VILNIUS UNIVERSITY
FACULTY OF MATHEMATICS AND INFORMATICS
INSTITUTE OF COMPUTER SCIENCE
DEPARTMENT OF COMPUTATIONAL AND DATA MODELING

Scientific Research Project

Traceability of Funds in Blockchain Privacy Protocols

Lėšų susekamumas blokų grandinių privačiuose protokoluose

Done by:
Justinas Lekavičius signature

Supervisor:
lekt. Linas Būtėnas

Vilnius
2022

Contents

Abstract

This scientific research project is the creation of a model for fund traceability in private blockchain protocols. Nowadays, cryptocurrencies are widely used for a variety of payments due to their accessibility, decentralization, and security, while security aspects include anonymity and untraceability, depending on the protocols and cryptocurrencies used. However, funds may be stolen or this form of payment may also be used for illegal purposes, such as money laundering and ransomware payments, and as more anonymity-ensuring blockchain platforms are created, more threats arise. Therefore, the goal of this scientific research project is to create a model that ensures traceability of funds in blockchain privacy protocols by evaluating their risk factor without compromising the anonymity of users. Research was done on how entity evaluations can be taken into account when marking actions as risky on not risky for transactions in a privacy platform. The result is a theoretical model that, based on data from third-party oracles, can help evaluate the risk score of a transaction on a privacy protocol, estimating the risk factor of funds and preventing money laundering and other illegal activities.

Santrauka

Lėšų susekamumas blokų grandinių privačiuose protokoluose

Šis mokslo tiriamojo darbo projektas yra sukurtas modelis, skirtas lėšų susekamumui privačiuose blokų grandinių protokoluose. Šiais laikais kriptovaliutos plačiai naudojamos įvairiems mokėjimams dėl jų prieinamumo, decentralizacijos ir saugumo, o saugumą užtikrina anonimiškumas bei nesusekamumas, priklausomai nuo naudojimų protokolų bei kriptovaliutų. Deja, lėšos gali būti pavogtos arba mokėjimai gali būti naudojami neteisėtiems tikslams, pavyzdžiui, pinigų plovimui ar išpirkoms. Todėl šio mokslo tiriamojo darbo projekto tikslas – sukurti modelį, užtikrinantį lėšų susekamumą privačiuose blokų grandinių protokoluose, įvertinant lėšų rizikos faktorių ir nepažeidžiant naudotojų anonimiškumo. Ištirta kaip asmenų rizikingumo įvertinimai gali būti panaudojami pažymint atliekamus veiksmus rizikingais arba ne atliekant transakcijas privačioje platformoje. Darbo rezultatas yra teorinis modelis, kuris, remiantis trečiųjų šalių orakulų duomenimis, padeda įvertinti transakcijos privačioje platformoje rizikos balą bei numatyti lėšų rizikos faktorių, užkertant kelią pinigų plovimui ir kitoms nelegalioms veikloms.

Introduction

Nowadays there are more ways for people to send and receive funds for goods and services than ever before, as new various platforms and protocols emerge that provide transparency, safety, and decentralization when it comes to making online payments. Cryptocurrencies, such as Ethereum and Bitcoin, are becoming increasingly more popular as decentralized applications are processing crypto funds worth billions of dollars daily, and the total cryptocurrency market cap was reported to be $25 billion in March 2017 [1]. Nevertheless, even though cryptocurrency payments themselves provide anonymity, the payments are still recorded on a public ledger and can be traced to the payee's address. To solve this problem, some platforms provide an ever-higher level of privacy, such as Tornado Cash which aim to break the link between sender and receiver addresses for depositing and withdrawing funds by utilizing mixer contracts [2]. These mixer contracts allow users to deposit their funds into a "mixer" (or a "tumbler"), for example with a locally-stored secret key, and then withdraw funds into a new wallet using that same secret key for identification, effectively removing traces that may lead back to the original sender wallet.

However, as more users enter the market, more risks arise in terms of the potential illegal uses of cryptocurrencies. A prominent and relevant to this scientific research project example would be the Ronin Network's hack when an exploit was used to steal more than 21000 Ether cryptocurrency (worth more than $25 million dollars at the time of writing), and the stolen funds were moved to the Tornado Cash privacy exchange platform [3], allowing the hackers to withdraw the stolen funds after some time, essentially committing the crime of money laundering. Such incidents not only expose the vulnerabilities of the attacked systems but also lessen trust in the blockchain privacy protocols that are used to store the stolen funds and scramble them via mixers for withdrawal to newly created wallets. To combat the overall potential illegal use of cryptocurrencies, tools are developed for risk management of the crypto market participants, such as wallet screening services that scan an address and analyze links to other entities which may be officially sanctioned or simply participants in platforms that have been deemed either of questionable legality or simply highly likely used for illegal actions. However, such tools are most suitable for either banking systems or platforms that lean more towards transparency instead of user privacy and unlinkability. Therefore, it becomes clear that users with their funds cannot safely participate in such privacy platforms until there are better controls in place. Motivated by a lack of specific solutions for ensuring the safety of such systems, and having an interest in cryptocurrencies myself I decided to create a solution that can potentially reduce the risk for existing users in such blockchain privacy platforms and also onboard new participants that require more control, as well as filter out bad actors.

The main goal of this scientific research project is to create a model that ensures traceability of funds by inspecting the risk factor of entities and marking the fund transactions as either risky or not in a privacy platform without removing the anonymity aspect of the users. The implementation of the model can serve as a deterrent against illegal activities such as money laundering or making payments using stolen funds, as the marked "risky" payments can either be rejected outright or allowed at the cost of the entity's self-exposure as a criminal.

The tasks of this scientific research project are as follows:

- Analysis of related work and projects relevant to the scientific research project
- Design of the model (theoretical part of the project)
- Implementation of the model (experimental/practical part of the project)

The result of the scientific research project is a created theoretical model that utilizes data from third-party oracles to evaluate the potential risks of transactions on privacy protocols. The estimation of the risk factor of transactions can be expressed by a risk score that indicates the likelihood of the illegal nature of the funds, effectively either disallowing the withdrawing or depositing of such funds or simply identifying the sender or receiver of funds as "risky" or "not risky", eliminating the possibility of money laundering or transferring of stolen funds to a new "clean" wallet in an attempt to hide their true nature.

1 Related Work Analysis

This section contains the analysis of other works related to the scientific research project. A review of the state of the art (including the blockchain protocols, wallet checking/screening solutions, and methodologies) is beneficial when creating a solution for solving the problems of controlling the actions of potentially suspicious entities, either by taking preemptive measures or filtering out such entities altogether.

1.1 Cryptocurrency Variants and Modifications

Nowadays, there are several solutions for those that aim to achieve as much anonymity as possible, one of them being using certain cryptocurrencies that have a different design and functionalities when compared to other public, more popular digital currencies such as Ethereum, Bitcoin, and others. Some are more privacy-oriented, and some may have controls in place to prevent them from being traded by sanctioned entities.

1.1.1 Privacy-oriented Coins

There are cryptocurrency coins that are claimed to be untraceable, more secure, and more private, such as Zcash and Monero, and are used more commonly when compared to other public and popular ones such as Ethereum or Bitcoin, as the former are preferred for their privacy-enhancing algorithms [4]. These digital tokens are some of the more prominent when it comes to privacy-protecting cryptocurrencies, however, the implementations of such coins may not guarantee absolute privacy and legitimate use. For example, the Monero blockchain utilizes cryptographic methods to sign transactions and hides the transaction output (and the amount of the funds transferred) [5]. Nevertheless, while the results of the paper by Kumar, Amrit, et al may prove the untraceability guarantee of the Monero blockchain to be arguable, the point remains that the untraceable transactions normally can not only be legitimate and simply anonymous but also potentially malicious or even illegal.

1.1.2 Cryptocurrency Coin Modifications

When developing a new cryptocurrency coin, i.e., writing the token contract with a programming language such as Solidity, there may be several solutions to prevent unwanted users from performing transactions using the aforementioned coin. One of the more simpler solutions would be to implement a control mechanism in the privacy platform itself, i.e. implement a platform contract check. This would require the modification of the privacy platform's smart contract (code written using Solidity programming language) to integrate whitelist/blacklist functionality. In simpler terms, any smart contract on the platform that interacts with the monitored coin would need to have a function that checks whether the user is allowed to perform actions or not, depending on their presence either in a whitelist or a blacklist.

Another, an arguably more efficient solution would be to simply modify the cryptocurrency coin itself, i.e., edit the token smart contract. In this case, the entity whitelist/blacklist check would be performed only in the token contract and all of the contracts that perform ERC standard transactions (e.g. *transfer*, *burn*, *mint* or other). This way the transaction senders will always have their credibility checked in the root token contract. It is more efficient as the function is written only once, for the cryptocurrency token itself. The transfer function (i.e. sending a payment) is overridden with the added modifier. It reverts the contract in case *require* statement is not passed. The whitelist (or blacklist) can be stored in the separate *mapping* or requested from a database or a blockchain service. A potential implementation of such function in Solidity programming language was written and an example is provided in the code block in appendix A.

However, this solution can only be easily applied for cryptocurrencies that are newly created and deployed, as already existing cryptocurrencies such as Bitcoin or Ethereum cannot be modified in such a way. Furthermore, issues may arise with maintaining the whitelist or blacklist, such as the list update rate and determining which entities are added to (or removed from) the lists. For a large platform, the list of blacklisted or whitelisted entities would have to be updated frequently to avoid as many potential illegal uses as possible.

1.2 Malicious Transaction Detection

When it comes to detecting possibly illegal cryptocurrency transactions, there are some proposed solutions. One proposal by Shengyun Xu is applying machine learning to predict possible ransomware payments. This can be achieved by using a data set of both legitimate transactions and possible ransomware transactions of various types to create a model that would be able to recognize and reject payments that are potentially malicious [6]. However, training such a model with high accuracy would require a large amount of transaction data from services such as Etherscan or Blockchain.com Explorer, which allow viewing the transaction history which can then be exported. Furthermore, the transactions would then need to be classified as either malicious or safe, for example using a scalable data-driven Bitcoin transaction analytics framework as done by Akcora, Cuneyt Gurcan, et al. [7]. Such a solution would be useful for public blockchains, as the potential ransomware payments could be flagged and the associated addresses either sanctioned or added to a blacklist.

1.3 Wallet Checking Solutions

As more and more entities participate in the cryptocurrency market, a fair assumption can be made that not all participants are law-abiding and the anonymity of cryptocurrency payments is practical for criminal activities such as extortion or ransomware payments, as law enforcement seldom effectively solves such cases [8]. When such payments are identified and associated with certain wallet addresses, however, or even if it is discovered that the wallet address is associated with known criminal entities, they may be added to official sanctions lists, an example US Department of the Treasury's Office of Foreign Assets Control (OFAC) Specially Designated Nationals (SDN) list.

One of the wallet-checking solutions is the Chainalysis API. It is a solution for inputting cryptocurrency wallet addresses and checking whether they are sanctioned or not. The evaluation is based on whether the wallet address is linked to a sanctioned entity, as detailed in the United States Department of the Treasury OFAC (Office of Foreign Assets Control) Specially Designated Nationals (SDN) sanctions lists [9]. The API can be used to get 5000 requests every 5 minutes, simply by passing a crypto wallet address (for example, starting with 0x...) as an argument and, if the address appears in the OFAC SDN list, the API returns a "Sanctioned" object, along with the text feature which describes the details of the sanction (possible reason, linked entities, etc.). If the wallet is not found on the list, however, the API returns an empty object, indicating zero results. This solution is a straightforward way to check if a crypto address is officially sanctioned, and may be used to block these sanctioned wallets from interacting in private protocols altogether. However, detailed information such as links to other addresses or payments is not provided, thus more sophisticated services would need to be used for a large and popular privacy platform. It is worth noting that also an oracle (on-chain smart contract) exists that performs checks against sanctioned addresses on EVM-compatible networks. Thus, the oracle is only available on networks such as Ethereum, BNB Smart Chain, Polygon, etc.

Another wallet checking solution is the Elliptic product suite, one of the products being Elliptic Lens, which can be used for blockchain analytics and wallet screening and monitoring, either via dedicated API or a user interface [10]. Such a solution can be used for financial crime risk management and crypto wallet screening in real-time, with the output being a risk score that depends on the wallet address's association with risky entities or risky transactions. The result is that risky transactions can be blocked before they are even sent, enforcing the security of fund exchanges and other platforms. Such a solution can be also beneficial to ensuring fund traceability on private protocols via wallet screening, and for example, if the wallet does not reach a predetermined score, e.g., an 8 out of 10 (has too many suspicious sources or destinations of funds), the address transactions could be either accepted or rejected.

1.3.1 Wallet Checking Solution Possible Use Cases

Taking into consideration the related works and their provided solutions, some possible use cases were modeled to visualize the possibilities of implementing the different solutions into a privacy platform and discover possible advantages and disadvantages. Furthermore, these solutions are more theoretical and their practical implementation may be various, depending on the blockchain protocol and cryptocurrency used in the protocol. One of the possible use cases when Elliptic Lens is integrated into a privacy platform is displayed in figure 1.

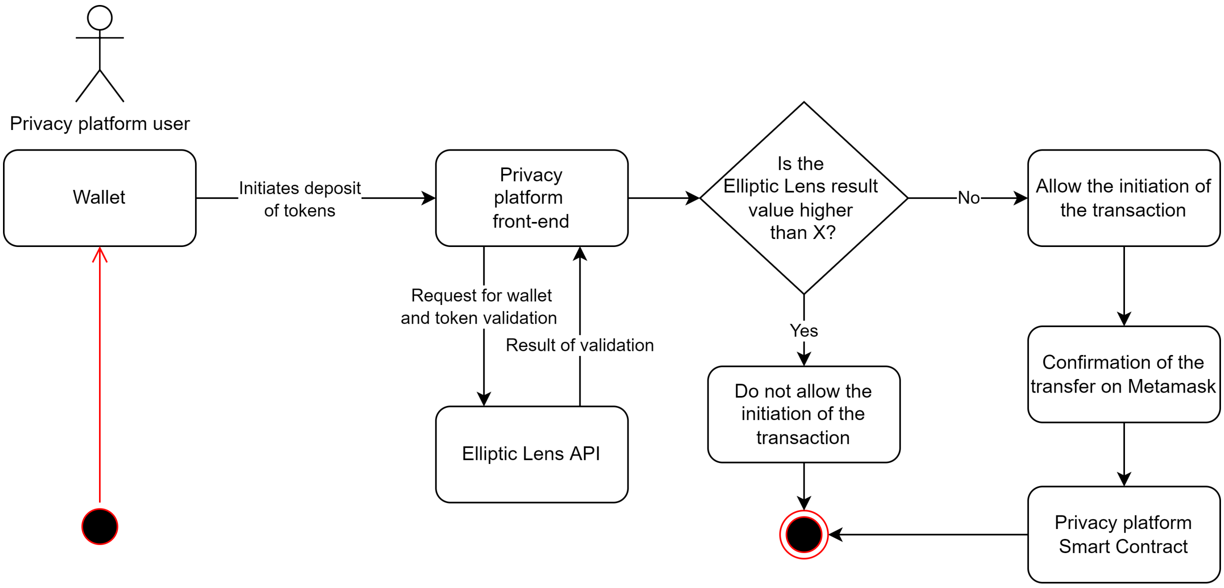

Privacy platform user
Wallet
Initiates deposit of tokens
Privacy platform front-end
Is the Elliptic Lens result value higher than X?
No
Allow the initiation of the transaction
Request for wallet and token validation
Result of validation
Yes
Confirmation of the transfer on Metamask
Do not allow the initiation of the transaction
Elliptic Lens API
Privacy platform Smart Contract

Figure 1. Possible use case of using service similar to Elliptic Lens in a privacy protocol.

A feasible scenario of using a wallet screening service with a privacy platform would involve the service's integration with the front-end part of the privacy platform, with the wallet screening being initiated during the action of crypto coin deposit into the system. Once the validation result is received, the decision can then be made whether to allow or disallow the initiation of the transaction, depending on the returned result. For example, if the returned risk score is higher than 3, the initiation of the transaction is now allowed. However, if the risk score is lower than 3, the transaction is allowed and then can be confirmed via Metamask for interaction with the privacy platform smart contract. This approach would work for absolute filtering of bad actors, disallowing them from effectively using the system, however, the entry criteria (i.e. the minimum allowed risk score) can be considered subjective and not unbiased towards the system users. Also, dependency on a single oracle for providing the wallet screening functionality may not be viable if the aforementioned service stops functioning. Thus, improvements would need to be made to such a model.

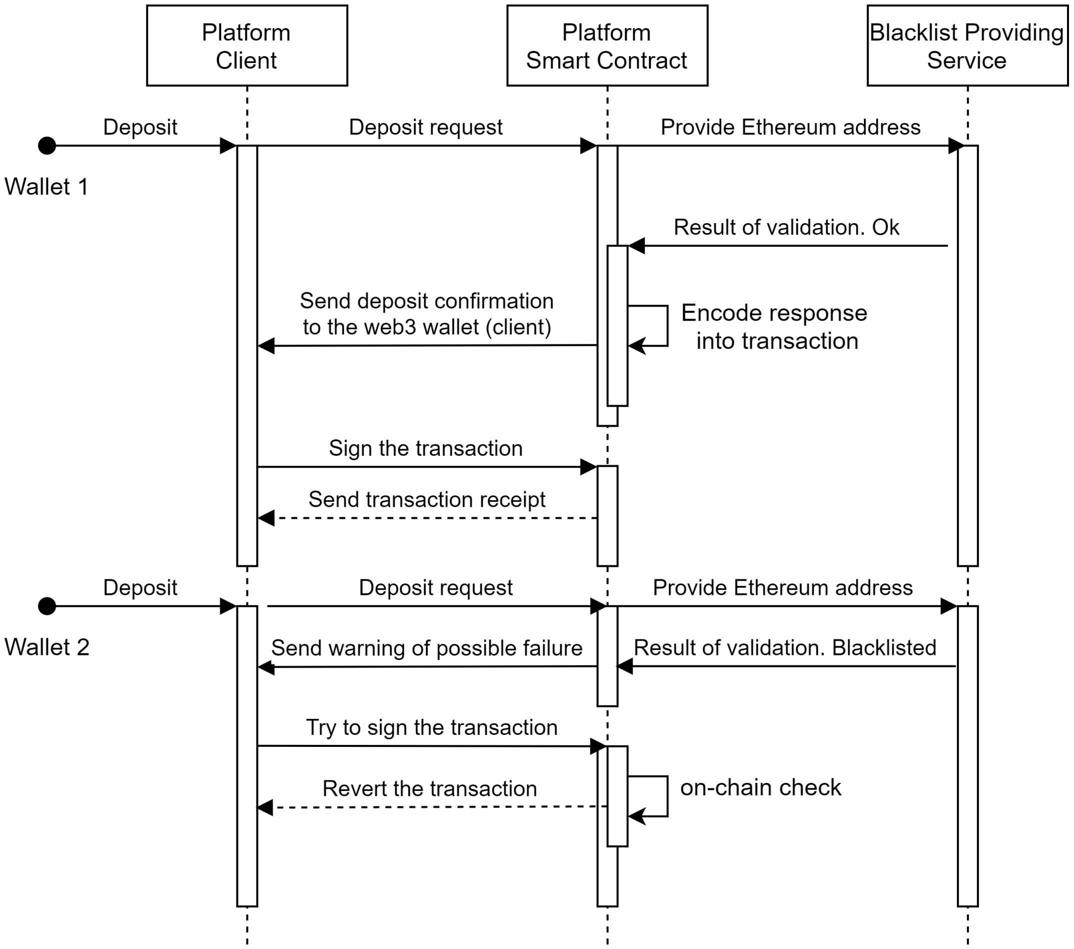


Figure 2. Proposed theoretical model of the blacklisted wallet prevention on a blockchain privacy protocol.

Another possible use case, when a service similar to Chainalysis oracle is used with the privacy protocol is displayed in figure 2.

In this case, there are several components of the potential blacklisted wallet prevention model:

- Platform client side (UI and web3 wallet)
- Platform back-end side (Smart Contract on Ethereum Virtual Machine)
- Blacklist providing service (in this case the data provided by Chainalysis oracle).

In this model *deposit* is taken as an exemplary ERC-based standard transaction that uses *transfer* function. In the first scenario, Wallet 1 tries to interact with the platform's smart contract deposit functionality. Firstly, a deposit request is sent to the contract, which forwards this request to the decentralized blacklist on-chain service (in this case, Chainalysis oracle which checks data on the OFAC's Specially Designated Nationals (SDN) list). In case of success (address is not sanctioned) - the response is encoded into the transaction and forwarded back to the client to sign via web3 wallet, and only then the deposit request is initiated. In the second scenario, the wallet has suspicious transactions recorded or is simply blacklisted in the decentralized blacklist service-provided data. After the deposit request is forwarded and confirmation of the suspicious wallet is sent - the user is warned about the possible failure of the transaction. In case a suspicious wallet wants to avoid client-side interaction via the platform front-end and wants to proceed to perform a transaction on the blockchain - an additional on-chain check is being performed. The transaction hash is generated, however - it will revert to the contract state. The advantage of such a model is that the implementation would be fairly straightforward without the need of calculating any risk scores, however, at the same time, it is also the disadvantage of the model, i.e., its unsatisfactory simplicity for more sophisticated privacy platforms. Furthermore, the oracle would have to be constantly up-to-date to guarantee the highest possible accuracy, timeliness, and validity of the provided information.

1.4 Conclusions from Analysis of Related Works

Several different solutions for the problem of fund traceability on private blockchain protocols have been analyzed, and while the various solutions have their advantages and disadvantages, as well as varying levels of implementation difficulty, such factors have been evaluated in an attempt to create a model that can assure the higher level of security in the private platform. The decision was made to create a model that would assure legitimate users of the system's privacy, as well as the funds' legality, and at the same time deter fraudulent users that may attempt to use the platform for moving funds that have been obtained illegally, as well as money laundering. The implementation of the model would allow the users to deposit funds and withdraw them at will, however, if an entity deemed risky by wallet screening oracles would make a deposit, the funds would be marked as risky as well, enforcing their traceability potential and making their withdrawal to a new wallet ineffective in terms of money laundering as the funds, as well as the new wallet would be marked as risky as well.

2 Theoretical Model of Fund Traceability Methodology

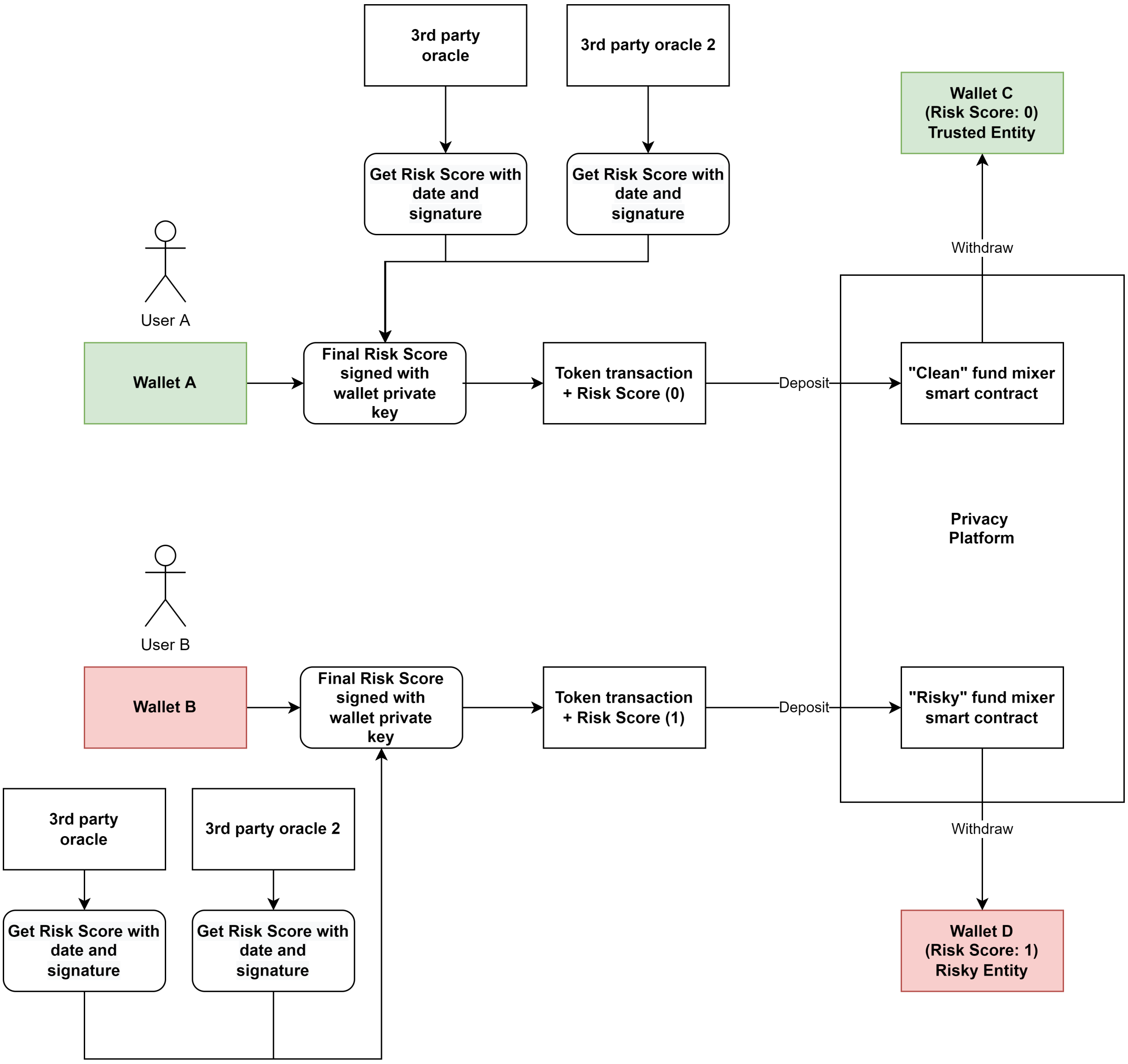

3rd party oracle
3rd party oracle 2
Get Risk Score with date and signature
Get Risk Score with date and signature
Wallet C (Risk Score: 0) Trusted Entity
Withdraw
User A
Wallet A
Final Risk Score signed with wallet private key
Token transaction + Risk Score (0)
Deposit
"Clean" fund mixer smart contract
Privacy Platform
User B
Wallet B
Final Risk Score signed with wallet private key
Token transaction + Risk Score (1)
Deposit
"Risky" fund mixer smart contract
Withdraw
3rd party oracle
3rd party oracle 2
Get Risk Score with date and signature
Get Risk Score with date and signature
Wallet D (Risk Score: 1) Risky Entity

Figure 3. Theoretical model of the methodology for fund traceability on a blockchain privacy protocol

The theoretical model of the methodology for traceability of funds in blockchain privacy protocols is displayed in figure 3. The model in question describes two cases when two different users (Wallet A and Wallet B) attempt to interact with the platform while having either a final risk score of 0 – false (safe user) or 1 – true (risky user). Depending on the evaluation of the entity's risk factor, their tokens will be deposited into a "clean" or "risky" fund mixer smart contract respectively. The "clean" fund mixer smart contract may contain funds that are deemed trustworthy, while the "risky" fund mixer smart contract would contain tokens that are suspected to be associated with criminal activities.

The created methodology for fund traceability assumes that the risk scores for different users are provided by several, at least two third-party oracle services. The risk scores are also assumed to be in range of 0 to 10, the lowest meaning zero risk, and 10 meaning the highest indicator of risk. It is also assumed that these hypothetical third-party oracles would output the relevant user's risk score along with the date of the screening, as well as the signature of the service, for the legitimacy of the wallet screening results and proof that the result was not fabricated by either the privacy platform or the user. The risk scores provided by the different oracle services are then presumably combined into a final risk score (for example, the mean of all the risk scores) and signed with the user wallet's private key for identification that the risk score belongs to the user in question and not somebody else.

Depending on the implementations in the privacy platform, the risk score is then converted into either a 0 or 1, false and true respectively. For example, if the risk score threshold is set to 5 in the privacy protocol configuration, any user with a final risk score of 5 and higher would receive the converted risk score of 1 – true (risky), and users with a lower risk score than 5 would be designated a risk score of 0 – false (not risky). Another way to associate the risk score with the relevant entity would be to sign the risk score with a private key generated by the privacy platform – this would be a case of off-chain identification.

While the security aspect of signing the risk score with the user wallet's private key may be debatable, the cryptography solution in this case is just a "first step" proposal and may be improved upon in the future. The assumption is also made that when a deposit is made, a secret key is generated for the user so that they could withdraw the exact amount of funds they have deposited during the transaction. Because the implementations of deposit and withdraw in privacy platform systems may differ, this detail was not included into the whole model diagram.

The main idea is that even though the funds are scrambled by the mixer smart contract functionality and the exact origin, such as the original sender to either Wallet A or Wallet B are unknown after withdrawal, the funds are still marked as either "clean" or "risky" as they have been withdrawn from one of the respective contracts. For example, if the tokens deposited by Wallet B into the "risky" fund mixer smart contract have been withdrawn into a Wallet D, which is a newly created wallet with a clean slate, i.e., no transaction history that can be viewed via a blockchain explorer such as Etherscan, the Wallet D will be marked with a risk score of 1 as the funds were transferred from the "risky" fund mixer smart contract, which contains presumably illegitimately obtained tokens. This would not only enforce AML (anti-money laundering) but also mark the newly created wallet as risky as well (in case of risky funds were withdrawn).

One advantage of the created model is that the anonymity of privacy protocol users is still intact, i.e., the mixer smart contracts do not expose their identity once tokens are withdrawn to a new wallet, however, the wallets are provided a risk score which then can be used for further judgment of the entity. For example, if the user (Wallet A) has valid intentions and uses legally obtained cryptocurrency for private deposit and withdrawing transactions, their new Wallet C will be associated with the "clean" fund mixer smart contract. However, if Wallet B attempts to do the same, although with funds linked to criminal activities, money laundering attempts by withdrawing the funds to a fresh new Wallet D will be void and the wallet will be exposed as having withdrawn tokens from the smart contract that only contains risky funds. The association between the Wallet D and the risky fund smart contract could be viewed in a blockchain explorer, by checking the address's transaction history.

Another advantage of the model and its potential implementation is the increased trust in the privacy protocol. Because the funds are split into two "pools", the user would not have to worry about receiving potentially illegal funds from the mixer, assuming they have deposited funds with a risk score of 0 – false (non-risky). Furthermore, in case a user with a risk score of 1 - true (risky) deposited tokens into the mixer, and it turned out that the risk score was in some way unfair and uninformative (for example, calculated by the third-party oracles inaccurately), the funds would not be confiscated and they could be withdrawn at any time. Nevertheless, if the risk score of 1 was legitimate for an entity with criminal intent, the "risky" fund mixer smart contract would serve as a deterrent, to drive bad actors away from the platform, making it unsuitable for criminal activities such as money laundering.

To test the model's efficiency in a real privacy platform, an experiment is made with mock data of risk scores, as real, actual third-party oracles are not used due to their difficulty to utilize for the project (because of difficulties such as cost, purchase options, etc.). The experiment is used to create a proof-of-concept of the model, gaining valuable insights and revealing potential ways of improving the model and discovering its possible weaknesses or disadvantages.

3 Fund Traceability Model Practical Implementation

This section covers the practical implementation of the created model for the scientific research project, i.e., the experiment part. Used technologies, relevant diagrams, and insights, as well as findings, are detailed below.

3.1 Used Technologies and Solutions

For the experiment, several technologies and solutions were used to create a proof-of-concept practical implementation of the model. The following are used for the project:

- Remix – smart contract development environment, used for interacting with, as well as deploying smart contracts written in Solidity programming language.
- Metamask – web3 wallet provider for creating wallets and sending, as well as confirming transactions, i.e., interacting with smart contracts.
- Solidity programming language – object-oriented high-level programming language, used for the smart contract programming. The smart contract files end in .sol.
- Truffle – development environment, used for compiling, testing and deploying the smart contracts for blockchains using the Ethereum Virtual Machine. Compiler version 0.7.6 was used.
- Tornado Cash – open source privacy protocol for Ethereum, used as an example platform for the model practical implementation – depositing and withdrawing funds with some example entity wallet addresses and their respective risk scores.

3.2 Risk Checking in Privacy Protocol Smart Contract Schema

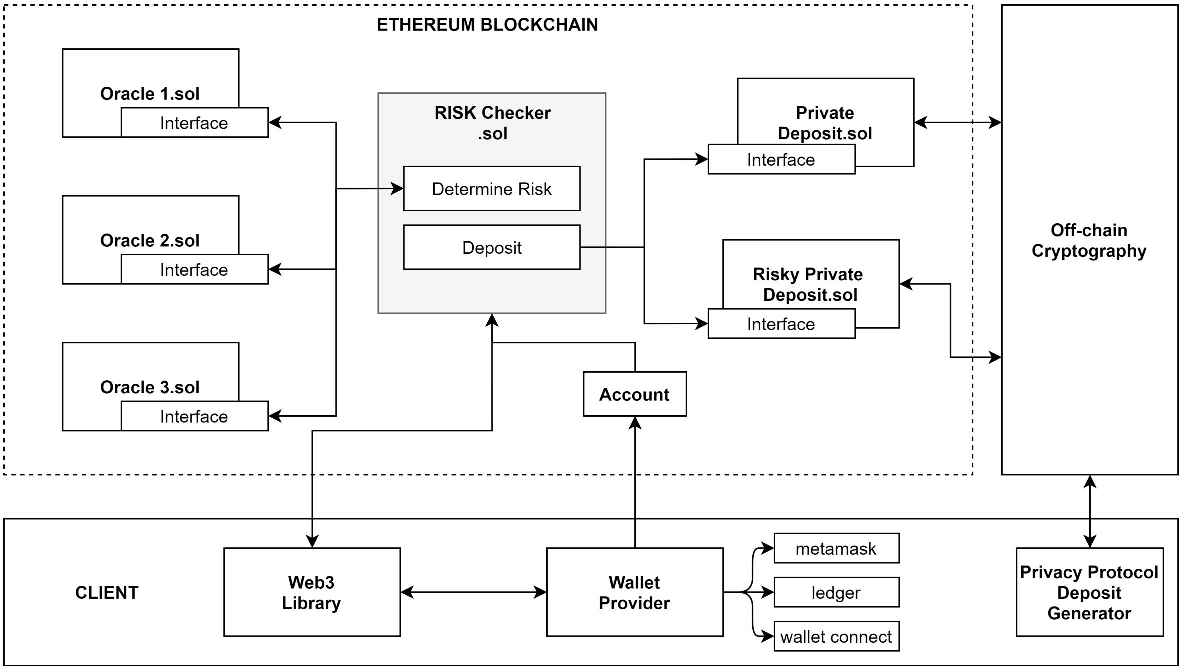


Figure 4. Schema of the smart contracts for risk checking integration in privacy protocol.

The proposed risk checking smart contract schema for fund traceability in a privacy protocol is presented in figure 4. The platform in this case consists of several components: Ethereum blockchain, Solidity-based smart contracts, client-side libraries (web3, wallet providers such as Metamask, Ledger, or WalletConnect), and off-chain cryptography libraries that are used in already existing privacy protocols. To successfully trace unreliable funds, users would have to access privacy protocols through the risk checker smart contract, which is the main model contract. The risk score would be calculated by communicating through existing oracle contracts either using their respective API requests (off-chain solution), or smart contract interfaces (on-chain solution).

3.3 Implementation in a Blockchain Privacy Protocol

The experiment is done by attempting to create a practical implementation of the model in a blockchain privacy protocol, in this case utilizing the Tornado Cash platform. This section also covers the experiment preparation (data selection) and the procedures done for the experiment: smart contract creation and functionality testing with the privacy protocol.

3.3.1 Experiment Preparation

For the testing of the model, some wallets with synthetic data were generated. Each wallet has an example risk score attached (in a real-world scenario different decentralized providers (third party oracles) would calculate the score according to their respective formulas). A risk check contract would take the average value and determine whether the wallet might have a higher probability of having illegal or hacked funds.

$$risk = \begin{cases} 1, & \text{if } \frac{1}{n}\sum_{i=1}^{n} r_i > 0.5, \quad r \in [0;1] \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

For example, if the mean is higher than 0.5 - the wallet is considered potentially unreliable - therefore marking all funds incoming from that wallet as risky (TRUE). The risk score threshold may be chosen at will, and the 0.5 threshold was chosen simply for the experimentation purposes. The risk category has only two values to preserve privacy for users that do not have a higher probability of illegal funds (it is much easier to identify a wallet with a 0.12 risk score during withdrawal instead of a risk score of 0 since the vast majority is going to have the same value). However, if the funds are marked as not trustworthy - the privacy would be reduced, as the set of such wallets is going to be much smaller. In addition to having funds marked, the possibility to link the withdrawing wallet with the depositing wallet is going to be much higher, as it is likely that there will be fewer transactions done to the "risky" fund mixer smart contract.

The test data is provided in table 1. For the experiment, eight Ethereum wallets were created and mock risk scores were added for them: r1, r2, and r3, each score from different respective third-party risk score information providing oracle. There is one wallet, which is identified as a "higher risk" wallet.

Ethereum wallet address	**r1**	**r2**	**r3**	**risk**
0x2d07F12807dC2e27C908BfeEcceCfE830103436a	0,45	0,4	0,45	FALSE
0x0C2cC7831fAd76B23CA437e95b5c323479e65EbE	0,2	0,12	0,2	FALSE
0xDa61e426A62c5506b531f6446EA731Bd2eA07E81	0,45	0,5	0,45	FALSE
0x3c4f0C330eF34f029aB61DDC2Afe7a51E7031D7b	0,12	0,01	0	FALSE
0x7a1abdE36E2853B0d1E3C1bbc663a178B6607e2c	0,65	0,5	0,5	TRUE
0x837699726524153165l167Bc7E46d445C62BD11F	0,2	0	0	FALSE
0x53c00342a9C4Bd63F0353F6CaBD6ec71Fb0aB058	0	0	0	FALSE
0xB5fdb2Cfd0bDd222c853E4f94d796d91e8EFEDB7	0,1	0	0,1	FALSE

Table 1. Synthetic data for model testing - wallets with assigned risk scores.

3.3.2 Smart Contract Creation and Deployment

For the practical implementation, smart contracts were written using Solidity programming language, and deployed on the Goerli Ethereum test network. Two contracts for both "risky" fund and regular fund mixers respectively were created and deployed. The compiler version for all of the contracts was "0.7.6+commit.7338295f.Emscripten.clang", though contracts could be deployed for all 0.7 versions. Migration to the test network was done using the truffle framework. Open source Tornado Cash contracts that were deployed for testing:

- Hasher.sol
- Mixer.sol

Risk checker custom protocol deployed contracts:

- Custom token contract (ERC20.sol);
- Privacy protocol contract (IPrivacyProtocol interface, RiskHandler.sol);

3.3.3 Testing on the Blockchain Privacy Protocol

The written smart contracts and their functionality were tested with the Tornado Cash privacy protocol as a proof-of-concept, however, it was deployed for the custom token in testnet. Mixer contracts were integrated into the main risk checker contract using interfaces and mappings:

```
        //Risky Funds
        privacyProtocols[true] = IPrivacyProtocol(0
            xAC8399668eCbF10D88933bd3762089B607325724);

        //Regular Funds
        privacyProtocols[false] = IPrivacyProtocol(0
            x47dB1d79899aed481De507AD685939ff719df516);
```

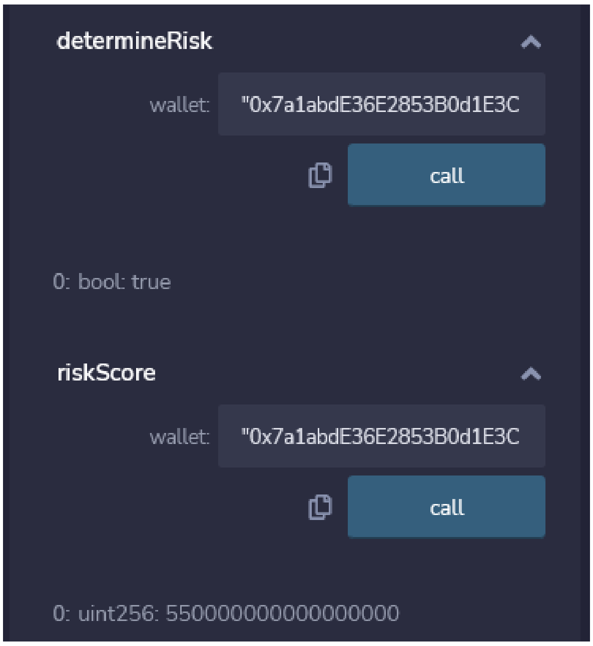
Main risk checker contract also acts as a router to the privacy protocol contracts. For example, if wallet is determined as "high-risk", it is routed to the specific mixer during deposit.

```
bool isRisky = determineRisk(msg.sender);
privacyProtocols[isRisky].deposit(commitment);
```

Tornado Cash open source project consists of several parts: off-chain cryptography (which creates public and private values for deposit and withdrawal) and smart contracts (which store public keys and perform token deposits on the blockchain). This protocol enables the privacy of the Ethereum wallets since the link between deposits and withdrawals is broken. The risk, model, however, communicates with such protocol by also marking incoming funds. Interfaces enable such communication and testing on the privacy protocol.

Users can not determine the difference between interacting with a mixer or risk-checking protocol. During deposit, the account provides public values whereas during withdrawal private proof is generated off-chain and is also being provided to the risk checking contract. All of the verification is being done in open-source Tornado contracts, however, all of the token transfers are controlled by a risk checker.

3.4 Experiment results



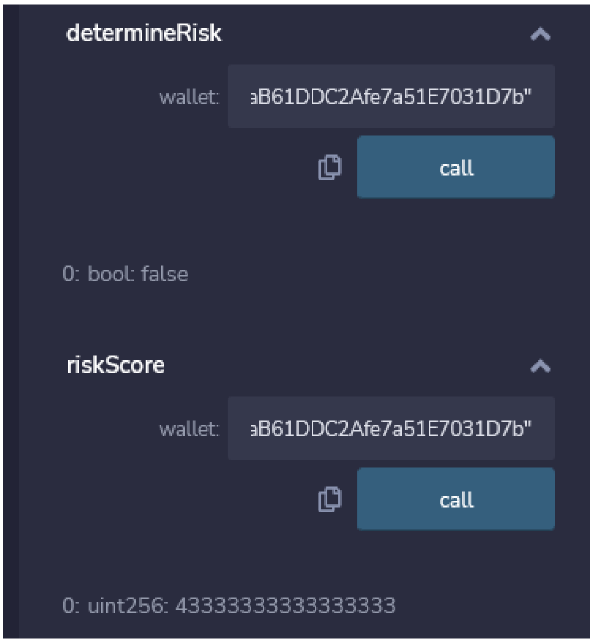
wallet: "0x7a1abdE36E2853B0d1E3C

0: bool: true

wallet: "0x7a1abdE36E2853B0d1E3C

0: uint256: 550000000000000000

Figure 5. Result for address with risk score more than 0.5 (risky)



wallet: B61DDC2Afe7a51E7031D7b"

0: bool: false

wallet: B61DDC2Afe7a51E7031D7b"

0: uint256: 43333333333333333

Figure 6. Result for address with risk score less than 0.5 (not risky)

By integrating the risk checker smart contract, risky wallets were successfully identified and their deposits were forwarded to the risky subset ("risky" fund mixer smart contract), alternatively, those with small risk scores were forwarded to the regular privacy protocol's fund mixer smart contract. The results of determining the risk of the wallet (using the address as the parameter when interacting with the smart contract) and retrieving the risk score (absolute value – 18 decimal Ethereum standard on handling values without decimal point) are detailed in figures 5 and 6 respectively for both risky and non-risky wallets. Remix environment was used for making calls to the risk-checking smart contract.

Action	**Transaction**	**Total deposits**
Risky wallet deposit	https://goerli.etherscan.io/tx/0x5f654210e11dbfce98b9adce27460016ebcc0c9e1ee14876874bc612a2bd7b53	1
Regular wallet deposit	https://goerli.etherscan.io/tx/0x0b84bf1536be4f0f220b04bf6b7edea06583cb2e55f25ab720cc90b9fa66874e	11
Withdrawal from risky privacy protocol	https://goerli.etherscan.io/tx/0x1cb42e77702b4e97c76abe78cf50a1b23333f8522fb161bf416b3c9ab7d7d22a	0
Withdrawal from regular privacy protocol	https://goerli.etherscan.io/tx/0xaced6eda1450f716c4965e1c330e9c42e4b52100144a16b4c1395cbcbc547c07	10

Table 2. Blockchain transactions made using integrated risk checker smart contract.

Many deposits were made with various wallets, and most of them were forwarded to the regular mixer, however - risky wallet funds were routed to the smaller one, i.e. the "risky" fund mixer. From the user’s perspective, it is hard to determine where the funds will be deposited since the entrance point is the main risk checker contract. Transactions with different outcomes are depicted in Table 2. There is a higher chance to have a regular risk category (false, which means not risky), therefore more deposits are made in the regular mixer. During the withdrawal of tokens from the risky fund mixer - the total number of deposits is equal to zero (in this example), therefore it is also easier to trace potentially malicious withdrawal addresses to ones that performed illegal deposits.

Conclusions and Recommendations

To sum up, the developed model for fund traceability on blockchain privacy protocols can be valuable for integrating a solution that at the same time both keeps the anonymity and safety of users impact, while at the same time filtering out bad actors with AML (anti-money laundering) measures. The performed experiments and tests of the model indicate that the implementation of the model using a risk check smart contract would result in the prevention of suspicious users using the system by deterrence instead of participation disallowance, as the potential criminal origins of the funds would not be possible to hide by withdrawing to a newly created wallet with no history. This approach would allow anyone to use the blockchain privacy protocol for fund depositing and withdrawing, but at the same time make it more trustworthy by indicating the possible suspicious and undependable history of the cryptocurrency funds. Nonetheless, the solution can be improved in numerous ways, and more advanced methodologies may be applied, with more research and tests. Some of these improvements include the revision of the model, streamlining some of the elements, and introducing some own solutions, such as a personal risk score data oracle. The improvements are planned for future work. If this model were implemented for a privacy platform such as a fund exchange, the developers could opt for a more straightforward approach, i.e. simply block users with high-risk scores from performing any transaction-related actions, however it is recommended not to do that, as the risk score may be subjective in some cases and may block potential legitimate entities from using the platform. Therefore, the risk score could be considered more as a guideline for indicating the suspicious origin of funds and not as a definite and guaranteed indicator.

Future work plan

For future work, several improvements could be done to the model of fund traceability on the blockchain privacy protocols. The current theoretical model only relies on the risk score data from third-party oracle services, however, for higher overall system reliability, a personal oracle could be created for use along with the third-party oracles. The personal oracle service would feature its risk score calculation methodologies that could be adjusted according to the integration into a privacy platform. Furthermore, the theoretical model can be optimized by having a single fund mixer smart contract instead of two separate smart contracts for both "clean" and "risky" funds respectively. In this case, however, the issue of integrating the risk score into a transaction would have to be investigated more, so that when the entity deposits funds into a mixer, the risk score information is encrypted along with the key used for withdrawal of the funds, ensuring the fund traceability and retaining of their origins when deposited into a newly created wallet. These improvements would be greatly beneficial for the model's usage in more advanced blockchain privacy protocols, and the research, as well as further developments, are planned to be done in the final master thesis.

References

[1] G. Hileman and M. Rauchs, ‘‘Global cryptocurrency benchmarking study,’’ *Cambridge Centre for Alternative Finance*, vol. 33, pp. 33--113, 2017.

[2] M. Wu, W. McTighe, K. Wang, I. A. Seres, N. Bax, M. Puebla, M. Mendez, F. Carrone, T. De Mattey, H. O. Demaestri, *et al.*, ‘‘Tutela: An open-source tool for assessing user-privacy on ethereum and tornado cash,’’ *arXiv preprint arXiv:2201.06811*, 2022.

[3] S. Malwa, ‘‘Ronin exploiter moved 21,000 ether to tornado cash in past week.’’ https://www.coindesk.com/tech/2022/04/13/ronin-exploiter-moved-21000-ether-to-tornado-cash-in-past-week/, Apr 2022.

[4] S. Dyson, W. J. Buchanan, and L. Bell, ‘‘The challenges of investigating cryptocurrencies and blockchain related crime,’’ *arXiv preprint arXiv:1907.12221*, 2019.

[5] A. Kumar, C. Fischer, S. Tople, and P. Saxena, ‘‘A traceability analysis of monero’s blockchain,’’ in *European Symposium on Research in Computer Security*, pp. 153--173, Springer, 2017.

[6] S. Xu, ‘‘The application of machine learning in bitcoin ransomware family prediction,’’ in *2021 the 5th International Conference on Information System and Data Mining*, pp. 21--27, 2021.

[7] C. G. Akcora, Y. Li, Y. R. Gel, and M. Kantarcioglu, ‘‘Bitcoinheist: Topological data analysis for ransomware detection on the bitcoin blockchain,’’ *arXiv preprint [Web Link]*, 2019.

[8] N. Kshetri and J. Voas, ‘‘Do crypto-currencies fuel ransomware?,’’ *IT professional*, vol. 19, no. 5, pp. 11--15, 2017.

[9] Chainalysis, ‘‘Chainalysis blockchain sanctions analysis.’’ https://public.chainalysis.com/docs/index.html#definition-of-sanctioned-entities, 2022.

[10] Elliptic, ‘‘Elliptic - preventing and detecting criminal activity in cryptocurrencies..’’ https://www.elliptic.co/solutions/crypto-wallet-screening, 2022.

Appendices

The document contains two appendices:

- Appendix A which contains an example of how entity whitelisting functionality could be implemented for a cryptocurrency coin smart contract. The code block was written in Solidity programming language.
- Appendix B which contains the risk checker – functions for fetching the overall risk score and determining the risk factor of the entity, based on the final risk score. The code block was written in Solidity programming language as well.

A Example of Cryptocurrency Whitelist Functionality

```
mapping(address=>bool) isWhitelisted;
modifier onlyWhitelisted (address _user)
{
    require(isWhitelisted[_user], "Sender is not whitelisted");
    _;
}
function transfer(address _to, uint256 _value)
         onlyWhitelisted(msg.sender)
         public
         virtual
         override
         returns (bool)
{
    _transfer(msg.sender, _to, _value);
    return true;
}
```

B Risk Checker

```
    function riskScore(address wallet)
        public
        view
        returns (uint)
    {
        uint sum = 0;

        for(uint i = 0; i <= numberOfRiskOracles; i++){
            sum += riskScores[i][wallet];
        }
        return sum / numberOfRiskOracles;
    }

    function determineRisk(address wallet)
        public
        view
        returns (bool)
    {
      uint256 riskMean = riskScore(wallet);
        if(riskMean > 500000000000000000){
          return true;
        } else{
          return false;
        }
    }
```