`isClosed`	`address account`	Returns `bool` whether `account` is closed (can not receive funds).
`isFrozen`	`address account`	Returns `bool` whether `account` is frozen (can not send funds).
`transfer`	`address destination` `uint256 amount`	Withdraws `amount` from caller `msg.sender` and deposits `amount` into `destination`. Restricted to `msg.sender` that `canSend` and `destination` that `canReceive`. Logs `Transfer` event for `source`, `destination`, and `amount`.

Table 21. Accounts functions

3.4.4. Approving Contract

Approving contract exposes the functions necessary for approving accounts to transact in Euro 2.0. Only the `accountApprover`, Table 22, can approve accounts. Modifiers, Table 23, restrict access to functions, Table 24, to `accountApprover`.

Name	Type	Description
`accountApprover`	`address`	Account approver address.

Table 22. Approving contract data

Function	Args	Description
`onlyAccountApprover`		Validates if `msg.sender` is `accountApprover` otherwise throws an exception.

Table 23. Approving contract modifiers

Function	Args	Description
`appointAccountApprover`	`address next`	Assigns address of `accountApprover` to `next`, removing approving privileges to the old address. Restricted to `onlyAccountApprover`.
`approveAccount`	`address account`	Sets status of `account` to APPROVED to be able to send money. Restricted to `onlyAccountApprover`. Logs `AccountApproved` event for the approved `account`.
`approveAccounts`	`address[] accounts`	Sets status of each account in `accounts` to APPROVED to be able to send money. Restricted to `onlyAccountApprover`. Logs `AccountApproved` event for the approved `account`.
`closeAccount`	`address account`	Sets status of `account` to CLOSED to prevent the account from receiving money. Restricted to `onlyAccountApprover`. Logs `AccountClosed` event for the closed `account`.

Table 24. Approving contract functions

3.4.5. Reserve Contract

The *Reserve* contract exposes the functions to increase and decrease supply of EUR2 in the Euro 2.0. Only the `reserveBank` account, Table 25, can perform these actions. The modifier in Table 26 restrict access to functions listed in Table 27 to `reserveBank`.

Name	Type	Description
`reserveBank`	`address`	Reserve bank address.

Table 25. Reserve contract data

Function	Args	Description
`onlyReserveBank`		Validates if `msg.sender` is `reserveBank` otherwise throws an exception.

Table 26. Reserve contract modifiers

Function	Args	Description
`appointReserveBank`	`address next`	Assigns address of `reserveBank` to `next`, removing privileges of the old address. Restricted to `onlyReserveBank`.
`totalSupply`		Returns `uint256` total supply of Euro 2.0 system. No restrictions on calling this method.
`increaseSupply`	`uint256 amount`	Increases supply by `amount` and deposits `amount` of newly created EUR2 to the `reserveBank`. Restricted to `onlyReserveBank` and `reserveBank canReceive`. Throws exception if supply plus amount overflows.
`decreaseSupply`	`uint256 amount`	Decreases supply by `amount` and withdraws `amount` from `reserveBank`, destroying the EUR2. Restricted to `onlyReserveBank` and `reserveBank canSend`. Throws exception if supply is less than amount. Logs `SupplyChanged` for the new supply `amount`.

Table 27. Reserve contract functions

3.4.6. Enforcement Contract

The *Enforcement* contract exposes the functions for law enforcement to freeze and seize funds. The two roles are the law enforcer, who can do the actions to freeze and seize funds, and account designator, who can control the account where the funds can be seized, described in Table 28. Modifiers in Table 29 restrict access to functions based on role. The functions are listed in Table 30.

Name	Type	Description
`lawEnforcer`	`address`	Law enforcer address.
`accountDesignator`	`address`	Account designator address.
`account`	`address`	Law enforcement account.

Table 28. Enforcement contract data

Function	Args	Description
`onlyLawEnforcer`		Validates if `msg.sender` is `lawEnforcer` otherwise throws an exception.
`onlyAccountDesignator`		Validates if `msg.sender` is `accountDesignator` otherwise throws an exception.

Table 29. Enforcement contract modifiers

Function	Args	Description
`appointLawEnforcer`	`address next`	Assigns address of `lawEnforcer` to `next`, removing privileges of the old address. Restricted to `onlyLawEnforcer`.
`appointAccountDesignator`	`address next`	Assigns address of `accountDesignator` to `next`, removing privileges of the old address. Restricted to `onlyAccountDesignator`.
`withdraw`	`address from` `uint256 amount`	Withdraws `amount` from account `from` and deposits `amount` into law enforcement `account`. Restricted to `onlyLawEnforcer` and `account canReceive`. Logs `Transfer` event with `from`, designator `account`, and `amount`.
`freezeAccount`	`address target`	Sets status of `target` account to FROZEN so it can no longer send funds. Restricted to `onlyLawEnforcer`. Logs `AccountFreeze` event with `target` and value true.
`unFreezeAccount`	`address target`	Removes FROZEN status of `target` account so it can send funds again. Restricted to `onlyLawEnforcer`. Logs `AccountFreeze` event with `target` and value `false`.
`designateAccount`	`address account`	Sets law enforcement `account` to given `account`. Restricted to `onlyAccountDesignator` and `account canReceive`.

Table 30. Enforcement contract functions

3.4.7. AccountRecovery Contract

The *AccountRecovery* exposes functionality intended for an account recovery mechanism. An account owner can designate a trusted party to withdraw all funds and close the account. The two functions enabling this functionality are listed in Table 31.

Function	Args	Description
`designateRecoveryAccount`	`address recoveryAccount`	Sets the recovery account for `msg.sender` to address `recoveryAccount`, replacing an existing `recoveryAccount`. To remove a recovery account the address 0 can be set.
`recoverBalance`	`address from` `address into`	Withdraws all funds from account `from` and deposits all funds to account `into` and then closing account `from` by setting its status to CLOSED. Restricted to account `from canSend` and account `into canReceive`. Logs `AccountClosed` event of `from` account and `Transfer` event of `from` account, `into` account, and `amount`.

Table 31. AccountRecovery contract functions

3.4.8. Delegation Contract

The *Delegation* contract allows a third party to initiate transactions in the Euro 2.0 system on behalf of other senders for a fee. This allows the third party to pay the fee in Ether of creating an Ethereum transaction while the original sender can just pay a fee in EUR2.

The transactions must be signed by the original sender meaning there is no possibility the delegate can make unsigned transactions on behalf of another sender. The functions of the contract are listed in Table 32.

Function	Args	Description
`nonceOf`	`address account`	Returns delegate transfer nonce for original sender `account`.

`transfer`	`uint256 nonce`
	`address destination`
	`uint256 amount`
	`uint256 fee`
	`bytes signature`
	`address delegate`
		Makes a delegate transfer from a `source` account, recovered by extracting public key from elliptic curve `signature`, to `destination` account of EUR2 `amount` with EUR2 fee `fee` paid to `delegate`. The `msg.sender` pays the Ether fee for the transaction instead of `source`. `nonce` is used to prevent replay attacks for the same `source` account. Logs `Transfer` event for `source` account (in signature), `destination`, and `amount` and if there is a fee, also logs `Transfer` event for `source`, `delegate`, and the `fee`.

`multitransfer`	`uint256 count`
	`bytes transfers`
	`address delegate`
		Performs the same logic as `transfer` for `count` number of transfers encoded in `transfers` byte array.

Table 32. Delegation contract functions

3.5. Centralized Components

The centralized components of the Euro 2.0 prototype are built with traditional client server architecture. Currently these are maintained by the Euro 2.0 Foundation, but would ideally be given over to government or the central bank to run their own monetary system. We only give a high level overview of the centralized component architecture, not going into any significant details of their implementation. At the time of the writing the components are based off of an unfinished prototype with quite a few bugs and some components not yet built. The goal of the subsection is to describe how the centralized components could be set up in such a system, where are the important keys and data are held, and in Section 3.6 how these components interact with the client and Ethereum to fulfill the features described in Section 3.2.

Figure 2 shows the architecture of the Euro 2.0 digital currency with its centralized and decentralized components. The figure is quite a helpful reference point when relating components to their place in the system. The Client is shown in purple as it does not necessarily have an owner. The green clouds represent external third party services not under control of the Euro 2.0 foundation. The blue represents all centralized components under control of the Euro 2.0 foundation. The green box represents the Ethereum network and deployed smart contracts in the system. Yellow keys represent the distribution of admin keys and whether they are or are not located on a server. Finally the arrows between components show the data flow between components

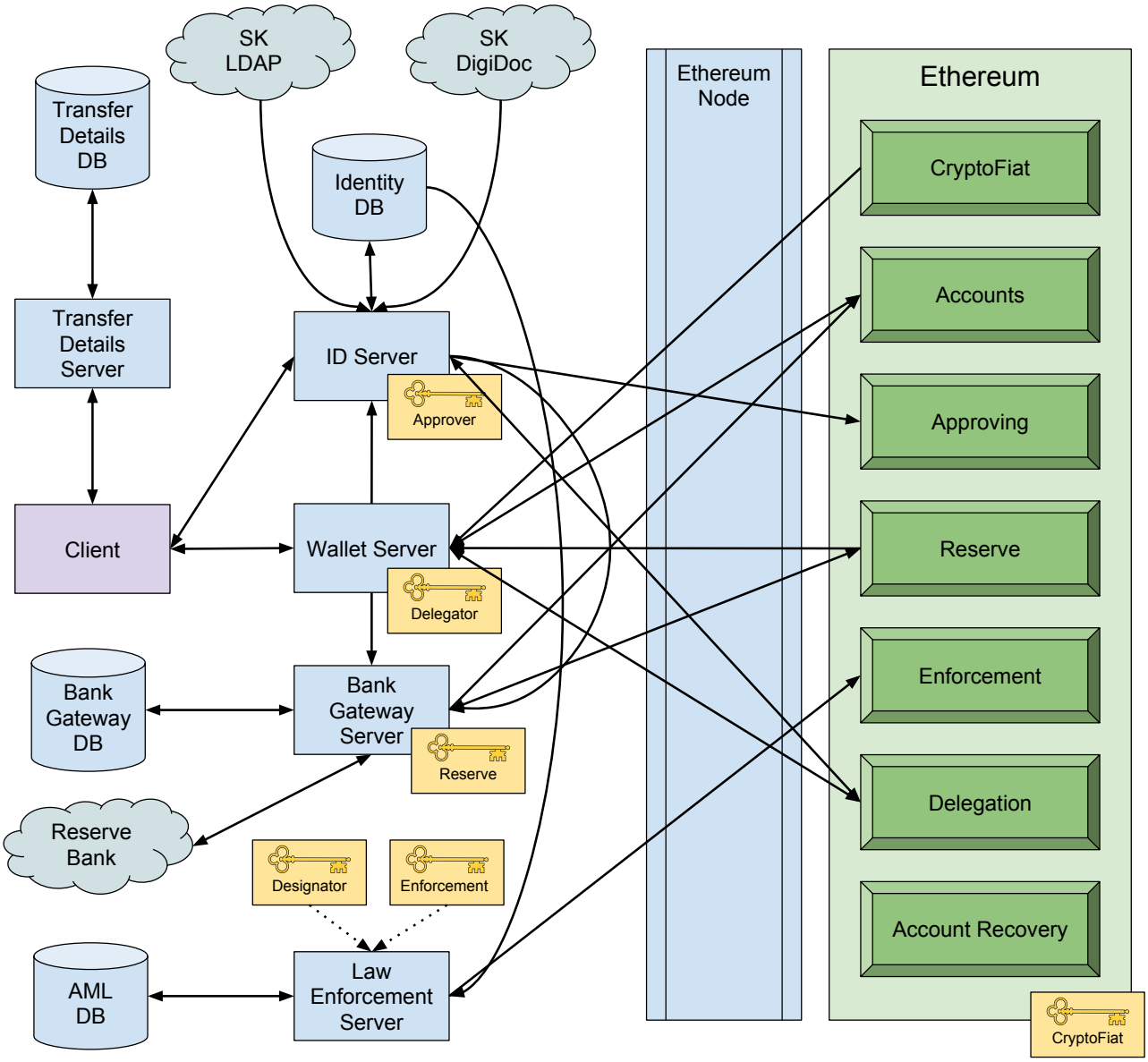

Figure 2. Euro 2.0 Architecture

3.5.1. Client

For our purposes the client refers to code running on a user's device interacting with the Euro 2.0 system. This can be a smart phone application or laptop browser. This is typically Javascript displaying the UI and making HTTP REST calls to the servers for interactions with the system. In the current prototype the client library does not interact directly with the decentralized components, but goes through the servers for all actions. The client will typically be storing a user's private keys in order to access their EUR2 funds.

3.5.2. Identity Server

The Identity server is an interface to all the ID operations of the system. It's main functions include:

- Authorization process for ID card, mobile ID, or bank transfer
- Approving addresses authorized by an ID
- Maintaining the mapping between addresses and ID
- Create and Claim Escrowed funds to an ID
- Addresses and keys backup system for an ID

3.5.3. Identity Database

The Identity Database holds the data needed for the Identity server. The tables include:

- **ethereum_account** - stores ID to Ethereum address mapping with columns: PID *owner_id*, Ethereum *address*, boolean *activated*, *creation_time*, *modification_time*, Estonian ID *authorisation_type*, and Ethereum *transaction_hash*
- **escrow** - stores escrow transactions to an ID with columns: PID *id_code*, encrypted *private_key*, Ethereum *address*, boolean *cleared*, and *clearing_hash*
- **backup_challenge** - stores information about backup challenges with columns: PID *id_code*, *plaintext*, *encrypted* text, and boolean *active*
- **key_backup** - stores actual encrypted key backups with columns: *challenge*, Ethereum *address*, encrypted *key_enc*, and *active*
- **ldap_response** - stores response from SK LDAP lookups with columns: *first_name*, *last_name*, PID and *id_code*.
- **pending_authorisation** - stores pending authorizations for an Estonian ID or Mobile ID card authentication with columns: *auth_identifier*, *type*, *address*, *serialised_mobile_id_session*, *creation_time*, *modification_time*, and *bank_transfer_payment_reference*

3.5.4. SK LDAP Directory Service

Estonian certificate center (SK)[8] provides an LDAP (Lightweight Directory Access Protocol) server to look up public certificates and legal name for a given Estonia ID number[10][11]. This service is the provider for Identity server to look up real names given an Estonian ID code.

3.5.5. SK DigiDocService

Estonian certificate center (SK) provides a DigiDocService which allows registered services to provide the ability for users to authenticate themselves with Estonian Mobile-ID or ID card[9]. This service is the provider for Identity Server to authenticate

users using their Mobile ID or Estonian ID card.

3.5.6. Wallet Server

Wallet server is the main entry point for conveniently creating transactions in the Euro 2.0 system and retrieving data from Etherum. It provides the following functionality:

- Get transaction info for a transaction hash
- Create delegated transfer to another an approved address where the server pays the ether
- Create transfer to a bank account via a delegated transfer to the reserve bank
- Get address status, balance, and delegate transfer nonce
- Get all transfers involving an address
- Get transfer fees in EUR2
- Get supply amount

3.5.7. Transfer Details Server

Transfer details server has a single purpose of saving transfer details for transactions off chain. It provides a general purpose interface that is up to the client to ensure proper encryption of transfer details so that sender and recipient can view the details. The functionality is simply:

- Save transfer details for a transaction hash
- Load transfer details for a transaction hash

3.5.8. Transfer Details Database

This is the database providing for the Transfer Details Server. It's simply a key value store with transaction hash as the key and a blob of data as the value.

3.5.9. Bank Gateway

The Bank Gateway is the entry point for the real Euros in Euro 2.0. The functionality includes:

- Update total reserve from uploaded bank statement files
- Update total supply
- Parse received EUR transactions and send EUR2 to addresses
- Handle EUR2 to EUR payments

3.5.10. Bank Gateway Database

The Bank Gateway Database store sent or received EUR transaction details from Reserve Bank account for records of operations with EUR.

3.5.11. AML Server

The AML server provides the functionality to law enforcement to police the system, including:

- Convenience interface for freezing and seizing funds
- Convenience interface for accessing AML database
- Runs jobs to fill AML database with transaction data
- Runs name check jobs on user identities

3.5.12. AML Database

The AML database provides law enforcement the transaction and identity data needed for AML investigations.

3.5.13. Ethereum Gateway

The Ethereum Gateway is an Ethereum full node able to process Ethereum JSON-RPC calls to post and receive data from the Ethereum network[64]. All RPC calls from centralized components go through this node.

3.6. Implementation of Features

This section describes how the decentralized components of Section 3.4 and the centralized components of Section 3.5 work together to fulfill the features described in Section 3.3. Again Figure 2 is a great reference for how the centralized and decentralized components fit together. The overview is this chapter is very high level. The exact implementations are in progress and so the most up to date source is the code in the Github repository linked to in Appendix A.1.

3.6.1. Account Features Implementation

The account features are described in Section 3.3.5. An account is created (**AC1**) in the Client when a private and public key are created. The account only actually gets to Ethereum when a transaction is sent from this address or the account is approved. The client fulfills viewing balance (**AC2**), that is the sum of all balance of all addresses for an ID by first getting all addresses for a locally saved ID from the Identity server, then getting all transactions for those addresses from the Wallet Server. The Wallet server aggregates all address (**AC2**) by querying all transaction events for all addresses using Ethereum events.

3.6.2. Identity Features Implementation

The identity features are described in Section 3.3.6. Address approval (**IC1**) for a Client goes through asynchronous calls to Identity Server using an Estonian ID authentication scheme. Identity Server uses SK DigiDocService to check if the Estonian ID is valid and for authentication (**IS1**). For a valid authenticated ID the Identity Server uses the Identity Admin key to approve the given Ethereum address (**IS2**) with a Ethereum transaction and then saves the ID to address mapping in the Identity database (**IS3**). The identity server provides looking up Ethereum address by ID (**IS4**) by simply querying the Identity database for the newest approved address for the ID.

3.6.3. Transaction Features Implementation

The transaction features are described in Section 3.3.7. Sending EUR2 to an address (**TC1**) happens by the client making a request to the Wallet server, which then does a delegate transfer on behalf of the client to the Ethereum address. Sending EUR2 to an ID (**TC2**) happens very similarly, except the Wallet server looks up the address for the ID from Identity Server. Receiving EUR2 to an address (**TC3**) is as simple as giving the sender a newly created Ethereum address, the address creator will need to approve the address before they can use received funds. Send to an ID (**TC4**) requires the client to make a request to Identity server to hold a transaction in escrow, in which Identity server creates a new temporary not-approved Ethereum address (**TS2**). When the ID recipient creates and approves a new address, Identity server will immediately send the funds to the new address and abandoning the temporary escrow address (**TS3**). Sending EUR2 to a EUR bank account is a special call to wallet server, which then creates a delegate transfer to the reserve bank, and communicates the receiving EUR bank account number on a private communication channel. Resolving an ID to an address (**TS1**) happens by asking Identity server. The Bank Reserve admin is responsible for executing valid EUR transactions (**TS4**) via an online banking interface. Executing valid EUR2 transactions happens by sending signed transactions to Ethereum accounts contract via the Ethereum node.

3.6.4. Reserve Features Implementation

The reserve features are described in Section 3.3.8. Converting EUR to EUR2 (**RC1**) starts by a user sending EUR to the reserve bank account (displayed in the Client). When the reserve bank receives the EUR bank transaction, it also receives the ID code of the sender. The Reserve Bank then asks the Identity server to get the Ethereum address (**RS1**) of the ID, then the reserve uses it's Reserve Admin key to increase supply by sending a transaction to the Reserve contract to create EUR2 (**RS2**), then the reserve creates a regular transaction to send the equivalent amount of EUR2 to the user address.

Converting EUR2 to EUR (**RC2**) happens very similarly. The client creates a transaction through Wallet server to the reserve with the bank details communicated secretly to the Reserve Bank. Then the Reserve Bank creates decreases the EUR2 using it's public key thereby destroying EUR2 (**RS3**), and then initiates a regular EUR bank transaction to the received bank details. Finally assigning reserve bank address (**RA1**) happens by signing a transaction to the reserve contract with the reserve admin key. Managing the reserve bank account (**RA2**) is provided by traditional online bank.

3.6.5. Enforcement Features Implementation

The enforcement features are listed in Section 3.3.9. Freezing an account (**EA1**) and seizing funds to close an account (**EA2**) are performed by signing an Ethereum transaction to the enforcement contract using the enforcement admin key calling the respective function. Assigning a new enforcement (**EA3**) is done by signing an Ethereum transaction with the enforcement admin key to the enforcement contract. Assigning a new designator (**EA4**) or assigning a new designator address (**EA5**) is done again by signing an Ethereum transaction with the designator admin key to call a function in the enforcement contract. Creating an AML database with transactions and IDs linked to addresses (**ES2**) requires securely importing the ID to address mapping from the Identity address then querying all transactions from the Ethereum blockchain and saving them in the AML database. Running AML checks on all names (**ES1**) can be done on the AML database with standard AML check procedures of normal banks. Opening up this AML and transaction to an analysis (**EA6**) is provided by a standard WEB UI to the AML database.

3.6.6. Convenience Features Implementation

The convenience features are listed in Section 3.6.6.

Wallet Security Features Implementation

Wallet security in the client is provided by standard AES encryption with a user chosen password of the private keys before storing in local storage (**CC1**). Similarly private keys are decrypted with AES decryption of the encrypted private keys with the user password (**CC2**).

Key Backup Features Implementation

Key backup happens with a cryptographic challenge created by the client (**CC3-7**) and administered by API calls to the Identity server (**CS1-3**). The exact details of the cryptographic challenge are not explained here or considered for the security analysis.

Transfer Details Features Implementation

Transfer details is described almost entirely in detail in the well in the Convenience Features Section. Transfer details encryption, posting, loading, and decryption for a transaction hash (**CC8-11**) happens on clients and the Transfer details server simply saves and loads the pairs of encrypted transfer details (**CS4-5**) for a given Ethereum transaction hash.

Account Recovery Features Implementation

Account recovery has not been yet implemented as a convenience method for clients and servers in the Euro 2.0 system. But, since the AccountRecovery contract is deployed on the blockchain any user could create a client or sign Ethereum transactions with the command line to this contract set up and utilize (**CC12-13**) an account recovery scheme (**CS6**).

Delegated Transfers Features Implementation

Delegated transfers are implemented by Clients making transfer via the Wallet Server API. The wallet server takes requests with the signed transactions from the client (**CC13**) and then repackages the request into an to Ethereum transaction signed by the Delegator admin key and to the Delegation contract to be executed.

3.7. Conclusion

3.7.1. Summary

In this chapter we made a best effort to communicate the features, architecture, and implementation of the proposed Euro 2.0 digital currency system. In Section 3.2 we gave an overview of the motivation behind the Euro 2.0 system and what goals it's trying to achieve. Section 3.3 listed the specific features of Euro 2.0 by domain. Section 3.4 gives a detailed overview of the Ethereum smart contracts and data comprising of the decentralized components of Euro 2.0. Section 3.5 gives a high level overview of the client server centralized components of the Euro 2.0 system and a very useful architectural diagram in Figure 2. Then Section 3.6 attempts to describe roughly how the centralized and decentralized components work together to implement the features. Together we have a good idea of what the Euro 2.0 system looks like to analyze its impact on stakeholders in Chapter 4 - Euro 2.0 Change Analysis and use as a basis of an information security risk analysis in Chapter 5 - Euro 2.0 Security Risk Analysis.

3.7.2. Future Work

Describing a complex, experimental, technical system in writing is extremely challenging. Future work would be avoidance of such task. Technical documentation can never come to the clarity of implemented software code. Any attempt describe a complex technical system in will sacrifice the clarity of code for misleading summaries and approximate drawings. Yet code is usually a disaster to understand without an extensive suite of unit and integration tests, especially in a prototype built through a series of casual hacking sessions. There are two future avenues of research.

First is to avoid diving into details in the code level and limit all analysis to formalized business process models of the system. Ethereum and client server components can be abstracted to goal models and business processes analyzed with formal models like Colored Petri Nets[48][40].

Second would be to reduce the scope of the security analysis altogether to just the Ethereum smart contracts. There is substantial research about smart contract vulnerabilities that could fuel a more focused and fruitful code level security analysis of the smart contracts involved with the Euro 2.0 system[12][13][28].

4. Euro 2.0 Change Analysis

4.1. Introduction

In this chapter we explore the social technical implications of the Euro 2.0 system on key stakeholders. We do this by exploring the research question:

> **RQ2:** *How does adoption of Euro 2.0 digital currency impact the relationship with money for key stakeholders: users, merchants, banks, and governments?*

In which we explore in the following sub research questions.

> **RQ2.1:** *What is the current relationship of money for stakeholders?*
>
> **RQ2.2:** *What changes in the relationship of money for stakeholders after adopting the Euro 2.0 digital currency?*
>
> **RQ2.3:** *What are the enabling and inhibiting factors influencing the transition to the Euro 2.0 digital currency?*

We approach the research questions design and UX influenced interpretation of the *Change Formula* from the field of Organization Development[27]. In Section 4.2 we define methods and terminology of the analysis, first the *Change Formula*, then our interpretation *Change Analysis*, and then define the stakeholders of Euro 2.0 system for the analysis. In Section 4.3 we answer **RQ2.1** by describing the current state of money for each stakeholder. Likewise in Section 4.4 we answer **RQ2.2** by describing the future state of money for each stakeholder under hypothetical use of the Euro 2.0 digital currency. Note in both 4.3 and 4.4 we use state and relationship interchangeably to correspond to *Change Analysis* terminology. In Section 4.5 we perform the change analysis for each stakeholder to answer **RQ2.3** describing potential enablers and inhibitors to change to a digital currency system. And finally we summarize our findings and future work in the conclusion, Section 4.6.

4.2. Analysis Preparation

4.2.1. Change Formula

The *Change Formula* was originally written on a chalkboard by scientist David Gleicher and stuck in the mind of organizational behavior expert Richard Beckhard[22]. The formula was first published in Sloan Management Review[16] and later in book by Beckhard and Harris[15]. Succinctly described in the review paper, Cady et al., *"The formula describes the conditions, that when met, will move an individual, group, or whole system in a direction of their choosing."* [22].

The improved version of the formula by Dannemiller[27] states that change will occur when the following condition is satisfied as represented by the qualitative equation:

$$D \times V \times F > R$$

- D : Dissatisfaction with the present situation
- V : Vision of what is possible
- F : First steps towards reaching the vision
- R : Resistance to change

If any one of D, V, or F is zero, i.e. neglected or overlooked, then the resistance to change will not be overcome and the intended behavioral change will not ensue.

We use the *Change Formula* in the context of Euro 2.0 to analyze the change of stakeholders using Euro 2.0 digital currency instead of Euro in the traditional monetary system. This is the background for analysis technique proposed in the following section.

4.2.2. Change Analysis

Change Analysis is a UX and User Design interpretation of the *Change Formula*. It is a framework to analyze how a new system will affect key stakeholders and avenues of suggest directions of future design research. The concepts of *Change Analysis* are depicted in Figure 3 and described below. Stakeholders refer to human or organizational actors interacting with the system.

Current State is the current state of the world for the stakeholders before the proposed change. This includes actions and functions stakeholders can do, motivations and psychological factors, as well as consequences and side-affects of these functions. In regards to the Change Formula, the current state includes the dissatisfaction D and provides a starting point for listing factors influencing first steps F and resistance R.

Future State is the state of the world after the proposed change. This also includes actions and functions stakeholders can do, motivations and psychological factors, as well as consequences and side-affects of these functions. In regards to the Change Formula, the future state is the vision of change V.

Enablers are the *Patterns*, *Habits*, *Behaviors*, and *Motivation* in the current state that would indicate stakeholders are ready to adapt the proposed change in the future state of the world. In regards to the Change Formula, enablers are those factors supporting first steps to the change F.

Inhibitors are the factors in the current state that would discourage stakeholders from the proposed change to the future state of the world including *Existing Policy Legislation*, *Parties with Vested Interest*, *Infrastructure and Technology*, and *Personal Barriers to Change*. In regards to the Change Formula, inhibitors are those factors contributing to resistance of change R.

The outcome of the change analysis is an understanding of how the change will impact the stakeholders, a set of inhibitors and enablers for that change, and further areas of design

research to validate or disprove the proposed hypotheses.

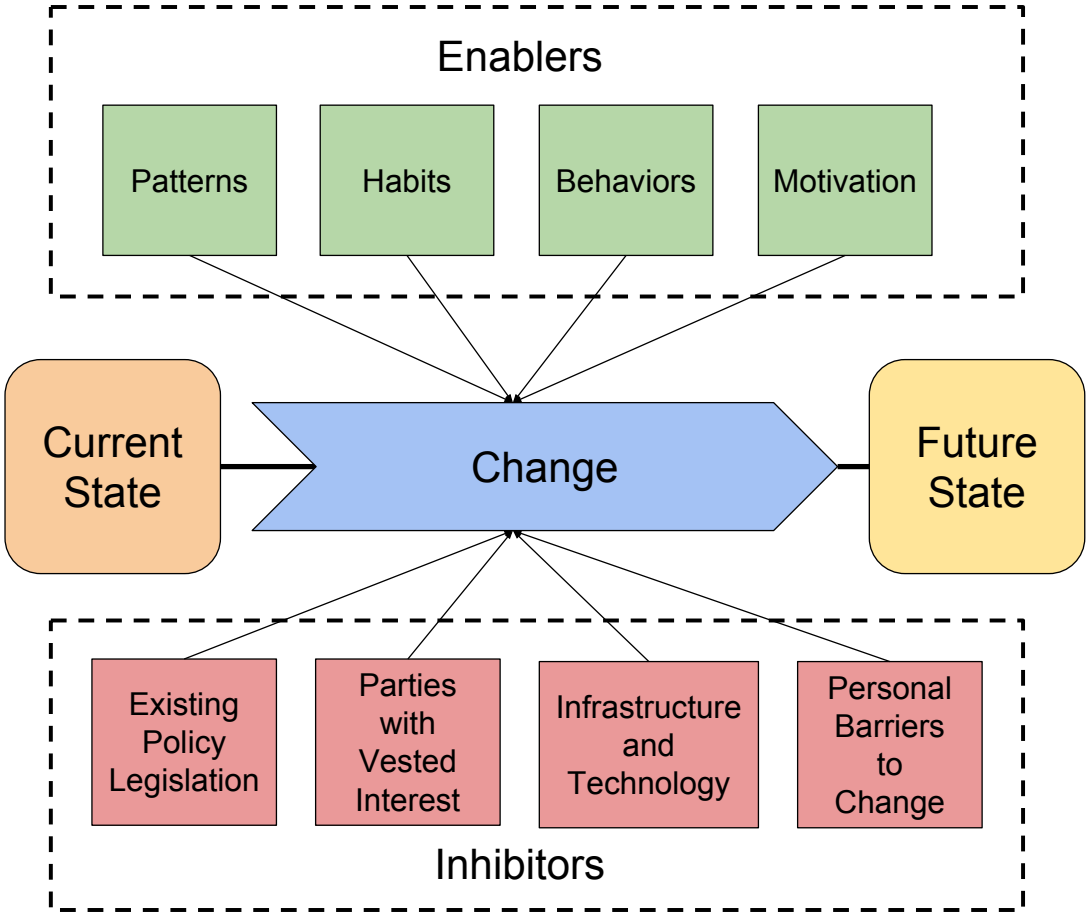

Figure 3. Change Analysis Overview

4.2.3. Key Stakeholders

The key stakeholders of the Euro 2.0 system are listed below.

Users are everyday users of financial systems sending and receiving transactions for personal use.

Merchants represent businesses users of financial system sending and receiving money for business use. In the context of this analysis we use merchants as the first use case representative of business entities.

Banks represent financial institutions in the financial system, including traditional banks and payment processors. For this analysis a bank is both a traditional bank making money off of holdings and a payment processor making money off of transactions.

Government represents the government entities in a financial system including the regulators defining the rules of a financial system, the central bank managing the economic value of the fiat currency, the law enforcement policing money, and the tax board collecting taxes.

4.3. Current State of Money

In this section we describe the *Current State* of money for each stakeholder.

4.3.1. Current State: Users

Users of the current monetary system have the following functions and use cases:

- Use bank transfers to make payments
- Use credit and debit cards to make payments
- Use bank accounts for short term and long term storage of funds
- Use cash for off record transactions
- Exchange money for goods and services
- Pay bills
- Settle small debts
- Save money
- Invest and grow money
- Borrow money
- Keep account numbers and access credentials secure

Users are influenced by the following psychological factors and behaviors:

- They want to keep their money
- They don't trust themselves with a significant amount of money
- They don't trust others with their money
- They believe in rules and repercussions of breaking the rules
- They like to outsource functions to experts
- They do what other people do
- They deal only with very small fragments of financial institutions

The current side affects and consequences of the monetary system for users are:

- They make money by working
- They make money by investing
- They lose money in transaction fees
- They lose money in financial crime
- They lose money in bad investments
- They lose money by spending
- They lose money by inflation
- They lose money by currency exchange rate movements

4.3.2. Current State: Merchants

Merchants in the current monetary system have the following functions and use cases:

- They accept payments for goods and services

- They accept cash payments
- They accept online payments
- They accept card payments
- They accept bank transfers
- They pay employees
- They invest money in the business
- They track money (accounting)
- They borrow money for the business

Merchants are influenced by the following psychological factors and behaviors:

- They need to cater to convenience of customers
- They want to make money
- They want to be ahead of their competitors
- They want to stay in business

The current side affects and consequences of the monetary system for merchants are:

- They make money from profit
- They lose money to reversible payments
- They lose money to transaction fees
- They lose money from money admin fees
- They lose money to opportunity cost of unusable money in transit
- They lose money by spending
- They lose money from inflation
- They lose money from currency fluctuations

4.3.3. Current State: Banks

Banks in the current monetary system have the following functions and use cases:

- They keep other peoples' money safe
- They make money with other peoples' money
- They keep track of other peoples' money
- They offer money through loans
- They facilitate transactions
- They hold balances of money
- They offer financial instruments to protect other peoples' money
- They verify and identify people for AML & KYC reasons
- They pay salaries of employees

Banks are influenced by the following psychological factors and behaviors:

- They want to make money
- They want to incentivize people to give them their money
- They want to keep regulators happy to stay in business

The current side affects and consequences of the monetary system for banks are:

- They make money from transactions
- They make money from account admin fees
- They make money from interests on loans
- They make money through investing other peoples' money
- They make money on hedging and other financial instruments
- They lose money on miscalculated hedges and backfiring financial instruments
- They lose money on bad investments
- They lose money on defaulted loans
- They lose money on fines
- They lose money on financial crime

4.3.4. Current State: Government

Government in the current monetary system has the following functions and use cases:

- It collects taxes of all citizens and businesses
- It pays salaries of employees
- It buys goods and services for the state
- It sets the monetary policies
- It tracks money
- It freezes assets of criminals
- It enforces policy through audits
- It issues documents used to verify identities
- It borrows money
- It loans money
- It collects fines
- It pays benefits
- It issues currency and tracks total currency in the system

Government is influenced by the following factors:

- It wants to keep the economy profitable for the country
- It wants to ensure the wellbeing of its citizens
- It wants to keep itself in power
- It wants to ensure the viability of its currency

The current side affects and consequences of the monetary system for government are:

- It loses money in tax evasion
- It loses money on defaulted loans

4.4. Future State of Money

In this subsection we look at the future state of money under adoption of Euro 2.0 system for stakeholders. In Section 4.4.1 we describe the key changes of the Euro 2.0 system on the relationship with money. Then we describe what changes for the stakeholders in the following sections Section 4.4.2, Section 4.4.3, Section 4.4.4, and Section 4.4.5.

4.4.1. Future State: Euro 2.0

The key changes imposed by the Euro 2.0 system are as follows:

- Transactions and balances are on a public ledger of Ethereum addresses
- Financial institutions are no longer needed to do transactions and store balances
- Publicly accessible decentralized computing network for transactions
- Reserve bank can convert back and forth between Euro and digital Euro
- Ability to send and receive money directly to Estonian ID codes
- Euros encoded as secret cryptographic keys
- All Ethereum addresses in the system are linked to real identities
- No money admin fees for holding assets in the system
- Irreversible transactions
- Money can be moved 24/7

The side effects of the Euro 2.0 system:

- All transactions must performed online
- Losing cryptographic keys loses access to funds (unless they are recovered or seized)
- Identifying an owner behind an Ethereum address reveals the entire balance and transaction history of that address for that owner

4.4.2. Future State: Users

What changes particularly for users in the Euro 2.0 system are:

- Less fees when financial intermediary is removed from transactions
- The good and bad side affects of transparency of money
- No money stuck in transit in pending transactions
- Simpler taxes
- Limited to transact with on record goods and services (no under the table cash transactions)

4.4.3. Future State: Merchants

What changes particularly for merchants in the Euro 2.0 system are:

- Less fees when financial intermediary is removed from transactions

- Every transaction is automatically on record
- The good and bad side affects of transparency of money
- No money stuck in transit in pending transactions
- Simpler taxes
- Limited to transact with on record goods and services (no under the table cash transactions)

4.4.4. Future State: Banks

What changes particularly for banks in the Euro 2.0 system are:

- No longer holding other people's money
- No longer collecting transaction fees
- Must invest in new service offerings to attract customers
- May run into severe lack of capital if enough people use the system

4.4.5. Future State: Government

What changes particularly for government in the Euro 2.0 system are:

- Easier to track money
- Can freeze and seize assets more easily
- Can collect taxes without a middleman or fees
- Enables realtime tax collection
- Eliminate tax evasion
- Accurate statistics of the state and amount of money in the entire monetary system

4.5. Changes Analysis

In this section we present the *Inhibitors* and *Enablers* of change for each stakeholder of the Euro 2.0 system.

4.5.1. Change Analysis: Users

Enablers

- Users are growing increasingly distrustful against banks and financial institutions after the 2008 financial crisis.
- Users are starting to consider and adopt Bitcoin and other cryptocurrencies for finances and investments.
- Users would like to reduce fees and increase the availability of their assets.
- Users are starting to turn more and more to cards and electronic forms of payment.

Inhibitors

- Users may not be ready to manage their own money.
- Users do not see how banks make money off of holding their financial assets and keep most of the return.
- Users are already comfortable with the existing financial system.
- Users will not adopt a new digital currency if the goods and services they purchase everyday do not accept the digital currency.
- Users don't understand the implications of a public ledger of transactions.
- Regulations may lag to protect users from crime using digital currency.

4.5.2. Change Analysis: Merchants

Enablers

- Merchants will save money accepting payment with low fees of digital currency over existing solutions.
- Merchants will save time with automation of accounting and taxes.
- Merchants will not have to worry about chargebacks and fraud risks of reversible payment methods.
- Merchants will not be affected by delays of pending payments.

Inhibitors

- Merchants may not move to the new system if users are not ready to use it.
- Merchants intentionally accepting cash for off the record transactions have no incentive to use a transparent system.
- Installing new technology or systems to accept payments has overhead costs.
- Accounting and tax firms will resist automation of their businesses.

4.5.3. Change Analysis: Banks

Enablers

- Banks show interest in using distributed ledger technology in intra bank processes.
- Banks want to offer convenience to their users.
- Banks are in the best position to offer a convenient interface for digital currency.

Inhibitors

- Resistance to accepting Bitcoin as valid payment method.
- Resistance to incorporate new technology into their processes.
- No longer profit from transaction fees.

- No longer profit from investing other peoples' money.
- Unwilling to have information about financials in the public ledger.
- Lack of trust in a public distributed network handling finance.

4.5.4. Change Analysis: Government

Enablers

- Governments are becoming more interested in digital currency.
- Governments are starting to regulate Bitcoin and other cryptocurrency through exchanges.
- Governments want to see more into the flow and use of money for tax and law enforcement reasons.

Inhibitors

- Banks may lobby Governments against adoption of a digital currency affecting their core business model.
- Government may not have the expertise or capacity to launch and maintain a digital currency like Euro 2.0.
- Government may not understand exactly the benefits or intricacies of a digital currency.
- Regulation may be too inflexible to allow for creation of a digital currency.

4.6. Conclusion

In this chapter we presented a change analysis of the stakeholders of the Euro 2.0 system. In Section 4.2 we presented the *Change Formula* and the interpreted a *Change Analysis* framework. In Section 4.3 we presented the current relationship of stakeholders with money, the current state in change analysis. In Section 4.4 we presented the predicted relationship of stakeholders with money using the Euro 2.0 system, the future state in change analysis. Then in Section 4.5 we presented the enabling and inhibiting factors for users to transition to the Euro 2.0 system, concluding the change analysis. Together, the sections of this chapter paint a picture of how the adoption of Euro 2.0 could affect key stakeholders.

The analysis presented in this chapter was based on the author's observations of the world of finance and cryptocurrencies from experience working in the financial technology space. The hypotheses and claims in this chapter and not supported with any previous academic research. Future work consists of adding substance to the presented hypotheses with previous academic research or validation with future user research, including the following:

- Research validating the current relationship of money for each of the stakeholders.
- Research user testing the Euro 2.0 prototype on users and merchants and learning from their experiences.
- Research presenting the Euro 2.0 to governments and banks and recording the findings.
- Research supporting or rejecting the hypotheses of the of the enablers and inhibitors to change presented in this chapter.

5. Euro 2.0 Security Risk Analysis

5.1. Introduction

After explaining the features and implementation details of the Euro 2.0 system in Chapter 3 - Euro 2.0 Digital Currency Prototype and exploring the change in relationship with money of stakeholders Chapter 4 - Euro 2.0 Change Analysis, we are ready to analyze the security risks of Euro 2.0 system. This chapter answers the research question:

> **RQ3:** *How do we assess the security risks of Euro 2.0 digital currency system and potential impact on stakeholders?*

In which we explore in the sub research questions:

> **RQ3.1:** *What are risk criteria for the digital currency consistent with the mission of the foundation?*
>
> **RQ3.2:** *What are the information assets and asset containers of the digital currency?*
>
> **RQ3.3:** *What are the threats and risks of the digital currency and consequences to stakeholders?*
>
> **RQ3.4:** *What are the mitigation strategies or alternative design suggestions for the Euro 2.0 system?*

RQ3.1 through *RQ3.4* are answered by applying Steps 1 through 8 of the OCTAVE Allegro risk assessment framework[23] to the system described in Chapter 3 - Euro 2.0 Digital Currency Prototype. Section 5.2 answers *RQ3.1* by establishing a risk measurement criteria consistent with the values of the Euro 2.0 foundation. Section 5.3 works through constructing information asset profiles and identifying information asset containers to answer *RQ3.2*. In Section 5.4 we perform the risk analysis by identifying areas of concern and threat scenarios to construct comprehensive information asset risk profiles, answering *RQ3.3*. Section 5.5 uses the risk profiles to categorize the risk analysis results and propose mitigation strategies and design suggestions answering *RQ3.4*. Finally from the above analysis we can make conclusions of the overall risks in Euro 2.0 to answer *RQ3* and also future research in Section 5.3.2.

The OCTAVE Allegro risk assessment was chosen for its simplicity and organization value based risk analysis methodology. The literature was interpreted for application to the Euro 2.0 System and Foundation assuming a hypothetical organization structure when the system is production ready. For the risk assessment, user represents both personal users and merchants in the Euro 2.0 digital currency.

5.2. Foundation and Risk Criteria

In this section we perform *Step 1 - Establish Risk Measurement Criteria* of the OCTAVE Allegro risk assessment. The main outcome of this section is a description of the theoretical organization structure of the Euro 2.0 foundation, its business objectives, and a set of impact areas with risk measurement criteria to be used as the basis of the risk analysis in subsequent sections.

5.2.1. Foundation Mission and Objectives

The Euro 2.0 Foundation is a non profit initiative to bring transparency and friction to the monetary system through digital currency. In order for the system to reach widespread adoption and financial sustainability, it must be completely taken over and supported by the government. Throughout the risk analysis we will refer to the *Euro 2.0 Foundation* as the government entity responsible for launching and maintaining this government backed digital currency. In the OCTAVE Allegro risk assessment, we interpret all references to Business and Organization to mean the Euro 2.0 Foundation.

The Euro 2.0 Foundation mission is to *"Reduce friction in transfer of value in the economy by unlocking the power of digital currency"*. It's objectives are:

- Provide the means to conveniently secure and use digital currency for users and merchants
- Avoid for profit financial intermediaries to reduce costs
- Be fully compliant with AML and CTF regulations
- Interoperate with the existing financial system

The organization is structured with the following functions:

Department	**Function**
Reserve Bank	Manage the conversion between EUR and EUR2 as well as the creation and destruction of EUR2.
Identity	Manage the ID to address mapping and approval of users to the system, communication with Estonian ID servers, as well as user key backups and escrow transactions.
Law Enforcement	Manage the AML proccesses, seizing, and freezing of funds.
Usability	Responsible for building and maintaining the clients to use the system, the convenience servers, the delegate transfer mechanism.
Smart Contracts	Responsible for building, deploying, and maintaining the Ethereum smart contract and full node used to to communicate with the Ethereum network. Also assists other departments in managing private admin keys.

Table 33. Euro 2.0 Organization Structure

5.2.2. Defining Risk Measurement Criteria

OCTAVE Allegro Step 1 Activity 2 guides the definition of a qualitative set of impact areas with risk measurement criteria to the foundation. Each impact area is a factor in the risk analysis in Section 5.5. The supplemented impact areas by the OCTAVE Allegro framework were used as a starting point for defining the impact areas specifically for Euro 2.0, although most of the resulting impact areas were created specifically for this risk assessment. Table 34 lists the final risk measurement impact areas:

Impact Area	**Symbol**
Reputation	R
User One Time Monetary Loss	M
Foundation One Time Financial Loss	F
Fines and Legal Penalties	L
Privacy - User Financial Information Revealed	P
Availability - Payments Network Unusable	A

Table 34. Risk Impact Areas

Each impact area has criteria for levels not applicable (N/A), low, medium, and high. The N/A level is not part of the OCTAVE Allegro methodology, but was added to this risk assessment so that final risk analysis rankings were not skewed by risks that don't affect the impact areas. The derivation of these impact areas and the their risk measurement criteria levels are described in more details below.

5.2.2.1 Ignored Impact Areas

OCTAVE suggested impact areas *Productivity* and *Safety and Health* were not included in the Euro 2.0 risk assessment. Productivity is not applicable since there is no clear measurable output of the Foundation based off of human hours. Safety and Health are not applicable since the Foundation does not have any physical proccesses or strenuous activities that reasonably need consideration in a security risk assessment.

5.2.2.2 (R) Reputation

Reputation is derived from the *Reputation and Customer Confidence* Allegro Worksheet 1. This impact area describes the loss of user confidence in the Euro 2.0 Foundation and subsequent difficulty in attracting users following a realized risk in this area. From this worksheet, Customer Loss was also considered to be included as an impact area rebranded as *Loss of User Acquisition Rate*, but was not included because of the lack of meaningful way to measure its current level and expected change as a realized risk, making it no different than reputation. Table 35 describes the levels for this impact area.

(R) Reputation	
N/A	No impact on reputation.
Low	Reputation is minimally affected. Little or no effort or expense is required to recover.
Medium	Reputation is damaged, and some effort and expense is required to recover.
High	Reputation is irrevocably destroyed or damaged.

Table 35. Reputation Risk Criteria

5.2.2.3 (M) User One Time Monetary Loss

User One Time Monetary Loss started as an *Other* fill in impact area in *Risk Measurement Criteria - Reputation and Customer Confidence* Allegro Worksheet 1. This impact area represents the concern of Euro 2.0 users about the safety of their money, an important aspect of a monetary system. Table 36 describes the levels for this impact area.

(M) User One Time Monetary Loss	
N/A	No user monetary loss, 0 EUR.
Low	User monetary loss less than 100 EUR.
Medium	User monetary loss between 100 and 1000 EUR inclusive.
High	User monetary loss greater than 1000 EUR.

Table 36. User One Time Monetary Loss Risk Criteria

The loss amounts in the risk criteria are arbitrary and for approximate qualitative representation. The Low level represents the loss of a single transaction or daily spending amount, the Medium level represents the loss of a monthly salary of spending, while the High level represents the loss of a savings account or the total account.

5.2.2.4 (F) Foundation One Time Financial Loss

Foundation One Time Financial Loss is the only prescribed impact area used from the *Risk Measurement Criteria - Financial* Allegro Worksheet 2. This impact area describes the one time monetary losses the Euro 2.0 Foundation is liable for based on realized risks. *Operating Costs* did not make sense to include due to the lack of financials about the projected operating costs of the Euro 2.0 Foundation. *Revenue Loss* doesn't make sense to include since this is currently a non-profit initiative without the intention of revenue. The risk measurement criteria for this impact area are described in Table 37.

(F) Foundation One Time Financial Loss	
N/A	No one time financial loss.
Low	One time financial loss of less than 1000 EUR.
Medium	One time financial loss between 1000 and 10000 EUR.
High	One time financial loss greater than 10000 EUR.

Table 37. Foundation One Time Financial Loss Risk Criteria

The loss amounts in the risk criteria are arbitrary and for approximate qualitative representation. The Low level represents the loss easily covered from a 50 to 100 EUR donation by 10 to 20 members of the Foundation. The Medium level represents a loss that would be significantly challenge for Foundation members requiring possibly a debt payed back over a few months. While the High level represents a possibly crippling loss to the organization, possibly resulting in bankruptcy.

5.2.2.5 (L) Fines and Legal Penalties

Fines and Legal Penalties came directly from the *Risk Measurement Criteria - Fines and Legal Penalties* Allegro Worksheet 3. This impact area describes the monetary losses from penalties the Euro 2.0 Foundation would be liable for from realized risks. *Fines* and *Lawsuits* from the worksheet were combined into one impact area in this risk assessment since there is not enough existing knowledge to really distinguish the two impact areas in risk profiling. The risk measurement criteria for this impact area are described in Table 38.

(L) Fines and Legal Penalties	
N/A	No legal financial loss.
Low	Legal financial loss of less than 1000 EUR.
Medium	Legal financial loss between 1000 and 10000 EUR.
High	Legal financial loss of user greater than 10000 EUR.

Table 38. Legal - Fines and Legal Penalties Risk Criteria

The amounts are arbitrary and derived from the same reasoning as *Foundation One Time Financial Loss*.

5.2.2.6 (P) - Privacy - User Financial Information Revealed

Privacy - User Financial Information Revealed is a user defined impact area from *Risk Measurement Criteria - User Defined* Allegro Worksheet 6. The risk criteria qualifies the amount of customers with personally identifiable financial information disclosed. Users knowing about their financial data disclosed are likely to encourage other users to not use the Euro 2.0 system. The risk criteria are explained in Table 39.

(P) Privacy - User Financial Information Revealed	
N/A	No users with disclosed personally identifiable financial information.
Low	A few users (less than 10) with disclosed personally identifiable financial information.
Medium	Many users (between 10 and 1000) with personally identifiable financial information.
High	All users in the system with disclosed personally identifiable financial information.

Table 39. Privacy - User Financial Information Revealed Risk Criteria

5.2.2.7 (A) - Availability - Payments Network Unusable

Availability - Payments Network Unusable is a user defined impact area from *Risk Measurement Criteria - User Defined* Allegro Worksheet 6. The risk criteria qualifies the level of disruption to payment operations of users in the system. Users who cannot use the payments network will quickly lose trust in it for their financial needs. The risk criteria are explained in Table 40.

(A) Availability - Payments Network Unusable	
N/A	No disruption to usability of network (1-60 seconds per payment operation).
Low	Some disruption, 1 to 5 minute delay for payment operations.
Medium	Noticeable disruption, 5 to 60 minute delay for payment operations.
High	Severe disruption, more than 60 minute delay for payment operations.

Table 40. Availability - Payments Network Unusable Risk Criteria

5.2.3. Ranking Impact Areas

Activity 2 of Step 1 of OCTAVE Allegro prioritizes impact areas determined in Activity 1. Table 41 shows the ranked impact areas of Euro 2.0 ranked with 6 as the highest and 1 as the lowest.

Ranking	Abbreviation	Name
6	M	User One Time Monetary Loss
5	R	Reputation
4	P	Privacy - User Financial Information Revealed
3	A	Availability - Payments Network Unusable
2	F	Foundation One Time Financial Loss
1	L	Fines and Legal Penalties

Table 41. Impact Areas Ranking

The most important risk criteria are those that are significant detractors to user growth of the new digital currency. The financial livelihood of the user (**M**) is at the top of the list. A realized risk resulting in monetary loss of the user will likely lose the user and negative virality (user telling his friends not to use Euro 2.0) will prevent other users from joining. The reputation of the organization and monetary confidence (**R**) is next in the line as a realized risk affecting reputation discourages new users from joining the system. Next is the privacy of users (**P**) which also encourages a negative user experience and detracts growth if significantly breached. Next is the availability (**A**) of the system. Finally, the least important of the risk criteria, is the financial losses of the organization (**F**) and legal fines and legal penalties (**L**). Mistakes are likely to happen, as long as the foundation is still alive, there is more value at the start to ensure user happiness and growth.

5.3. Information Assets and Containers

In this section we define the information assets and asset containers following *Step 2 - Develop an Information Asset Profile* and *Step 3 - Identify Information Asset Containers* of the OCTAVE Allegro Risk Assessment. These information assets and asset containers are the main subject of risk profiles constructed in Section 5.4 which are analyzed and mitigated in Section 5.5. First we walk through defining the possible information assets of Euro 2.0. Next we narrow down to a *critical few* information assets to be the subject of the risk analysis. Then for each critical information asset, we build a full profile with information asset containers.

5.3.1. Enumerating Information Assets

An *Information Asset* is any information or data that can be valuable to the foundation existing in either electronic or physical form. Enumerating information assets can be done by answering the following questions[23]:

- *What information assets are of most value to your organization?*
- *What information assets are used in day-to-day work processes and operations?*
- *What information assets, if lost, would significantly disrupt your organization's ability to accomplish its goals and contribute to achieving the organization's mission?*
- *What other assets are closely related to these assets?*

In which we enumerate the following information assets for Euro 2.0 in Table 42:

Information Asset	Description
Reserve Bank Account Access	Used to access funds sent to and from the reserve bank account.
Ethereum Contract Data: Status Balance Delegate Transfer Nonce Recovery Account Total Supply	All of the decentralized data of the Euro 2.0 system.
Ethereum Admin Keys: Master Account Key Reserve Key Account Approver Key Enforcement Key Designator Key Delegate Key	The keys to all the special admin functions of the Euro 2.0 system.
DB Credentials: ID AML Transfer details Bank Gateway	The credentials to the traditional centralized databases of the Euro 2.0 system.
Customer Keys	Ethereum address private keys for the customer to access their money.
Customer Data: Sender and Receiver Identities Reserve Bank Transaction Data Backup Keys ID $\Leftrightarrow$ Address Mapping Escrow Data Bank Transaction Data	Various customer data.

Table 42. Euro 2.0 Possible Information Assets

5.3.2. Selecting Critical Information Assets

The *critical few* information assets are those that have a significant adverse affect on the organization if disclosed, modified, lost, destroyed, or access interrupted. From the list of enumerated information is derived the critical information assets:

(I) User Money

(II) Ethereum Admin Keys

(III) User Identity

These are chosen as critical information assets to be representative of the majority of information assets enumerated in Table 42. The critical information assets are described in more detail in Table 44, Table 46, and Table 48. Limiting the number of critical information assets reduces the number of risk profiles in Section 5.4 and simplifies the complexity of the risk analysis in Section 5.5. *Ethereum Contract Data* and *User Keys* were also considered as critical information assets but are adequately represented by *User Money*. *User Transfer History* was also considered as a critical information asset but is related enough to *User Identity* to be disregarded.

5.3.3. Information Asset and Containers Analysis

The following subsections describe in detail the critical information asset profiles and information assets containers for I. User Money, II. Ethereum Admin Keys, and III. User Identity that will be the basis of enumerating risk profiles in Section 5.4 .

Allegro Worksheet 8 summarizes the key aspects of a *Critical Information Asset Profile* filled out during *Step 2 - Develop Information Asset Profile*. The Asset Profile includes information about the asset, rationale for selection, organizational description, owners, and security requirements. The security requirements characterize how an information asset is to be protected and are key for evaluating impact of risks. Security requirements of information assets are explained in Table 43.

Security Requirement	**Description**
Confidentiality	Ensuring that only authorized people or systems have access to the information asset
Integrity	Ensuring that an information asset remains in the condition that was intended by the owner and for the purposes intended by the owner.
Availability	Ensuring that the information asset remains accessible to authorized users.

Table 43. Security Requirements of Information Assets

Information Asset Containers are where information assets are stored, transported, or processed. *"In an information security risk assessment, the identification of containers is essential to identifying risks to the information asset itself."* They are they key points of vulnerability and threats as well as the key points to apply mitigations and preventive measures for an information asset. An information asset containers can be technical, physical, or people and internal or external to the organization. In the context of Euro 2.0 risk analysis we focus mainly on technical and people asset containers. Three important points about the security of information assets with regards to information asset

containers:

1. The **way** in which an information asset is protected or secured is through controls implemented an the container level.
2. The **degree** in which an information asset is protected or secured is based on how well the controls are implemented at the container level.
3. Any **vulnerabilities** and **threats** to the container in which the information asset lives are inherited by the information asset.

The information asset containers for the critical information assets were identified by working through OCTAVE Allegro *Step 3 - Identify Information Asset Containers* and listed in the description of critical information asset profiles in subsections 5.3.4, 5.3.5, and 5.3.6.

5.3.4. Critical Asset I: User Money

(1) Critical Asset	User Money	
(2) Rationale for Selection	User money is the most important information asset for users of a digital currency and must be secured. Discrepancy of balance of money undermines the entire usefulness of Euro 2.0. This asset is definitely subject to regulatory requirements which are out of scope of this thesis.	
(3) Description	This is all forms of money, including the EUR and EUR2 holdings of users, merchants, and other admin accounts as they enter, reside, and leave the Euro 2.0 system.	
(4) Owners	Ultimately ownership of user, merchant, and admin funds are the respective parties. While EUR2 is being held and transacted in the system, an equivalent amount of EUR is held and shared ownership by the Reserve Bank.	
(5) Security Requirements	**Confidentiality**	Only the owner of funds should be able to use them. By design the amount of funds is not confidential but the owner of the funds should be (which is critical Asset III).
	Integrity	100%. This asset should never be changed against rules of the currency.
	Availability	Ideally available more than 99% of the time, but not super important.
(6) Most Important Security Requirement	**Integrity**. Money being created and destroyed arbitrary completely undermines the system.	

Table 44. User Money Asset Profile

Container	Type	Owners	Description
Wallet Server	Technical Internal	Convenience	The wallet service performs the EUR2 delegate transfer operation.
Identity Server	Technical Internal	Identity	The Identity server currently handles escrow of funds to an ID and addresses and keys backup.
Bank Gateway Server	Technical Internal	Reserve Bank	The bank gateway server handles the conversion from EUR to EUR2.
Ethereum Node	Technical Internal	Smart Contracts	This node is the interface to all the operations with the Ethereum network.
User Client	Technical External	User Convenience	User device to interact with the Euro 2.0 system.
Ethereum Smart Contracts	Technical External	No Owner Smart Contracts	Deployed smart contracts on Ethereum blockchain that hold all of the decentralized balance and account data and operations in the system.
Reserve Bank Account	Technical External	External Bank Reserve Bank	Bank account used for all EUR operations.
Reserve Bank Account Admin	People Internal	Reserve Bank	Bank account used for all EUR operations.
Internet	Technical External	No owner	Communication channel for all communication between all components.

Table 45. User Money Asset Containers

5.3.5. Critical Asset II: Ethereum Admin Keys

(1) Critical Asset	The Ethereum Admin keys are those Ethereum address keys that control the critical functionality inside the Euro 2.0 smart contracts.	
(2) Rationale for Selection	The core of the monetary system is controlled with these keys. Deploying and changing new contracts and rules of how data can be read and written. A compromised admin key could prove devastating for the entire system.	
(3) Description	These are electronically generated cryptographic keys. The admin keys and functions include: *CryptoFiat Master* - can deploy and upgrade contracts *Reserve* - can manage supply, create and destroy money *Approver* - can approve addresses *Delegate* - can sign delegate transfers on behalf of wallet server *Enforcement* - can freeze accounts and seize funds *Designator* - can access seized funds	
(4) Owners	Each key is owned by a different department of the Foundation: *CryptoFiat Master* - Smart Contracts *Reserve* - Reserve Bank *Approver* - Identity *Delegate* - Convenience *Enforcement* - Law Enforcement *Designator* - Law Enforcement	
(5) Security Requirements	**Confidentiality**	This asset must be kept absolutely 100% confidential.
	Integrity	Keys don't work if they are modified so this is quite important.
	Availability	Approver, Delegate, and Reserve keys need to be available most of the time (99%) for automatic operations. Enforcement and Designator only need to be available during rare operations. CryptoFiat master only needs to be available on contract deploys.
(6) Most Important Security Requirement	**Confidentiality**. Without a question a breach of confidentiality of the keys is devastating to the digital currency.	

Table 46. Ethereum Admin Keys Asset Profile

Container	Type	Owners	Description
Key Generating Device	Technical Internal	Smart Contracts	The device used to generate an Ethereum admin key and address.
Key File	Technical Internal	Key Owner	File used to store Ethereum private key.
Physical Backup of Key	Physical Internal	Key Owner	Physical backup of admin key.
Key Encryption Password	People Internal	Key Owner	Password used to encrypt and decrypt a key file.
CryptoFiat Master Device	Technical Internal	Smart Contracts	Device holding the CryptoFiat master key file.
CryptoFiat Master Transactor	Technical Internal	Smart Contracts	Device used to broadcast master operations to Ethereum network.
Law Enforcement Device	Technical Internal	Law Enforcement	Device holding the Law Enforcement key file.
Designator Device	Technical Internal	Law Enforcement	Device holding the Law Enforcement key file.
AML Server	Technical Internal	Law Enforcement	Device used to broadcast Law Enforcement and Enforcer operations to the Ethereum network.
Bank Gateway	Technical Internal	Reserve	Device holding the reserve key file used to sign transactions and do operations for the reserve.
Identity Server	Technical Internal	Identity	Server holding the approver key file needed to approve addresses for authorized IDs.
Wallet Server	Technical Internal	Convenience	Server holding the delegator key file needed to sign delegate transactions on behalf of users.
Ethereum Node	Technical Internal	Smart Contracts	This node is the interface to all the operations with the Ethereum network.

Table 47. Ethereum Admin Keys Asset Containers

5.3.6. Critical Asset III: User Identity

(1) Critical Asset	User Identity in any format in the Euro 2.0 system.	
(2) Rationale for Selection	If the Identity of a user is linked to their Ethereum address, all transaction history is visible on the Ethereum blockchain under the current design.	
(3) Description	Identity refers to ID code and any other personally identifiable information.	
(4) Owners	The User and partially the Identity department.	
(5) Security Requirements	**Confidentiality**	Highly confidential, only the Identity server, owning user, and other users that have been told that ID code.
	Integrity	The ID should be exactly as is for the Identity server or user approving an address.
	Availability	Only needs to be available when user looks up an address by ID or approves an address.
(6) Most Important Security Requirement	**Confidentiality**. User transaction history is public in the current design if identity is disclosed.	

Table 48. User Identity Asset Profile

Container	Type	Owners	Description
Identity Server	Technical Internal	Identity	Manages ID authentication and approval of Ethereum addresses for users.
Identity Database	Technical Internal	Identity	Database storing ID to Ethereum address mapping, escrow, and keys backups.
AML Server	Technical Internal	Law Enforcement	Server providing AML operations to Law Enforcement.
AML Database	Technical Internal	Law Enforcement	Database storing transactions, addresses, and IDs for AML analysis.
Ethereum Node	Technical Internal	Smart Contracts	This node is the interface to all the operations with the Ethereum network.
User Client	Technical External	User Convenience	User's device to interact with the Euro 2.0 system.
Ethereum Smart Contracts	Technical External	No Owner Smart Contracts	Deployed smart contracts on Ethereum blockchain that hold all of the decentralized balance and account data and operations in the system.
SK ID LDAP	Technical External	SK ID Solutions AS	Directory lookup for information based on an Estonian ID number.
SK DigiDocService	Technical External	SK ID Solutions AS	Estonian ID authentication services.
People	People External	Anyone	Knowledge of ID numbers can be found in many external documents and proccesses in Estonia.

Table 49. User Identity Asset Containers

5.4. Risk Profiles and Analysis

In this section we perform a risk analysis on the critical information assets and containers of Section 5.3. The output of this section is multiple *Information Asset Risk Profiles* for each critical information asset, the results of performing *Step 5 - Identify Threat Scenarios*, *Step 6 - Identify Risks*, and *Step 7 - Analyze Risks* of the OCTAVE Allegro methodology. Section 5.4.1 gives an overview of the methodology used to construct the profiles and the subsequent sections present the completed *Information Asset Risk Profiles* for Risk Profiles I: User Money, Risk Profiles II: Ethereum Admin Keys, and Risk Profiles III: User Identity.

5.4.1. Risk Analysis Methodology

The goal of the risk analysis is to enumerate the most relevant risks for critical information assets. A *risk* is the possibility of suffering harm or loss to the foundation or users, the combination of a *threat* (condition), an *impact* (consequence), and an *probability* (uncertainty) of the risk being realized. The *Information Asset Risk Profiles* resulting from the risk analysis will have the structure outlined in Table 50.

Property	Description
(1) Threat Scenario	A descriptive statement of a situation that could affect an information asset.
(2) Actor	Who would exploit the threat?
(3) Means	How would the actor exploit the threat?
(4) Motive	What would be the actor's incentive to exploit the threat?
(5) Outcome	What would the resulting effect be on the information asset? ■ Disclosure ■ Modification ■ Destruction ■ Interruption
(6) Security Requirements	How would the information asset's security requirements be breached?
(7) Probability	What is the likelihood this scenario could occur? ■ High ■ Medium ■ Low
(8) Consequences	What are the consequences to the organization or information asset owner as an outcome of the breached security requirements?
(9) Severity	How severe are the consequences to the organization or asset owner by impact area?

Table 50. Information Asset Risk Profile Structure

In the OCTAVE Allegro methodology *Threat Scenario (1)* is also known as an *Area of Concern*. They are synonymous, both a statement describing a potential undesirable situation that can affect the security criteria of an information asset. A threat describes the scenario more systematically with properties (2) through (5). *Areas of concern* are conceived in *Step 4 - Identify Areas of Concern* from intuition of the risk analyst. The *Areas of Concern* act as seeds to systematically develop *Threats* in *Step 5 - Identify Threat Scenarios* by considering *Threat Trees* and *Threat Scenario* questionnaires for each type of information asset container. The outcome of the exercise is to fill in properties (1) through (7) in the *Information Asset Risk Profile* in Table 50. *Step 6 - Analyze Risks* provides guidance for documenting the consequences, property (8), of the realized risk on the organization and information asset owners.

Finally in *Step 7 - Analyze Risks*, a qualitative *impact value* is assigned to describe the extend of the impact to the organization the consequences of a realized risk for an information asset. The impact value is computed for every risk profile in a risk analysis matrix using the impact areas defined in Table 34 with the following columns:

- *Impact Area* - the impact area for the described by the row
- *Ranking* - the ranking of the impact area described in Table 41
- *Impact Value* - probability of the threat scenario occurring with weight: N/A (0), Low (1), Medium (2), High (3)
- *Score* - the ranking multiplied by the impact value

Finally the total impact value for the risk profile is computed as the some of the scores for

each row. These impact scores are used to prioritize and categorize the threat scenarios for mitigation approaches in Section 5.5.

5.4.2. Risk Profiles I: User Money

The risk profiles presented in this section are for the critical information asset I. User Money described in Section 5.3.4.

5.4.2.1 (1) Keys Stolen from User Device

Property	Description
(1) Threat Scenario	Attacker steals user private key from client application and sends all EUR2 to an account of his choice stealing the funds.
(2) Actor	Attacker wishing to steal EUR2 for financial gain.
(3) Means	■ The attacker could use a phishing attack to install malware that grabs the keys when they're unencrypted. ■ An attacker could make a fake application client to get the user to sign away his/her funds.
(4) Motive	Financial gain.
(5) Outcome	■ Disclosure of the user private keys to a fraudster. ■ Interruption of the service of funds to the user.
(6) Security Requirements	Confidentiality and Availability of the funds are breached since they are no longer accessible.
(7) Probability	High
(8) Consequences	■ User loses all of his money in the EUR2 system. ■ User will tells his friends that EUR2 is insecure for storing money.

Table 51. Risk Profile: Keys Stolen from User Device

Impact Area	Description	Ranking	Value	Score
M	User One Time Monetary Loss	6	High (3)	18
R	Reputation	5	High (3)	15
P	User Financial Information Revealed	4	Low (1)	4
S	Payments Network Unusable	3	N/A (0)	0
F	Foundation One Time Financial Loss	2	N/A (0)	0
L	Fines and Legal Penalties	1	Medium (2)	2
			TOTAL	39

Table 52. Severity: Keys Stolen from User Device Severity

5.4.2.2 (2) Programming Error in the Ethereum Contract

Property	Description
(1) Threat Scenario	Ethereum smart contract has a programming error that allows attacker to create counterfeit EUR2.
(2) Actor	Attacker with high knowledge of Ethereum smart contract exploits.
(3) Means	Attacker find vulnerability in Accounts contract to increase a balance without restriction.
(4) Motive	Use falsely created EUR2 to buy goods and services or cash out from the reserve into EUR.
(5) Outcome	Modification of the balances against the rules of the system.
(6) Security Requirements	Integrity is compromised as the attacker's EUR2 balance does not represent a real world counterpart of EUR. Availability may be compromised if system vulnerability cannot be fixed before exploitation is contained.
(7) Probability	Medium
(8) Consequences	▪ Foundation will be liable for difference of real and falsified EUR2 and may charged with fines and penalties. ▪ Entire Euro 2.0 system may need to be shutdown if vulnerability is exploited enough and funds cannot be frozen before being used.

Table 53. Risk Profile: Programming Error in the Ethereum Contract

Impact Area	Description	Ranking	Value	Score
M	User One Time Monetary Loss	6	High (3)	18
R	Reputation	5	High (3)	15
P	User Financial Information Revealed	4	N/A (0)	0
S	Payments Network Unusable	3	Medium (2)	6
F	Foundation One Time Financial Loss	2	High (3)	6
L	Fines and Legal Penalties	1	High (3)	3
			TOTAL	48

Table 54. Severity: Keys Stolen from User Device Severity

5.4.2.3 (3) Spoof ID to Ethereum Address Lookup

Property	Description
(1) Threat Scenario	An attacker spoofs the process of looking up an Ethereum address for an ID and replaces it with his own.
(2) Actor	Outside attacker with high knowledge of computer networking.
(3) Means	Attacker sets up man in the middle attack for API calls going to Identity server from outside sources and replaces the response with his choice.
(4) Motive	Financial gain by forcing all transfers to an ID (including newly created EUR2) be sent to an address under his control instead of intended recipient address.
(5) Outcome	■ Modification of intended recipient of funds. ■ Interruption of the service of funds transfer to legitimate users.
(6) Security Requirements	Confidentiality and Availability. Only the intended recipient of funds should be able to use them.
(7) Probability	Medium
(8) Consequences	■ Recipients don't get paid for goods and services. ■ Users lose EUR2 intended to be for themselves. ■ Foundation may be liable for losses. ■ Fines and lawsuits may ensue for bad security measures.

Table 55. Risk Profile: Spoof ID to Address Lookup

Impact Area	Description	Ranking	Value	Score
M	User One Time Monetary Loss	6	High (3)	18
R	Reputation	5	High (3)	15
P	User Financial Information Revealed	4	Medium (2)	8
S	Payments Network Unusable	3	High (3)	9
F	Foundation One Time Financial Loss	2	High (3)	6
L	Fines and Legal Penalties	1	High (3)	3
			TOTAL	59

Table 56. Severity: Spoof ID to Address Lookup

5.4.2.4 (4) User Keys Stolen from Backup System

Property	Description
(1) Threat Scenario	An attacker breaks the challenge of backup system and steals user keys.
(2) Actor	Outside attacker with knowledge of cryptography.
(3) Means	Attacker queries Identity server with ID codes acquired from outside source, gets challenge information, then breaks the challenge to get encrypted keys, and uses a dictionary attack to find the password to decrypt the keys.
(4) Motive	Financial gain by stealing to user funds.
(5) Outcome	■ Disclosure of user private keys. ■ Interruption of access of funds to user with lost keys.
(6) Security Requirements	Confidentiality and Availability of users funds.
(7) Probability	Low
(8) Consequences	■ User loses his funds, leaves the system, and tells his friends Euro 2.0 cannot be trusted. ■ Foundation loses trust of users.

Table 57. Risk Profile: User Keys Stolen from Backup System

Impact Area	Description	Ranking	Value	Score
M	User One Time Monetary Loss	6	High (3)	18
R	Reputation	5	High (3)	15
P	User Financial Information Revealed	4	Low (1)	4
S	Payments Network Unusable	3	N/A (0)	0
F	Foundation One Time Financial Loss	2	Medium (2)	4
L	Fines and Legal Penalties	1	Medium (2)	2
			TOTAL	43

Table 58. Severity: User Keys Stolen from Backup System

5.4.2.5 (5) Ethereum Scalability

Property	Description
(1) Threat Scenario	Ethereum decentralized network can't keep up with the volume of transactions in the the system and delays transactions.
(2) Actor	No intentional actor.
(3) Means	The Ethereum foundation does not solve scalability issues and cannot handle all the activity of all decentralized applications. Transactions are either significantly delayed or never picked up by miners.
(4) Motive	No motive for no actor.
(5) Outcome	Interruption or delays in processing transactions.
(6) Security Requirements	Availability of users funds is compromised if network can no longer keep up.
(7) Probability	Low
(8) Consequences	■ Transactions are delayed because of difficult for miners to pick them up. ■ Euro 2.0 becomes unusable for payments because of low availability. ■ Foundation reputation suffers because it can no longer fulfills objectives. ■ Potential fines and lawsuits for lost business.

Table 59. Risk Profile: Ethereum Scalability

Impact Area	Description	Ranking	Value	Score
M	User One Time Monetary Loss	6	N/A (0)	0
R	Reputation	5	High (3)	15
P	User Financial Information Revealed	4	N/A (0)	0
S	Payments Network Unusable	3	High (3)	9
F	Foundation One Time Financial Loss	2	N/A (0)	0
L	Fines and Legal Penalties	1	Medium (2)	2
			TOTAL	26

Table 60. Severity: Ethereum Scalability

5.4.3. Risk Profiles II: Ethereum Admin Keys

The risk profiles presented in this section are for the critical information asset II. Ethereum Admin Keys described in Section 5.3.5.

5.4.3.1 (1) Steal Admin Key from Online Server

Property	Description
(1) Threat Scenario	Attacker finds vulnerability on server with admin key and: ■ Steals Approver key to approve accounts not linked to a real identity to aid in criminal money. ■ Steals Reserve key to create EUR2 without a EUR counterpart. ■ Steals Delegate key to steal fees.
(2) Actor	Attacker with system infiltration knowledge.
(3) Means	Find and exploit a server vulnerability on public internet and steal the key from the local filesystem.
(4) Motive	Financial gain and aid criminal activity by accessing addresses not linked to an ID.
(5) Outcome	Disclosure of confidential admin keys.
(6) Security Requirements	Confidentiality of admin keys.
(7) Probability	Medium
(8) Consequences	■ Addresses can be approved and transact without a linked real identity. ■ Can create EUR2 not backed by EUR creating financial loss for the Foundation. ■ Delegation fees stolen. ■ Reputation of organization affected by news. ■ Fines and penalties from allowing transactions without proper identification.

Table 61. Risk Profile: Steal Admin Key from Online Server

Impact Area	Description	Ranking	Value	Score
M	User One Time Monetary Loss	6	High (3)	18
R	Reputation	5	High (3)	15
P	User Financial Information Revealed	4	N/A (0)	0
S	Payments Network Unusable	3	Medium (2)	6
F	Foundation One Time Financial Loss	2	High (3)	6
L	Fines and Legal Penalties	1	High (3)	3
			TOTAL	48

Table 62. Severity: Steal Admin Key from Online Server

5.4.3.2 (2) Tamper CryptoFiat Key Generation

Property	Description
(1) Threat Scenario	An attacker tampers with the key generation process and compromises the CryptoFiat master key to secretly deploy a new contract with backdoors into the entire system.
(2) Actor	A clever attacker with insight on key generation malware and smart contracts.
(3) Means	Use social engineering or physical break in to implant virus on key generators computer.
(4) Motive	Create backdoors to the entire system for unlimited financial gain.
(5) Outcome	Disclosure of confidential master admin key.
(6) Security Requirements	Confidentiality of master admin key is compromised.
(7) Probability	Low
(8) Consequences	■ Every aspect of the system can be compromised when attacker gains the ability to deploy contracts with access to Euro 2.0 balances. ■ Reputation of foundation is gone and users no longer trust the system with their finances. ■ Fines and penalties for weak security practices.

Table 63. Risk Profile: Tamper Key Generation

Impact Area	Description	Ranking	Value	Score
M	User One Time Monetary Loss	6	High (3)	18
R	Reputation	5	High (3)	15
P	User Financial Information Revealed	4	N/A (0)	0
S	Payments Network Unusable	3	High (3)	9
F	Foundation One Time Financial Loss	2	High (3)	6
L	Fines and Legal Penalties	1	High (3)	3
			TOTAL	51

Table 64. Severity: Tamper CryptoFiat Key Generation

5.4.3.3 (3) Bribe Key Custodian

Property	Description
(1) Threat Scenario	An attacker gains access to admin keys by bribing key custodians.
(2) Actor	A nation state or global corporation with a great amount of resources.
(3) Means	Bribe Key Custodian with an offer they can't refuse for access to the keys.
(4) Motive	Will to sabotage the Euro 2.0 system for financial or political gain.
(5) Outcome	Disclosure of confidential master key and Destruction of the system through malicious activity.
(6) Security Requirements	Confidentiality of an admin key is compromised and Availability of the system could be at stake.
(7) Probability	Low
(8) Consequences	■ Every aspect of the system can be compromised when attacker gains the ability to deploy contracts with access to Euro 2.0 balances. ■ Reputation of foundation is gone and users no longer trust the system with their finances. ■ Fines and penalties for weak security practices.

Table 65. Risk Profile: Bribe Key Custodian

Impact Area	Description	Ranking	Value	Score
M	User One Time Monetary Loss	6	High (3)	18
R	Reputation	5	High (3)	15
P	User Financial Information Revealed	4	N/A (0)	0
S	Payments Network Unusable	3	High (3)	9
F	Foundation One Time Financial Loss	2	High (3)	6
L	Fines and Legal Penalties	1	Medium (2)	2
			TOTAL	50

Table 66. Severity: Bribe Key Custodian

5.4.4. Risk Profiles III: User Identity

The risk profiles presented in this section are for the critical information asset III. User Identity described in Section 5.3.6.

5.4.4.1 (1) Identity or AML Database Hacked

Property	Description
(1) Threat Scenario	An attacker finds a security vulnerability in the Identity or AML server and gains access to database.
(2) Actor	An attacker with knowledge of server exploits.
(3) Means	Attacker exploits a security vulnerability in the Identity or AML server to obtain access to the database.
(4) Motive	The ID to address mapping allows an attacker to link all transactions on the Ethereum blockchain to real identities and sell this information to interested parties or publish the mapping online as an act of cyberterrorism.
(5) Outcome	Disclosure of user identities and their mapping to Ethereum blockchain addresses.
(6) Security Requirements	Confidentiality of the Identity of user transactions is breached.
(7) Probability	Medium
(8) Consequences	▪ Identity of all users' transactions is compromised and published for attacker gain. ▪ The Foundation loses the trust of its users with their money. ▪ The Foundation may receive fines and penalties for weak security practices.

Table 67. Risk Profile: Identity or AML Database Hacked

Impact Area	Description	Ranking	Value	Score
M	User One Time Monetary Loss	6	Low (1)	6
R	Reputation	5	High (3)	15
P	User Financial Information Revealed	4	High (3)	12
S	Payments Network Unusable	3	High (3)	9
F	Foundation One Time Financial Loss	2	Medium (2)	4
L	Fines and Legal Penalties	1	Medium (2)	2
			TOTAL	48

Table 68. Severity: Identity or AML Database Hacked

5.4.4.2 (2) Brute Force ID Address Lookup

Property	Description
(1) Threat Scenario	An attacker enumerates all valid Estonian ID codes and queries Identity server for addresses.
(2) Actor	An attacker with knowledge of server exploits and Estonian ID code structure.
(3) Means	Identity server has no restrictions on address lookup. Attacker enumerates possible Estonian ID codes and queries the Identity server for Ethereum addresses. For given ID codes, owner information can be queried from SK LDAP directory. The number of possible Estonian ID codes is approximately 144 million based on the following structure [49]: ■ gender/century of the birth digit (one digit for two attributes) ■ date of birth digits (YY+MM+DD) ■ three random digits ■ one checksum digit
(4) Motive	The ID to address mapping allows an attacker to link all transactions on the Ethereum blockchain to real identities and sell this information to interested parties or publish the mapping online as an act of cyberterrorism.
(5) Outcome	Disclosure of user identities and their mapping to Ethereum blockchain addresses.
(6) Security Requirements	Confidentiality of the identity of user transactions is breached.
(7) Probability	High
(8) Consequences	■ Identity of all users' transactions is compromised and published for attacker gain. ■ The Foundation loses the trust of its users with their money. ■ The Foundation may receive fines and penalties for weak security practices.

Table 69. Risk Profile: Brute Force ID Address Lookup

Impact Area	Description	Ranking	Value	Score
M	User One Time Monetary Loss	6	Low (1)	6
R	Reputation	5	High (3)	15
P	User Financial Information Revealed	4	High (3)	12
S	Payments Network Unusable	3	High (3)	9
F	Foundation One Time Financial Loss	2	Medium (2)	4
L	Fines and Legal Penalties	1	High (3)	3
			TOTAL	49

Table 70. Severity: Brute Force ID Address Lookup

5.4.4.3 (3) Spoof ID Authorization

Property	Description
(1) Threat Scenario	An attacker exploits an insecure implementation of Estonian ID card authentication to approve addresses without a valid ID.
(2) Actor	An attacker with knowledge of exploits on Estonia ID authentication.
(3) Means	Attacker spoofs communication channel between Identity Server and DigiDocService to fake authentications.
(4) Motive	Create addresses on Euro 2.0 system anonymously that cannot be traced to a valid ID useful for criminal activity.
(5) Outcome	Modification of information of what is a valid or invalid ID.
(6) Security Requirements	Integrity of identity is breached.
(7) Probability	Low
(8) Consequences	■ Attacker is able to create an address not linked to a real ID that can be used to get away with fraud and other attacks on the system. ■ Foundation can no longer know for certainty what real world identity belongs to which addresses. ■ The Foundation may receive fines and penalties for weak AML practices.

Table 71. Risk Profile: Spoof ID Authorization

Impact Area	Description	Ranking	Value	Score
M	User One Time Monetary Loss	6	Low (1)	6
R	Reputation	5	High (3)	15
P	User Financial Information Revealed	4	Low (1)	4
S	Payments Network Unusable	3	High (3)	9
F	Foundation One Time Financial Loss	2	Low (1)	2
L	Fines and Legal Penalties	1	High (3)	3
			TOTAL	39

Table 72. Severity: Spoof ID Authorization

5.5. Risk Mitigation

In this section we use the results of the risk profiles and severity analysis in Section 5.4 to determine a risk mitigation approach for each critical information asset with the guidelines of OCTAVE Allegro *Step 8 - Select Mitigation Approach.* In Section 5.5.1 we enumerate the impact value risk score for each risk profile, categorize them by probability and risk score in a *Relative Risk Matrix*, and determine pools of risk mitigation approaches. Then for all risk profiles of each critical information asset, we select a risk mitigation approach and describe the risk mitigation strategy in sections Risk Mitigation I: User Money, Risk Mitigation II: Ethereum Admin Keys, Risk Mitigation III: User Identity.

5.5.1. Risk Categorization

For all documented risks in a security risk analysis an organization must decide on one of the following risk mitigation approaches:

- **Accept** - take no action to address the risk and accept the consequences.
- **Mitigate** - address the risk by developing and implementing the controls to counter the underlying threat and/or minimize the resulting impact.
- **Defer** - neither accept nor mitigate the risk, but perform further investigation

Risk mitigation requires a balanced approach of risk avoidance and limitation.

- **Avoidance** - implement appropriate controls to prevent threats and vulnerabilities from being exploited.
- **Limitation** - implement strategies that limit the adverse impact on the organization if a risk is realized.

Along with an element of cost, in which the cost of avoiding and limiting a risk must correspond to the value of the asset being protected and potential impact on the organization.

To aid the decision of which mitigation approach to apply for each risk, we build a *Relative Risk Matrix* of probability vs. risk score. The results of the risk analysis for critical information assets (Section 5.4) are enumerated in Table 73.

Symbol	Information Asset	Risk Profile	Probability	Score
(I:1)	User Money	Keys Stolen from User Device	High	39
(I:2)	User Money	Programming Error in the Ethereum Contract	Medium	48
(I:3)	User Money	Spoof ID to Ethereum Address Lookup	Medium	59
(I:4)	User Money	User Keys Stolen from Backup System	Low	43
(I:5)	User Money	Ethereum Scalability	Low	26
(II:1)	Admin Keys	Steal Admin Key from Online Server	Medium	48
(II:2)	Admin Keys	Tamper CryptoFiat Key Generation	Low	51
(II:3)	Admin Keys	Bribe Key Custodian	Low	50
(III:1)	User Identity	Identity or AML Database Hacked	Medium	48
(III:2)	User Identity	Brute Force ID Address Lookup	High	49
(III:3)	User Identity	Spoof ID Authorization	Low	39

Table 73. Risk Analysis Results

We arbitrarily divide total risk score into the three buckets 0 to 30 , 31 to 45, and 45 to 63 to represent a qualitatively low, medium, and high total risk score respectively. Based on risk analysis results we construct Relative Risk Matrix in Table 74.

	Risk Score		
Probability	**45 to 63**	**31 to 45**	**0 to 30**
High	(III:2)	(I:1)	
Medium	(I:2) (I:3) (II:1) (III:1)		(II:1)
Low	(II:2) (II:3)	(I:4) (III:3)	(I:5)

Table 74. Relative Risk Matrix

In which we assign the following risk mitigation decision categories (Table 75).

Categorization	Mitigation Approach	Applicable Risks
	Mitigate	(I:1), (III:2)
	Mitigate or Defer	(I:2), (I:3), (II:1), (III:1)
	Defer or Accept	(I:4), (II:2), (II:3), (III:3)
	Accept	(I:5)

Table 75. Relative Risk Matrix

We present the risk mitigation decisions for the risk profiles in Section 5.5.2, Section 5.5.3, and Section 5.5.4.

5.5.2. Risk Mitigation I: User Money

We present the risk mitigation strategy for User Money information asset defined in Section 5.3.4 with risk profiles defined in Section 5.4.2.

5.5.2.1 (I:1) Keys Stolen from User Device

Mitigation Approach: **Mitigate**

Key management is currently the toughest problem of cryptocurrency user experience. The reputation of the Euro 2.0 foundation hinges on the ability of users to securely manage their EUR2 keys. The current client wallet implementation stores keys in uses password protected local storage, which has the disadvantages of being easy to hack and easy to lose keys. Fortunately, the wallet solutions for Bitcoin keys are completely applicable to Ethereum. Eskandari et al. build a comprehensive comparison of key management techniques for Bitcoin with baselines of cash and online banking[31]. We propose two mitigation approaches to helping users secure their keys, an air-gapped wallet like Armory or hosted wallet like Coinbase.

An air-gapped key storage wallet like Armory[7] provides users a more malware resistant, offline, and recoverable key storage without a trusted third party, thus reducing the chance

of theft or loss of keys. Since Bitcoin and Ethereum use the same key generation mechanism a similar solution could be built and offered by the Euro 2.0 foundation.
Another approach is for key storage to be completely delegated to a online third party, such as the popular exchange Coinbase. This creates a user experience with the same properties as an online bank[31] suitable for users who want to avoid key management.

5.5.2.2 (I:2) Programming Error in the Ethereum Contract

Mitigation Approach: **Mitigate**
Consequences are extremely high for a bug in the Euro 2.0 contracts that could basically result in a total meltdown of the digital currency. There are two suggestions mitigate risk of a programming error in the smart contracts:

- Write tests for the contract to clearly define exactly functionality of the code using a testing framework, such as Truffle[34].
- Perform a security audit of Euro 2.0 smart contracts by industry professionals, such as Smart Contract Solutions[39].

5.5.2.3 (I:3) Spoof ID to Ethereum Address Lookup

Mitigation Approach: **Mitigate** and **Defer**
There are two clients looking up Ethereum Addresses by ID, user clients and the reserve bank gateway. Each has its own mitigation strategy.
The risk of bank gateway calls to Identity server being spoofed can be mitigated by the Identity Server and Bank Gateway communicating on a secure network with SSL or a VPN.
The risk of user client calls to Identity server for address lookup can be deferred for further research. Establishing a secure connection between clients and Identity server will be at the expense of an open API by requiring clients to have credentials to establish a TLS connection.

5.5.2.4 (I:4) User Keys Stolen from Backup System

Mitigation Approach: **Defer**
User keys stolen from the backup system is a risk that can be accepted for the time being. It only affects a single users who chooses to backup their keys. More research needs to be done on how cryptographically vulnerable is the challenge to attacks.

5.5.2.5 (I:5) Ethereum Scalability

Mitigation Approach: **Accept**

Ethereum and blockchain scalability is an active avenue of development in the community with scalability initiatives such as Sharding[3] and Proof of Stake[2]. For the near future scalability is not an immediate threat to the viability of an Ethereum based distributed application like Euro 2.0.

5.5.3. Risk Mitigation II: Ethereum Admin Keys

We present the risk mitigation strategy for User Money information asset defined in Section 5.3.5 with risk profiles defined in Section 5.4.3.

5.5.3.1 (II:1) Steal Admin Key from Online Server

Mitigation Approach: **Mitigate**
Admin keys of the Euro 2.0 digital currency system should be protected at all cost. We propose mitigation strategies depending on the key hosted on a server.
The *Reserve* key should never touch the internet. Ideally this key should be kept completely offline in an air gapped storage for manual transactions. For automated reserve processes, the node should live on a dedicated private network with only a dedicated Ethereum node to broadcast signed transactions.
The *Enforcer* and *Designator* key should never need to to touch the internet. All risk is mitigated if these keys live completely offline to sign transactions.
The *Approver* and *Delegate* keys unfortunately must live on a server touching the internet to be used in automated processes. Best practices must be followed with these keys on live servers with the following possible mitigations:

- Key should be stored password protected on the server and the password entered on application startup (though memory dumps now have extremely confidential information, the password).
- Have jobs see if approver or delegate keys have been used in Ethereum blockchain events without proper events on the servers.
- Invest in a HSM (Hardware Security Module) to store the key.

5.5.3.2 (II:2) Tamper CryptoFiat Key Generation

Mitigation Approach: **Defer**
This risk can be mitigated later by researching and designing an auditable key ceremony to generate the CryptoFiat master key (or any admin key) on new hardware to make it impossible for an attacker to tamper the key generation of any admin keys.

5.5.3.3 (II:3) Bribe Key Custodian

Mitigation Approach: **Defer**

Although bribery proposes a significant risk, it is not important until the value of the system vastly outweighs the value transacting on the system. This risk can be significantly mitigated by implementing Multi-sign M of N signatures for CryptoFiat Master, Enforcement, and Delegator contract operations. This would require multiple individuals to be bribed in order to compromise the admin functionality and increase security of distributing admin functionality to multiple individuals.

5.5.4. Risk Mitigation III: User Identity

We present the risk mitigation strategy for User Money information asset defined in Section 5.3.6 with risk profiles defined in Section 5.4.4.

5.5.4.1 (III:1) Identity or AML Database Hacked

Mitigation Approach: **Mitigate**
We propose mitigations for each database separately.
The AML database never needs to be accessible from the internet and only needs to be hosted in the law enforcement private network.
The Identity Server should have standard security procedures applied for databases. The database should not be open up to the public internet, but only accessible from a private network where the server lives. The endpoints to Identity server should be carefully audited to prevent security breaches and code injection.
Finally a secure communication channel needs to exist to transfer between Identity server and AML server to transfer ID to address mappings on a private connection, either an internal network or VPN.

5.5.4.2 (III:2) Brute Force ID Address Lookup

Mitigation Approach: **Mitigate**
Brute Force ID address lookup can be prevented by adding rate limiting to the API endpoint to a reasonable usages per source IP.

5.5.4.3 (III:3) Spoof ID Authorization

Mitigation Approach: **Accept**
Estonian ID card system has been in production for more than 15 years today and is trusted with banks, telecoms, and the government of Estonia for authentication needs[49]. A properly implemented Estonia and Mobile ID implementation should not be not something the Euro 2.0 system should worry about.

5.6. Conclusion

5.6.1. Summary

In this chapter we applied the OCTAVE Allegro security analysis methodology[23] to perform a security risk analysis for the Euro 2.0 digital currency system. Euro 2.0 foundation objectives, risk measurement criteria, and impact areas were introduced in Section 5.2. Critical information assets, containers, owners, and security requirements were presented in Section 5.3. For each critical information asset, risk profiles were defined and severity of realized risks computed in Section 5.4. In Section 5.5, risk scores were ranked and categorized and for each risk profile mitigation strategies were proposed for Euro 2.0.

5.6.2. Euro 2.0 Implementation Suggestions

The following are suggested improvements to the Euro 2.0 system based on the security analysis:

- Help users secure the keys with air-gapped wallets.
- Consider building an online interface to Euro 2.0 simulating an online bank avoiding key management altogether.
- Write tests for the Ethereum Smart Contracts.
- Establish secure communication channels between Euro 2.0 internal servers (Identity, Wallet, Reserve Bank, Ethereum Node).
- Analyze the security of the key backup challenge.
- Implement best practice security measures for admin keys (Approver and Delegate) running on web facing servers.
- Ensure admin key generation is secure, malware proof, and process audited and documented.
- Move manual usage of critical admin keys to M of N signatures to reduce risks of misuse, loss, and bribery.
- Protect databases with best practice security measures.
- Secure ID to address lookup endpoint to prevent against brute force attacks.

5.6.3. Limitations

Firstly the security analysis exhibited in this thesis was by no means comprehensive and for a hypothetical organization structure. We only analyzed three representative critical information assets in detail from the many possible information assets considered and those not yet discovered. Security is very contextual to a given organization. The results of the performed analysis yield relevance only to the assumptions of the hypothetical organization structure, which could be flawed. Hence, if the Euro 2.0 system does go live

with a government backed structure, further security risk analysis would need to be done in that specific organizational setup.
Secondly the security analysis performed was reactive. Since the Euro 2.0 is a not yet live prototype, research in security requirements engineering could yield more beneficial and actionable results for the foundation. This could be done with a similar approach outlined by Ahmed and Matulevičius[1] to explore securing business process at earlier developments stages and eliciting security requirements.
Finally, the security analysis was performed at a very high level. No major implementation details of the communication protocols and centralized servers were used in the analysis. More low level threats and vulnerabilities should be considered, such as those listed in the *Extensible Pattern-based Library and Taxonomy of Security Threats for Distributed Systems* of Uzunov et al[62].

5.6.4. Future Work

A few major topics of a decentralized distributed ledger based systems like Ethereum were overlooked by the security risk analysis of this thesis: **Scalability**, **Privacy**, and **Network Attacks**.
Scalability refers to the long term stability of the Ethereum platform when every node on the network must run all the code and store the current state of the world. Eventually blockchain storage will be larger than an average server's disk space, and hence need considerable improvements to continue with the same architecture. We accepted the risk of scalability in the Euro 2.0 security risk analysis even though it is a serious problem that needs to be further investigated before a state can sponsor a digital currency on a decentralized platform like Ethereum. Ethereum has proposed projects like Sharding[3] and Proof of Stake[2] to combat scalability. General discussion about a scalable blockchain protocol *"for nodes bounded by $O(N)$ computational power to process a transaction load and state size of $O(N^{2-\varepsilon})$"* is described in Buterin's *Notes on Scalable Blockchain Protocols*[20].
Privacy heavily influenced the decision to include user identity as a critical information asset (Section 5.3.6) because revealing an identity behind an Ethereum address would link the entire transaction history of that address to a user. Having linkable transaction histories on a public blockchain is design decision the currently proposed Euro 2.0 system can't get around. Dynamically generating Ethereum addresses for every new transaction can help, but is still be susceptible to clustering and graph analysis as outlined by Don and Shamir *Quantitative Analysis of the Full Bitcoin Transaction Graph*[56]. Zcash (formally Zerocash) provides a truly anonymous decentralized payments experience employing zk-SNARKs *zero-knowledge Succinct Non-interactive ARguments of Knowledge*[17]. New developments such as HAWK, explained by Kosba et al. *Hawk: The Blockchain*

Model of Cryptography and Privacy-Preserving Smart Contracts[42], bring similar privacy guarantees to smart contracts. Bridging the gap between Zcash and Ethereum adding privacy to Ethereum smart contracts is gaining interest and developing in the community[53][54][55]. Future research would entail incorporating cutting edge methods in smart contract privacy into the Euro 2.0 digital currency.

Network Attacks have been studied extensively in the Bitcoin community, such as in Eclipse attacks[38], *Majority is not Enough: Bitcoin Mining is Vulnerable*[33], and the rigorous approach to proving properties of the blockchain *The Bitcoin Backbone Protocol: Analysis and Applications*[35]. Similar research is lacking in the Ethereum space. As mining pools become larger and more commercialized there is more of a risk of mining attacks on the Ethereum network. Further research can be done to analyze and asses the risk of attacks on the network.

6. Summary

In this thesis we explored the feasibility of a decentralized digital currency supporting a government backed monetary system. The first four chapters set the stage for the security risk analysis that follows. Chapter 1 - Introduction introduced the motivation, research questions, and research methods of the thesis. In Chapter 2 - Bridge of Knowledge we gave an overview of the existing literature on currencies and payments, Bitcoin, Trust and Cryptocurrencies, Ethereum, Estonian ID Card, the ISSRM field, and OCTAVE Allegro risk analysis framework. In Chapter 3 - Euro 2.0 Digital Currency Prototype we presented the motivation behind the Euro 2.0 digital currency system, documented the features, meticulously walked through the decentralized Ethereum smart contract implementation of the core of the system, described the client server components at a high level, and how they work together to fulfill the features. In Chapter 4 - Euro 2.0 Change Analysis we turned our attention to change analysis and hypothesized the impact of the Euro 2.0 digital currency adoption for stakeholders including users, merchants, banks, and government.
In Chapter 5 - Euro 2.0 Security Risk Analysis we performed a security risk analysis using the OCTAVE Allegro framework. In the context of the risk analysis we defined the business objectives and organizational structure of the hypothetical Euro 2.0 Foundation as well as impact areas and risk criteria for the analysis. Next we defined the information assets and containers for the key information assets of the security risk analysis, Critical Asset I: User Money, Critical Asset II: Ethereum Admin Keys, and Critical Asset III: User Identity. Then for each critical risk asset we enumerated multiple risk profiles for various threat scenarios. Finally we categorized the risk impact results and proposed mitigation strategies.
Despite the analysis, both outcomes mentioned in Section 1.2 are inconclusive. We cannot conclude from a high level security analysis that the Euro 2.0 digital currency system is free from significant risks to function as a monetary system; the highest risk is the unknown. At the same time, we could not deem any of the risks presented in Section 5.4 as so severe as preventing the digital currency to function as designed, especially if the mitigation strategies in Section 5.5 are considered and implemented. On the other hand, we have learned that a decentralized cryptocurrency implemented with smart contracts exposes severe risks on two fronts: admin keys and revealing user identity. Due to the Ethereum smart contract implementation of Euro 2.0, the holder of the CrytpoFiat master key possesses an unprecedented amount of power over the security and functionality of a monetary system unprecedented in existing financial systems. Likewise the complete openness of the public ledger, even with pseudo-anonymous identities and keys generated for each transaction, still exposes users to major privacy revelations about their transaction history if the identity mapping is ever revealed. These two risks are considerable and must be further researched in other cryptography based digital currency proposals.

Finally, one very important open question: *does a decentralized implementation of transactions and accounts in a digital currency give a security advantage over a completely centralized client-server implementation*? Why do we need Ethereum? Does decentralization improve the integrity of a monetary system over the complicated implementation and privacy tradeoffs?

References

[1] Naved Ahmed and Raimundas Matulevičius. “Securing Business Processes Using Security Risk-oriented Patterns”. In: *Comput. Stand. Interfaces* 36.4 (June 2014), pp. 723–733. ISSN: 0920-5489. DOI: 10.1016/j.csi.2013.12.007. URL: http://dx.doi.org/10.1016/j.csi.2013.12.007.

[2] Vitalik Buterin et al. *Proof of Stake FAQ*. 2017. URL: https://github.com/ethereum/wiki/wiki/Sharding-FAQ (visited on 04/16/2017).

[3] Vitalik Buterin et al. *Sharding FAQ*. 2017. URL: https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQ (visited on 04/16/2017).

[4] Judith Aldridge and David Décary-Hétu. *Not an 'Ebay for Drugs': The Cryptomarket 'Silk Road' as a Paradigm Shifting Criminal Innovation*. WorkingPaper. May 2014. DOI: 10.2139/ssrn.2436643.

[5] Syed Taha Ali, Dylan Clarke, and Patrick McCorry. “Bitcoin: Perils of an Unregulated Global P2P Currency”. In: *Revised Selected Papers of the 23rd International Workshop on Security Protocols XXIII - Volume 9379*. New York, NY, USA: Springer-Verlag New York, Inc., 2015, pp. 283–293. ISBN: 978-3-319-26095-2. DOI: 10.1007/978-3-319-26096-9_29. URL: http://dx.doi.org/10.1007/978-3-319-26096-9_29.

[6] Luke Anderson et al. “New kids on the block: an analysis of modern blockchains”. In: *CoRR* abs/1606.06530 (2016).

[7] Inc. Armory Technologies. *Armory Secure Wallet*. 2017. URL: https://www.bitcoinarmory.com/ (visited on 04/16/2017).

[8] SK ID Solutions AS. *About SK*. 2017. URL: https://www.sk.ee/en/about/ (visited on 04/04/2017).

[9] SK ID Solutions AS. *DigiDocService*. 2017. URL: https://sk.ee/en/services/validity-confirmation-services/digidoc-service/ (visited on 04/04/2017).

[10] SK ID Solutions AS. *LDAP directory service*. 2017. URL: https://www.sk.ee/en/repository/ldap/ (visited on 04/04/2017).

[11] SK ID Solutions AS. *LDAP technical description*. 2017. URL: https://www.sk.ee/en/repository/ldap/ldap-kataloogi-kasutamine/ (visited on 04/04/2017).

[12] Nicola Atzei, Massimo Bartoletti, and Tiziana Cimoli. *A survey of attacks on Ethereum smart contracts*. Cryptology ePrint Archive, Report 2016/1007. http://eprint.iacr.org/2016/1007. 2016.

[13] Nicola Atzei, Massimo Bartoletti, and Tiziana Cimoli. “A Survey of Attacks on Ethereum Smart Contracts (SoK)”. In: *Principles of Security and Trust: 6th International Conference, POST 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings*. Ed. by Matteo Maffei and Mark Ryan. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017, pp. 164–186. ISBN: 978-3-662-54455-6. DOI: 10.1007/978-3-662-54455-6_8. URL: http://dx.doi.org/10.1007/978-3-662-54455-6_8.

[14] MONICA J. BARRATT. “SILK ROAD: EBAY FOR DRUGS”. In: *Addiction* 107.3 (2012), pp. 683–683. ISSN: 1360-0443. DOI: 10.1111/j.1360-0443.2011.03709.x. URL: http://dx.doi.org/10.1111/j.1360-0443.2011.03709.x.

[15] R. Beckhard and R. Harris. *Organizational transitions: Managing complex change.* Addison-Wesley Pub. Co., 1977.

[16] Richard Beckhard. “Strategies for large system change”. In: *Sloan management review.* 16.2 (1975), pp. 43–55.

[17] Eli Ben-Sasson et al. *Zerocash: Decentralized Anonymous Payments from Bitcoin (extended version)*. URL: http://zerocash-project.org/media/pdf/zerocash-extended-20140518.pdf (visited on 04/23/2017).

[18] Richard Gendal Brown. *Introducing R3 Corda™: A Distributed Ledger Designed for Financial Services*. 2016. URL: http://www.r3cev.com/blog/2016/4/4/introducing-r3-corda-a-distributed-ledger-designed-for-financial-services (visited on 04/24/2017).

[19] Vitalik Buterin. *A Next-Generation Smart Contract and Decentralized Application Platform*. 2014. URL: https://github.com/ethereum/wiki/wiki/White-Paper (visited on 04/21/2017).

[20] Vitalik Buterin. *Notes on Scalable Blockchain Protocols (version 0.3.2)*. 2015. URL: https://github.com/vbuterin/scalability_paper/blob/master/scalability.pdf (visited on 04/23/2017).

[21] Vitalik Buterin. *Serpent*. 2017. URL: https://github.com/ethereum/wiki/wiki/Serpent (visited on 04/21/2017).

[22] Steven H. Cady et al. “The Change Formula: Myth, Legend, or Lore?” In: *OD Practitioner* 46 (3 2014), pp. 32–39.

[23] Richard Caralli et al. *Introducing OCTAVE Allegro: Improving the Information Security Risk Assessment Process*. Tech. rep. CMU/SEI-2007-TR-012. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2007. URL: http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=8419.

[24] Michael Carney. *Max Levchin doesn't own any bitcoin, is more bullish on government-backed alternative*. Mar. 14, 2014. URL: https://pando.com/2014/03/14/max-levchin-doesnt-own-any-bitcoin-is-more-bullish-on-government-backed-alternative/ (visited on 03/25/2017).

[25] CoinMarketCap. *All Currencies*. 2017. URL: http://coinmarketcap.com/all/views/all/ (visited on 04/23/2017).

[26] Jennifer Collins. *A short history of the debit card*. Aug. 18, 2011. URL: https://www.marketplace.org/2011/08/18/business/news-brief/short-history-debit-card (visited on 04/24/2017).

[27] Kathleen D. Dannemiller and Robert W. Jacobs. "Changing the Way Organizations Change: A Revolution of Common Sense". In: *The Journal of Applied Behavioral Science* 28.4 (1992), pp. 480–498. DOI: 10.1177/0021886392284003. eprint: http://dx.doi.org/10.1177/0021886392284003. URL: http://dx.doi.org/10.1177/0021886392284003.

[28] Kevin Delmolino et al. *Step by Step Towards Creating a Safe Smart Contract: Lessons and Insights from a Cryptocurrency Lab*. Cryptology ePrint Archive, Report 2015/460. 2015. URL: %5Curl%7Bhttp://eprint.iacr.org/2015/460%7D.

[29] Éric Dubois et al. "A Systematic Approach to Define the Domain of Information System Security Risk Management". In: *Intentional Perspectives on Information Systems Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 289–306. ISBN: 978-3-642-12544-7. DOI: 10.1007/978-3-642-12544-7_16. URL: http://dx.doi.org/10.1007/978-3-642-12544-7_16.

[30] The Editors of Encyclopædia Britannica. *credit card*. May 11, 2016. URL: https://www.britannica.com/topic/credit-card (visited on 04/24/2017).

[31] Shayan Eskandari et al. "A First Look at the Usability of Bitcoin Key Management". In: *Proceedings of the NDSS Workshop on Usable Security (USEC)*. 37.5% acceptance rate. 2015. URL: http://people.inf.ethz.ch/barrerad/files/usec15-eskandari.pdf.

[32] Ethereum. *Solidity*. Version 0.4.11. 2017. URL: https://solidity.readthedocs.io/en/develop/ (visited on 04/24/2017).

[33] Ittay Eyal and Emin Gün Sirer. "Majority is not Enough: Bitcoin Mining is Vulnerable". In: *CoRR* abs/1311.0243 (2013). URL: http://arxiv.org/abs/1311.0243.

[34] A CONSENSYS FORMATION. *Testing your contracts*. 2017. URL: http://truffleframework.com/docs/getting_started/testing (visited on 04/16/2017).

[35] Juan Garay, Aggelos Kiayias, and Nikos Leonardos. *The Bitcoin Backbone Protocol: Analysis and Applications*. Cryptology ePrint Archive, Report 2014/765. http://eprint.iacr.org/2014/765. 2014.

[36] Ron Gross. *Omni Protocol Specification (formerly Mastercoin)*. 2013. URL: https://github.com/OmniLayer/spec (visited on 04/23/2017).

[37] Nermin Hajdarbegovic. *UK Treasury Issues 'Call for Information' on Digital Currencies*. Nov. 4, 2014. URL: http://www.coindesk.com/uk-treasury-issues-call-information-digital-currencies/ (visited on 03/25/2017).

[38] Ethan Heilman et al. "Eclipse Attacks on Bitcoin's Peer-to-peer Network". In: *Proceedings of the 24th USENIX Conference on Security Symposium*. SEC'15. Washington, D.C.: USENIX Association, 2015, pp. 129–144. ISBN: 978-1-931971-232. URL: http://dl.acm.org/citation.cfm?id=2831143.2831152.

[39] Smart Contracts Solutions. Inc. *Security audits*. 2017. URL: https://smartcontractsolutions.com/security-audits (visited on 04/16/2017).

[40] Kurt Jensen, Lars Michael Kristensen, and Lisa Wells. "Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems". In: *Int. J. Softw. Tools Technol. Transf.* 9.3 (May 2007), pp. 213–254. ISSN: 1433-2779. DOI: 10.1007/s10009-007-0038-x. URL: http://dx.doi.org/10.1007/s10009-007-0038-x.

[41] Irni Eliana Khairuddin et al. "Exploring Motivations for Bitcoin Technology Usage". In: *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. CHI EA '16. Santa Clara, California, USA: ACM, 2016, pp. 2872–2878. ISBN: 978-1-4503-4082-3. DOI: 10.1145/2851581.2892500. URL: http://doi.acm.org/10.1145/2851581.2892500.

[42] A. Kosba et al. “Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts”. In: *2016 IEEE Symposium on Security and Privacy (SP)*. May 2016, pp. 839–858. DOI: 10.1109/SP.2016.55.

[43] Kristo Käärmann. *Cost of making payments*. Oct. 22, 2013. URL: http://kaarmann.com/the-cost-of-making-payments-brainstorming-with-central-bankers/ (visited on 04/24/2017).

[44] Kristo Käärmann. *Government backed bitcoin*. Mar. 29, 2014. URL: http://kaarmann.com/government-backed-bitcoin/ (visited on 03/25/2017).

[45] Kristo Käärmann. *The future of money may be in the ether*. May 30, 2016. URL: http://kaarmann.com/the-future-of-money-is-in-the-ether/ (visited on 03/25/2017).

[46] Tether Limited. *Tether: Fiat currencies on the Bitcoin blockchain*. 2014. URL: https://tether.to/wp-content/uploads/2016/06/TetherWhitePaper.pdf (visited on 04/23/2014).

[47] CoinDesk LLC. *BITCOIN PRICE INDEX CHART*. 2017. URL: http://www.coindesk.com/price/ (visited on 05/15/2017).

[48] M. Mahunnah et al. “Heuristics for Designing and Evaluating Socio-technical Agent-Oriented Behaviour Models with Coloured Petri Nets”. In: *2014 IEEE 38th International Computer Software and Applications Conference Workshops*. July 2014, pp. 438–443. DOI: 10.1109/COMPSACW.2014.74.

[49] Tarvi Martens. “Electronic identity management in Estonia between market and state governance”. In: *Identity in the Information Society* 3.1 (2010), pp. 213–233. ISSN: 1876-0678. DOI: 10.1007/s12394-010-0044-0. URL: http://dx.doi.org/10.1007/s12394-010-0044-0.

[50] Richard Milne. *Sweden’s Riksbank eyes digital currency*. Nov. 15, 2016. URL: https://www.ft.com/content/0e37795c-ab33-11e6-9cb3-bb8207902122 (visited on 03/25/2017).

[51] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. 2008. URL: https://bitcoin.org/bitcoin.pdf (visited on 03/12/2016).

[52] Inc. PayPal. *What are the fees for PayPal accounts?* 2017. URL: https://www.paypal.com/us/selfhelp/article/What-are-the-fees-for-PayPal-accounts-FAQ690 (visited on 04/24/2017).

[53] Christian Reitwiessner. *An Update on Integrating Zcash on Ethereum (ZoE)*. 2017. URL: https://blog.ethereum.org/2017/01/19/update-integrating-zcash-ethereum/ (visited on 04/23/2017).

[54] Christian Reitwiessner. *zkSNARKs in a nutshell*. 2017. URL: http://chriseth.github.io/notes/articles/zksnarks/zksnarks.pdf (visited on 04/23/2017).

[55] Christian Reitwiessner. *zkSNARKs in a nutshell*. 2017. URL: https://blog.ethereum.org/2016/12/05/zksnarks-in-a-nutshell/ (visited on 04/23/2017).

[56] Dorit Ron and Adi Shamir. "Quantitative Analysis of the Full Bitcoin Transaction Graph". In: *Financial Cryptography and Data Security: 17th International Conference, FC 2013, Okinawa, Japan, April 1-5, 2013, Revised Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 6–24. ISBN: 978-3-642-39884-1. DOI: 10.1007/978-3-642-39884-1_2. URL: http://dx.doi.org/10.1007/978-3-642-39884-1_2.

[57] BLOCKCHAIN LUXEMBOURG S.A. *Latest Blocks*. 2017. URL: https://blockchain.info/ (visited on 03/13/2017).

[58] Corina Sas and Irni Khairuddin. *Design For Trust: An Exploration Of The Challenges And Opportunities Of Bitcoin Users*. 2016. URL: http://eprints.lancs.ac.uk/83765/1/Design_for_trust.pdf.

[59] Emily Spaven. *Citi: UK Government Should Create Own Digital Currency*. May 20, 2015. URL: http://www.coindesk.com/citi-uk-government-should-create-digital-currency/ (visited on 03/25/2017).

[60] Drew Springall et al. "Security Analysis of the Estonian Internet Voting System". In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. CCS '14. Scottsdale, Arizona, USA: ACM, 2014, pp. 703–715. ISBN: 978-1-4503-2957-6. DOI: 10.1145/2660267.2660315. URL: http://doi.acm.org/10.1145/2660267.2660315.

[61] Don Tapscott and Alex Tapscott. *Blockchain Revolution: How the Technology Behind Bitcoin Is Changing Money, Business, and the World*. Brilliance Audio, 2016. ISBN: 1511357673, 9781511357678.

[62] Anton V. Uzunov and Eduardo B. Fernandez. "An Extensible Pattern-based Library and Taxonomy of Security Threats for Distributed Systems". In: *Comput. Stand. Interfaces* 36.4 (June 2014), pp. 734–747. ISSN: 0920-5489. DOI: 10.1016/j.csi.2013.12.008. URL: http://dx.doi.org/10.1016/j.csi.2013.12.008.

[63] Gavin Wood. *Ethereum: A Secure Decentralised Generalised Transaction Ledger*. Version EIP-150 REVISION (1e18248 - 2017-04-12). 2014. URL: https://ethereum.github.io/yellowpaper/paper.pdf (visited on 04/21/2017).

[64] Gavin Wood and Fabian Vogelsteller. *JSON RPC*. 2015. URL: https://github.com/ethereum/wiki/wiki/JSON-RPC (visited on 04/24/2017).

Appendix A - Files

A.1: Euro 2.0 Codebase

`http://github.com/cryptofiat`

A.2: CryptoFiat Contract

`https://github.com/cryptofiat/contract/blob/master/CryptoFiat.sol`

A.3: Archive of Cited Web Pages

`https://goo.gl/5jJKRc`