Distribution Agreement

In presenting this dissertation as a partial fulfillment of the requirements for an advanced degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my dissertation in whole or in part in all forms of media, now or hereafter known, including display on the world wide web. I understand that I may select some access restrictions as part of the online submission of this dissertation. I retain all ownership rights to the copyright of the dissertation. I also retain the right to use in future works (such as articles or books) all or part of this dissertation.

_____ _____

Shuaicheng Ma Date

Understanding and Incentivizing Behavior in Emerging Decentralized Ecosystems

By

Shuaicheng Ma
Doctor of Philosophy

Department of Computer Science

Ýmir Vigfússon
Advisor

Li Xiong
Committee Member

Nosayba El-Sayed
Committee Member

Gregory Chockler
Committee Member

Accepted:

Kimberly Jacob Arriola
Dean of the James T. Laney School of Graduate Studies

August 25, 2023

Date

Understanding and Incentivizing Behavior in Emerging Decentralized Ecosystems

By

Shuaicheng Ma

Advisor: Ýmir Vigfússon

An abstract of
A dissertation submitted to the Faculty of the
Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Department of Computer Science
2023

Abstract

Understanding and Incentivizing Behavior in Emerging Decentralized Ecosystems
By Shuaicheng Ma

This dissertation explores the application of blockchain technology and the incentivization of behavior in emerging decentralized ecosystems. We investigate its application in the healthcare sector, with a particular focus on genomic data sharing. We also examine its potential in deterring illicit activities such as cryptocurrency fraud and explore its role in confidential tracking within decentralized delivery systems.

We propose efficient strategies for data storage and retrieval in blockchain systems, specifically targeting cross-site genomic data sharing. Our blockchain-based log system provides a lightweight and widely compatible module for existing blockchain platforms. By ensuring accountability in cross-site genomic data sharing, we demonstrate the feasibility of blockchain technology in incentivizing responsible behavior and enhancing collaboration across different healthcare entities.

In addressing the issue of illicit activities in the blockchain ecosystem, we design a virtual taint system that marks cryptocurrency transactions as "tainted" if they are known to be involved in crime, fraud, or other illicit activities. This system serves to disincentivize such activities, revitalizing integrity within the cryptocurrency ecosystem.

Furthermore, we present a blockchain-based system for the confidential tracking of routes in decentralized delivery systems. By leveraging the transparency of blockchain while preserving business confidentiality, this system reveals necessary information only when fraud occurs, thereby incentivizing honest behavior and enhancing trust within the network.

Our research contributes significantly to both the understanding and incentivization of behavior in emerging decentralized ecosystems, particularly within the context of blockchain technology. The findings pave the way for secure and efficient data management solutions across various sectors and contribute to the creation of a safer and more secure digital environment.

Understanding and Incentivizing Behavior in Emerging Decentralized Ecosystems

By

Shuaicheng Ma

Advisor: Ýmir Vigfússon

A dissertation submitted to the Faculty of the
Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in Department of Computer Science
2023

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

The principle of decentralization has been a cornerstone in the field of computer science since the birth of the Internet in the 1960s [10]. This paradigm is manifest in the evolution of a multitude of systems, including the Domain Name System (DNS) [56], Email [79], Tor [34], and most recently, blockchain technology [60]. The ongoing pursuit among computer scientists is to engineer decentralized systems that not only demonstrate resilience against single-point failures, but also maintain high performance and engender trust.

In 2008, an innovation occurred in the field of decentralization with the emergence of the first blockchain system, Bitcoin, by an individual or group using the pseudonym Satoshi Nakamoto [60]. This system, use incentive and cryptographic data structure to ensure the trust of the system is not only accessible to everyone and resilient to centralization, but also trustworthy in a potentially malicious environment. The principles upon which Bitcoin is founded - decentralization, transparency, and immutability - have subsequently become the bedrock for all other blockchain systems [84, 20, 86].

The dissertation explores the multifaceted applications and challenges of blockchain technology, focusing on its potential for secure and efficient healthcare data storage, mechanisms to deter malicious behavior within the blockchain ecosystem, and the im-

plementation of cryptographic schemes to balance transparency with privacy, thereby contributing to the advancement of decentralized systems that foster trust and resilience.

1.1 Understanding Blockchain as Database for Healthcare Data

Bitcoin, the inaugural application of blockchain technology, epitomizes the core functionality of this technology. It operates as a public, immutable database, specifically engineered to function in an environment where participants may not necessarily trust each other, or may even harbor malicious intent. Bitcoin's primary purpose is to securely record critical financial transactions, demonstrating the potential of blockchain technology in ensuring data integrity and security in a trustless environment.

Data replication is an essential characteristic of blockchain systems, ensuring their resilience in potentially malicious environments. However, this replication process can significantly inflate storage costs, posing a challenge for the efficient use of blockchain technology. This is particularly pertinent in the context of healthcare data storage, where the volume of data can be substantial, and the need for secure and efficient data management is critical.

In this dissertation, we focus on the application of blockchain technology for healthcare data storage. We propose efficient data storage and retrieval strategies that are tailored to the unique characteristics of blockchain systems [47]. We also provide a comprehensive benchmarking [45] of existing solutions for storing healthcare data on blockchain. Through our research, we strive to enhance the practicality and feasibility of using blockchain technology for medical data storage, thereby contributing to the advancement of secure and efficient data management solutions in the healthcare sector.

1.2 Incentivizing Away from Malicious Behavior in Blockchain Ecosystem

Technological advancements, while delivering numerous benefits, often inadvertently create avenues for potential misuse. For instance, Distributed Denial of Service (DDoS) attacks can inundate websites or networks with excessive traffic, leading to service disruptions or even extortion attempts. Similarly, AI Deepfake technology [75] can generate highly realistic fake videos or images, which are frequently exploited for malicious purposes such as spreading disinformation, defaming individuals, or manipulating public opinion.

This pattern of misuse is also evident in the realm of blockchain advancements. The technology has been exploited in illicit use cases, such as cryptocurrency money laundering services [13, 12], Darknet marketplaces [57, 27], and facilitating ransomware payments [37, 76]. The inherently decentralized nature of blockchain not only facilitates the deployment of these illicit activities but also makes them difficult to trace and dismantle.

Recognizing the potential of blockchain technology and its significance for the future, we specifically address the issue of cryptocurrency money laundering in this dissertation. We propose and implement mechanisms that monitor transaction traffic, with the aim of steering users away from illicit activities. Our work contributes to creating a safer and more secure digital environment. In this environment, the benefits of technological advancements can be fully realized without being overshadowed by their potential for misuse.

1.3 Implementing Cryptographic Schemes for Enhanced Blockchain Privacy

The inherent immutability of blockchain technology renders it highly effective for storing tamper-evident data. This characteristic has been leveraged in various domains, such as the secure storage of confidential business information [49], health records [45], and database access logs [47]. The ability to maintain an unalterable record of data ensures the integrity and authenticity of the information stored, thereby enhancing trust among the users of the system.

However, a fundamental characteristic of blockchain is its transparency, which means that, by default, all data on the blockchain are visible to all participants. This transparency, while beneficial for ensuring accountability and traceability, also means that a user could potentially access data that do not belong to them. This could pose a significant concern, particularly in scenarios where the data stored on the blockchain is sensitive or confidential.

To mitigate this issue, we propose multiple cryptographic schemes proposed [49, 48] to confidentially tracking routes in a decentralized international phone call delivery system. These schemes aim to strike a balance between the transparency that is inherent to blockchain and the need for privacy and confidentiality. They employ various cryptographic techniques to ensure that while data is stored on the blockchain in a transparent manner, access to the specifics of the data is controlled and limited to authorized individuals. This way, the integrity and traceability of the data are maintained, while also ensuring the privacy and confidentiality of the data.

1.4 Contributions

In summary, the contributions of this dissertation are listed as follows:

- We propose several efficient data storage and retrieval strategies tailored for blockchain systems, with a specific focus on medical data storage. These strategies strike a balance between data replication and the performance of data retrieval. Our work significantly enhances the practicality and feasibility of employing blockchain technology for medical data storage. Consequently, it contributes to the advancement of secure and efficient data management solutions in the healthcare sector, paving the way for more robust and reliable medical data systems.

- We address the issue of cryptocurrency money laundering. We design and implement mechanisms that monitor transaction traffic, aiming to deter users from engaging in illicit activities. Our work plays a crucial role in creating a safer and more secure digital environment, where the advantages of technological advancements can be fully leveraged without the risk of misuse.

- We propose multiple cryptographic schemes that balance the need for transparency with privacy and confidentiality. These schemes employ various cryptographic techniques to control and limit access to data specifics to authorized individuals, while maintaining the integrity and traceability of the data. Our contributions in this area enhance the security and privacy of blockchain technology, making it more suitable for a wider range of applications.

1.5 Dissertation Organization

Chapter 2 offers an in-depth overview of the blockchain architecture and its surrounding ecosystem. Chapter 3 presents efficient strategies for the storage and retrieval of medical data on blockchain, and offers a comprehensive benchmarking of existing solutions in this domain. Chapter 4 delves into the problem of cryptocurrency money laundering, proposing a robust system to monitor and deter such illicit activities within

the blockchain. Chapter 5 details the construction of a system, FRAUD BUSTER, which confidentially traces fraudulent call drops within an international telecommunication carrier network. This system leverages several cryptographic schemes, striking a balance between transparency and privacy within the blockchain environment. Finally, Chapter 6 concludes the thesis, summarizing the key contributions and outlining potential future directions in this rapidly evolving field.

Chapter 2

Background

2.1 Blockchain Architecture

Blockchain technology, a decentralized, public, and tamper-evident database, is illustrated in Figure 2.1. This figure delineates the operational cycle of a blockchain, where each participant in the network maintains an identical local copy of the database.

2.1.1 Operational Cycle

In its rudimentary implementation, the operational cycle of a blockchain consists of the following steps:

1. A leader is chosen through a selection algorithm to propose a new block of transactions.

2. Other participants verify the transactions.

3. If the transactions are deemed valid, all participants update their local databases accordingly.

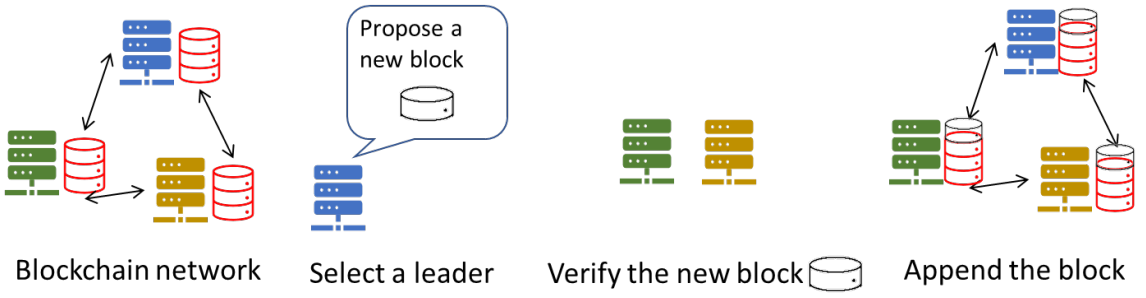


Figure 2.1: Blockchain Architecture

2.1.2 Transaction Process

A transaction in a decentralized blockchain network is initiated by a node through the application of a digital signature, facilitated by private key cryptography. These transactions are initially stored in an unconfirmed transaction pool before being disseminated across the network via a flooding protocol, known as the Gossip protocol.

Bitcoin Transaction

The primary purpose of Bitcoin, a digital cryptocurrency, is to allow a decentralized electronic cash payment system between different parties by eliminating central intermediaries. The main element of a bitcoin structure is unspent transaction output (UTXO), which refers to the unused output amount of a transaction.

Ethereum Transaction

Ethereum transactions enable the creation and execution of smart contracts, which are self-executing agreements coded directly onto the blockchain. Written in Solidity [8] and running on the Ethereum Virtual Machine (EVM), smart contracts automate various decentralized applications (DApps), such as decentralized finance (DeFi) and asset tokenization. They are deterministic and require "gas" paid in Ether for execution. While offering powerful interoperability and trustless automation, smart contracts also present challenges in security, scalability, and complexity.

Property	Public	Consortium	Private
Consensus determination	All miners	Selected set of nodes	One organization
Read permission	Public	Public or restricted	Public or restricted
Immutability	Nearly impossible to temper	Could be tampered	Could be tampered
Efficiency	Low	High	High
Centralized	No	Partial	Yes
Consensus process	Permissionless	Permissioned	Permissioned

Table 2.1: Comparisons between public blockchain, consortium blockchain, and private blockchain

2.1.3 Characteristics of Blockchain

Blockchain technology is characterized by the following key features:

- **Decentralization**: Transactions can be conducted between any two peers (P2P) without authentication by a central agency.
- **Persistency**: It is nearly impossible to tamper with transactions as they are confirmed and recorded in blocks distributed throughout the network.
- **Anonymity**: Users interact with the blockchain network with a generated address, preserving a certain amount of privacy.
- **Auditability**: All transactions are recorded by a digital distributed ledger, making it possible to audit and trace previous records.

2.1.4 Taxonomy of Blockchain Systems

Blockchain systems can be categorized into public, private, and consortium types. Table 2.1 compares these systems from different perspectives, including consensus determination, read permission, immutability, efficiency, centralization, and consensus process.

2.2 Blockchain Applications

Blockchain technology has diverse applications, including finance, public and social services, and reputation systems.

2.2.1 Finance

- *Financial Services.* The advent of blockchain systems like Bitcoin and Hyperledger has revolutionized traditional financial and business services. Peters et al. [65] emphasized that blockchain has the potential to disrupt the banking sector, with applications spanning across areas such as the clearing and settlement of financial assets. Morini [58] further illustrated how real business cases, such as collateralizing financial derivatives, could leverage blockchain to minimize costs and risks. This transformative technology has not only caught the attention of large software companies like Microsoft Azure and IBM, who are now offering Blockchain-as-a-Service, but also paved the way for innovative financial solutions.

- *P2P Financial Market.* Blockchain's secure and reliable nature can facilitate the creation of a Peer-to-Peer (P2P) financial market. Noyes [63] investigated the integration of peer-to-peer mechanisms with multiparty computation protocols to develop a P2P financial MPC (Multiparty Computation) market. This blockchain-based MPC market enables the distribution of computational tasks across a network of anonymous peer-processors, fostering a new era of decentralized financial interactions.

2.2.2 Public and Social Services

Blockchain technology extends beyond financial applications and has found significant use in public and social services, enhancing efficiency, security, and transparency.

- *Land Registration.* A prominent application of blockchain in public services is land registration. By recording land information, including physical status and associated rights, on a blockchain, transparency and accessibility are greatly improved. Any modifications to the land, such as transfers or mortgage establishments, can be securely recorded and managed on the blockchain. This not only streamlines administrative processes but also enhances the integrity and reliability of land records.

- *Free-Speech Rights.* Blockchain's decentralized nature can be leveraged to secure internet infrastructures like DNS and online identities. Namecoin, an experimental open-source technology, exemplifies this by enhancing the decentralization, security, censorship resistance, privacy, and speed of DNS and identities. By making the web more resistant to censorship, it plays a vital role in protecting free-speech rights online, fostering a more open and resilient internet landscape.

2.2.3 Reputation System

Reputation systems play a crucial role in various online platforms, serving as a measure of trust and credibility for users. The reputation score, often derived from previous transactions and interactions, reflects how trustworthy a user is perceived by the community. However, this system is not without flaws, and there have been increasing instances of reputation manipulation and falsification. For example, in the e-commerce sector, some service providers have been known to enlist fake customers to artificially inflate their reputation scores. Blockchain technology offers a promising solution to this challenge.

- *Immutable Records.* By utilizing blockchain's immutable ledger, reputation scores and associated transactions can be recorded securely. This ensures that once a reputation-related transaction is recorded, it cannot be altered or deleted,

thereby preventing fraudulent manipulations.

- *Transparency and Accountability.* Blockchain's transparent nature allows all participants to view and verify the history of transactions. This fosters a more honest and accountable reputation system, where users can trust the authenticity of the recorded reputation scores.

- *Decentralized Control.* Unlike centralized reputation systems, where a single entity has control, blockchain's decentralized structure distributes control across the network. This reduces the risk of biased or unfair manipulation, promoting a more equitable and robust reputation system.

By integrating these features, blockchain can revolutionize reputation systems, enhancing their integrity, reliability, and fairness, and providing a more secure and trustworthy environment for users across various platforms.

2.3 Tradeoffs and Challenges in Blockchain Technology

Blockchain technology, while revolutionary, faces several challenges and tradeoffs that must be addressed to realize its full potential. These include scalability, privacy leakage, interoperability, and regulatory issues.

2.3.1 Scalability

Scalability remains a significant challenge for blockchain technology. As the number of transactions increases, the blockchain's size grows, leading to storage and processing limitations. For example, the Bitcoin blockchain has exceeded 300 GB, and its processing capacity is limited to nearly seven transactions per second. This limitation

hampers the ability to process millions of transactions in real-time. Efforts to address scalability include:

- *Storage Optimization.* Techniques such as removing old transaction records and using an account tree to hold balances can reduce storage requirements. Lightweight clients and schemes like VerSum [81] also contribute to solving this problem.

- *Redesigning Blockchain.* Proposals like Bitcoin-NG [36] decouple blocks into key blocks and microblocks, addressing the tradeoff between block size and network security.

2.3.2 Privacy Leakage

While blockchain is often considered secure due to the use of generated addresses instead of real identities, research has shown that this does not guarantee complete privacy. Studies by Meiklejohn et al. [54] and Kosba et al. [42] have revealed that the values of all transactions and balances for each public key are publicly visible, undermining anonymity. Furthermore, methods have been developed to link user pseudonyms to IP addresses, even behind network address translation (NAT) or firewalls [25]. This can lead to the identification of the origin of a transaction. Efforts to improve blockchain anonymity include:

- *Mixing Services* [59]. These services enhance privacy by transferring funds between multiple input and output addresses. For example, Alice can send funds to Bob through an intermediary, Carol, who mixes the transaction with others, making it harder to trace. Mixcoin [26] provides a method to prevent dishonest intermediaries by encrypting users' requirements with a private key, allowing verification if the intermediary cheats.

- *Anonymous Protocols.* Solutions like Zerocoin [55] use zero-knowledge proofs to unlink payment origins from transactions, although they may still reveal destinations and amounts. Zerocash further addresses this by leveraging zero-knowledge Succinct Non-interactive Arguments of Knowledge (zk-SNARKs), hiding transaction details.

This subsection highlights the complexity of ensuring privacy in blockchain transactions and the ongoing efforts to enhance anonymity. It underscores the need for robust solutions to protect user information and maintain the integrity of the decentralized system.

2.3.3 Interoperability

Interoperability, or the ability of different blockchain systems to work together, is a significant challenge in the blockchain industry. Despite widespread interest across various industries in adopting blockchain technology [2], the absence of a standard protocol hinders collaboration and integration. This lack of interoperability not only stifles the growth of the blockchain industry but also restricts its application beyond cryptocurrencies. While the current landscape allows blockchain developers the freedom to code using various platforms, languages, consensus mechanisms, and protocols, it results in isolated networks that cannot interact with one another. For instance, GitHub hosts over 6500 active blockchain projects, each with unique characteristics, reflecting the fragmentation in the field. To unlock the full potential of blockchain across diverse business models, standardization is imperative. It would enable collaboration, facilitate the sharing of blockchain-based solutions, and allow seamless integration with existing systems, thereby fostering innovation and expanding the reach of blockchain technology.

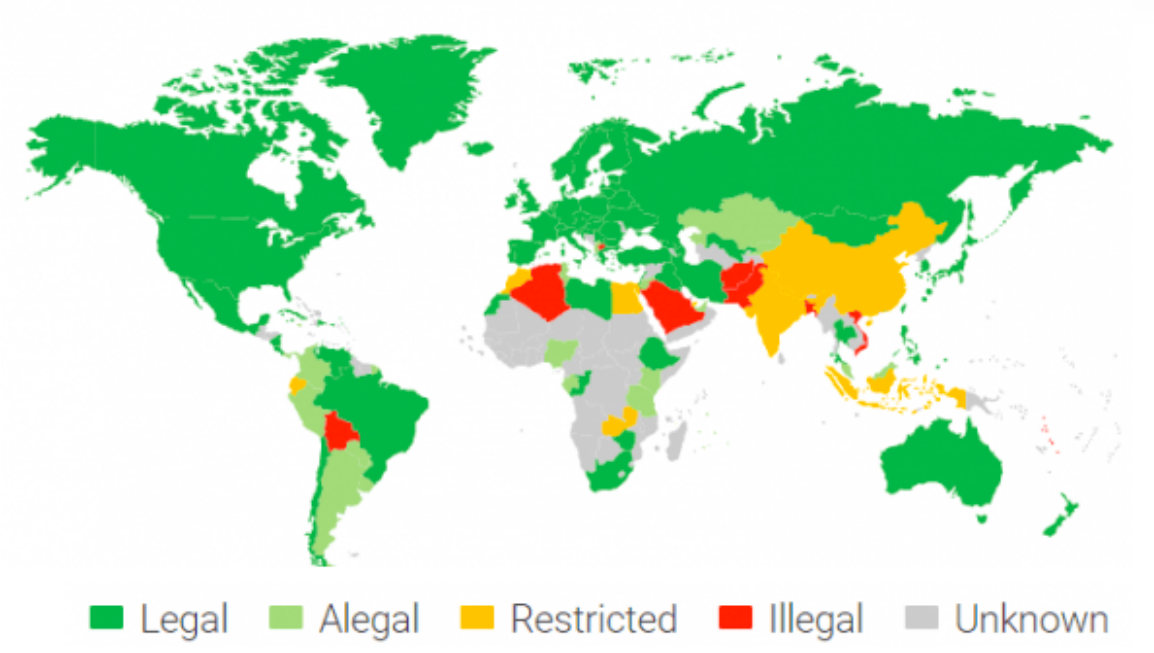


Figure 2.2: Bitcoin Legality Around the World

2.3.4 Current Regulation Problems

Regulatory challenges stem from the decentralized nature of blockchain, which can undermine central banks' control over economic policy. This has led to varying legal stances on cryptocurrencies, with some countries banning them outright. Fig. 2.2 illustrates the global legality of Bitcoin, and studies like [87] highlight the regulatory challenges that impact distributed technologies, particularly in the EU and the USA.

Chapter 3

Exploring Blockchain's Behavior in Healthcare Data

3.1 Benchmarking blockchain-based gene-drug interaction data sharing methods

Blockchain distributed ledger technology is just starting to be adopted in genomics and healthcare applications. Despite its increased prevalence in biomedical research applications, skepticism regarding the practicality of blockchain technology for real-world problems is still strong and there are few implementations beyond proof-of-concept. We focus on benchmarking blockchain strategies applied to distributed methods for sharing records of gene-drug interactions. We expect this type of sharing will expedite personalized medicine.

3.1.1 Gene-drug interaction data

Genetic variation is known to affect drug response. Presence of specific genetic variants can result in variability of drug efficacy and adverse drug reactions (ADR) through alternate pharmacokinetic (PK) and pharmacodynamic (PD) pathways. One such

example is warfarin, an anticoagulant commonly used to prevent or treat blood clots. It is notoriously challenging to correctly adjust warfarin doses due to interpatient variability resulting from both clinical data (e.g., age, sex, race, body mass index, conditions, and other medications) and genetics (e.g., variants in VKORC1, CYP2C9, and CYP4F2 genes) [33]. While patients with AA genotype in SNP rs9923231 of the VKORC1 gene are sensitive to warfarin and require lower doses, those with AG or GG genotypes are less sensitive. Complications arising from inadequate warfarin dosing constitute some of the most common ADRs reported to the Food and Drug Administration (FDA) [14]. For this reason, warfarin has been added to the FDA list of drugs with pharmacogenomics labeling; the recent list has 304 unique drugs [9].

Gene-drug relationship data are very important for clinicians and researchers. There are several publicly available gene-drug interaction datasets, such as the one produced by the Clinical Pharmacogenetics Implementation Consortium (CPIC) [5]. Based on these datasets, researchers may evaluate and investigate interactions for their associations with specific patient outcomes (e.g., improved, unchanged, or deteriorated), suspected gene-outcome-relations (e.g., yes, or no), and serious side-effects (e.g., yes, or no). However, these evaluation results may be siloed within an institution. A mechanism for institutions to share the evaluation results of the gene-drug interactions they obtained locally could help speed up research.

With the advance of sequencing technology, genetic testing is becoming more available, making pharmacogenetic-based drug dosing more viable in clinical practice. CPIC is one such effort to provide peerreviewed, updated, and evidence-based guidelines for gene-drug pairs. However, a level 1 quality guideline in CPIC requires consistent evidence, with large sample sizes in well-designed and well-conducted studies. Gathering sufficient and high-quality evidence of gene-drug outcomes is still a daunting task due to technical, economic, administrative, and ethical reasons.

3.1.2 Traditional methods and threat models

Intuitively, we can adopt a centralized method that uses a central server and collects the evaluation results (Fig. 3.1) via a traditional local software program performing logging/querying operations (Fig. 3.3). However, this setting could introduce multiple threats. As shown in previous studies [50, 43, 88], a central server and traditional program can present the barriers/challenges listed below:

1. Single-point-of-failure (e.g., the whole system stops working when the server stops due to a routine maintenance or a malicious attack).

2. Mutable data (e.g., the information on the server may be altered by the "root" user).

3. Unverifiable data source (e.g., the sources of the evaluation results may also be changed on the central server).

4. Non-transparent software (e.g., unspecified changes and thus inconsistent code).

5. Alterable programs (e.g., the deployed program can still be altered locally).

To overcome these issues, we consider a decentralized architecture to solve the above mentioned risks brought by a central server and traditional program. This architecture enables consistent and large-scale evidence gathering from multiple participating hospitals and individuals. Among the decentralized data storage methods, blockchain [8–11] is one of the more promising candidates (Fig. 3.2). The latest blockchain platforms, such as Ethereum [84], Hyperledger Fabric [20], support smart contracts, (Fig. 3.4) which are computer programs running on blockchain. The desired technical properties of blockchain with smart contracts include:

1. No single-point-of-failure (i.e., it is peer-to-peer).

2. Immutable data (i.e., it is very difficult to change the data on the chain).

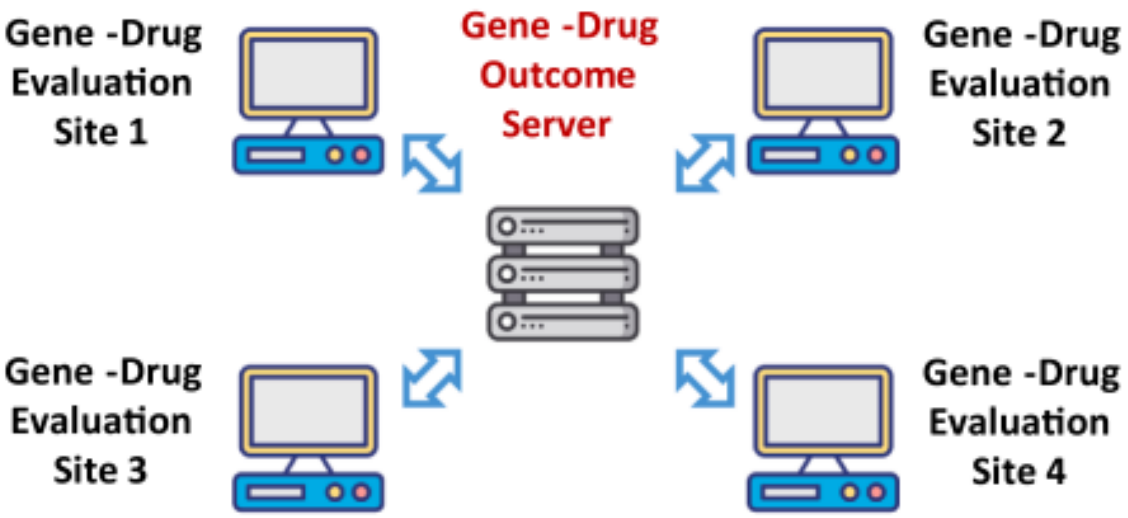


Figure 3.1: Centralized architecture (central server) where the centralized gene-drug outcome server can lead to a single-point-of-failure. The central server can change the records from other sites and can even modify the source of evaluation results.

3. Data provenance (i.e., the source of data is confirmed and therefore cannot be falsified).

4. Transparent software (e.g., each software change can be verified and confirmed).

5. Unchangeable program code (e.g., the deployed program is not alterable, and new versions of the program are recorded and visible to all nodes).

Therefore, using smart contracts on blockchain to store and query patient outcomes related to gene-drug data pairs could further improve the transparency and immutability of the software among the participating institutions. Blockchain has been proposed in various healthcare, genomic and biomedical applications [50, 43, 88], such as medical record management [21, 68, 82], dynamic consent in biobanking [52], genomic data access logging [44], pharmaceutical supply chain [77]. Meanwhile, applications in pharmacogenetics are still limited [68]. While blockchain has been the underlying infrastructure for cryptocurrencies such as Bitcoin [60] for more than a decade, the design and usability of blockchain have yet to be well understood in

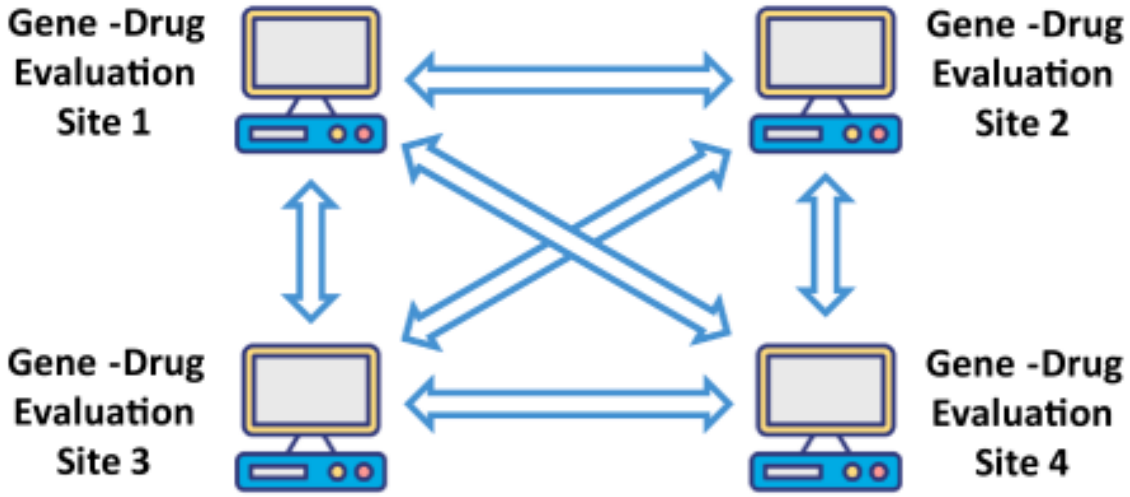


Figure 3.2: Decentralized architecture (blockchain) without a central server that can eliminate the possibility of a single point-of-failure. By adopting blockchain technology, the data are immutable and source-verifiable.

health sciences as they currently are in the world of finance. Although the idea of adopting blockchain and smart contracts for sharing gene-drug evaluation results may conceptually be feasible, practical issues in implementing such a system have yet to be investigated. Many blockchain-based solutions are still in early stages and the resources to support blockchain and smart contract developers are also scarce. Therefore, we aim at benchmarking the potential of a decentralized gene-drug system on blockchain, with smart contracts.

3.1.3 Methods

Data

The dataset for benchmarking was generated using the gene-drug relationship data from CPIC [5]. Each record contained the following six fields (Table 3.1): gene name, variant number, drug name, outcome, suspected gene outcome relation, and serious side effect. First, we obtained 127 unique gene names and 226 unique drug names from CPIC and randomly chose one gene name and one drug name as a pair to generate

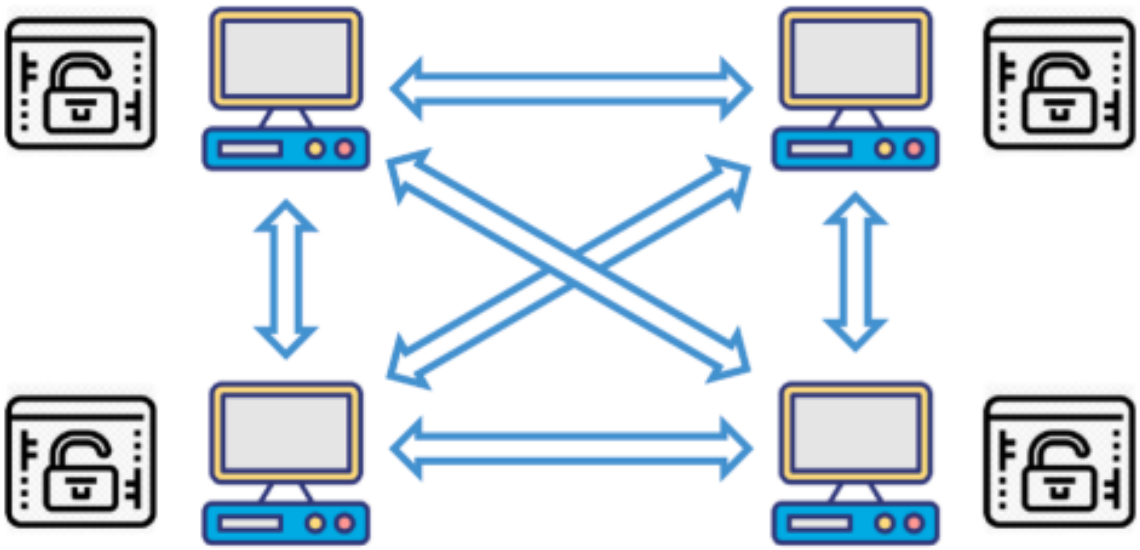


Figure 3.3: Traditional off-chain program that is non-transparent and mutable.

Table 3.1: Description of a record in our dataset.

Field	Possible Values	Example
Gene Name	127 unique gene names	HLA-B
Variant Number	1–99	57
Drug Name	226 unique drug names	abacavir
Outcome	Improved, Unchanged, or Deteriorated	Improved
Suspected Gene Outcome Relation	Yes or no	Yes
Serious Side Effect	Yes or no	No

a record. Next, for each record, we selected a variant number [1–99], an outcome status [Improved, Unchanged, Deteriorated], a suspected gene outcome relation [Yes, No], and a serious side effect [Yes, No], all randomly. For the development process the teams were provided with four patient outcomes of gene-drug pair files, each of which with 10,000 records representing the observed patient outcome for a gene-drug pair from four institutions. During the evaluation process we utilized 200 and 1,000 records from each of the four sites.

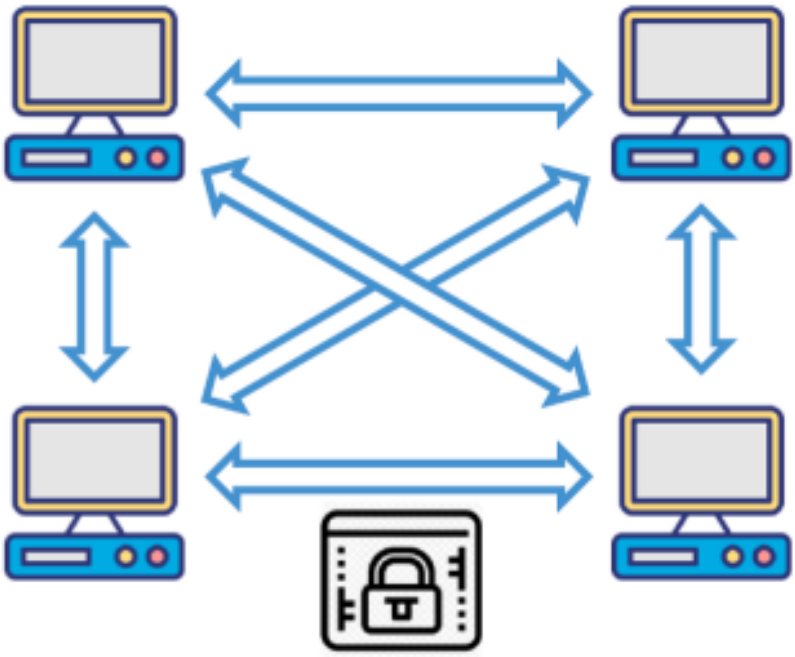


Figure 3.4: On-chain smart contracts that are transparent and immutable among the sites.

Query Index

The first method, Query Index, was a domain knowledge-based approach to implement a storage and query efficient solution. The following two kinds of domain knowledge in the gene-drug interaction data sharing were utilized in the design of an efficient solution: (1) the query output is the accumulated statistics of the gene-drug interaction data, and (2) the amount of unique gene-drug relations (i.e., approximately 106 in CPIC specification) is much smaller than the amount of raw gene-drug interaction records. This implementation utilized the above two facts, stored the statistical information of all unique genedrug relations (i.e., gene-variant-drug triples) in an upper-bounded size array and cached all indices in a hash table for fast insertion and query. Fig. 3.5 illustrates an example of the array and hash table data structure of Query Index. Every gene-variant-drug triple could be invoked in 8 different types of queries (i.e., a query specifying gene name, drug name, and variant number and 7 queries with wildcard characters in different fields). For example, the result of

GBA-nicotine74 will be returned in query (GBA, nicotine, 74), query (GBA, *, *), query (*, *, *), and so on. Based on this small number of query fields, a key-value hash table was built to support all possible queries. In the hash table, the keys were gene-variant-drug tuples and their wildcard alternatives, and the values were the indices of the actual information in the array. Upon receiving a query request, the Query Index method first found the matching index list in the hash table if the record existed, then traversed the indices to retrieve the actual information from the array. For the insertion, with the help of the hash table, the method could locate the index of the gene-variant-drug tuple in the array in O(1) time and update the counts. If the record did not exist, the method would append the record at the end of the array and insert corresponding entries in the hash table.

Index Everything

The second method, Index Everything, was a straightforward implementation approach. Since there were only a few hundred distinct genes and drugs, a unique 8-bit unsigned integer (uint8) value was assigned to each distinct gene (respectively, drug) value. These values were assigned lazily, i.e., the next available ascending value was assigned upon the first insert containing that gene or drug. As such, a unique 24-bit unsigned integer (uint24) could be trivially derived for each genevariant-drug triple, specifically by concatenating the corresponding three uint8s. Thus, for any observation, this uint24 derived by concatenation was used as an index into various outcome counts stored in the Solidity mapping structures. This indexing/storage scheme is illustrated in Fig. 3.6. The two query modalities (entryExists and query) implementations were similarly straightforward. Specifically, given the wildcard value ('*') in any position, all possible values were searched for that position, expressed as a triple for which any non-wildcard search value collapsed the specific dimension.

Array

Data*	PROC, 60, aspirin	UGT2B, 34, aspirin
Index	0	1

Data*: The complete data structure is { gene name, variant, drug name, total outcome count, improved count, unchanged count, deteriorated count, suspectedRelation count, sideEffect count }.

Hash Table

Key	
PROC, 60, aspirin	0
,, aspirin	0,1
, 60,	0
*, 60, aspirin	0
PROC,*,*	0
PROC,*, aspirin	0
PROC, 60,*	0,
UGT2B, 34, aspirin	1
UGT2B, 34,*	1
UGT2B,*, aspirin	1
*, 34, aspirin	1
UGT2B,*,*	1
, 34,	1
,,*	0,1

Figure 3.5: Example of two records for the Query Index method.

Dual-Scenario Indexing

The third method, Dual-Scenario Indexing, adopted a special data structure to store gene-drug relationship data. It was also assumed here that query operations (such as query and entryExists) were more frequently invoked than insert operations, thus the team focused on query performance optimizations. Two different data structures were used to support the precise search with all three given inputs (gene name, variant number and drug name) and the search with wildcard inputs under two scenarios: complete (i.e., gene-variant-drug) and wildcard searches. For the complete search scenario, a mapping structure named geneData mapping was used to store all GeneDrugRelation items with a key that was the concatenation of gene name A, variant

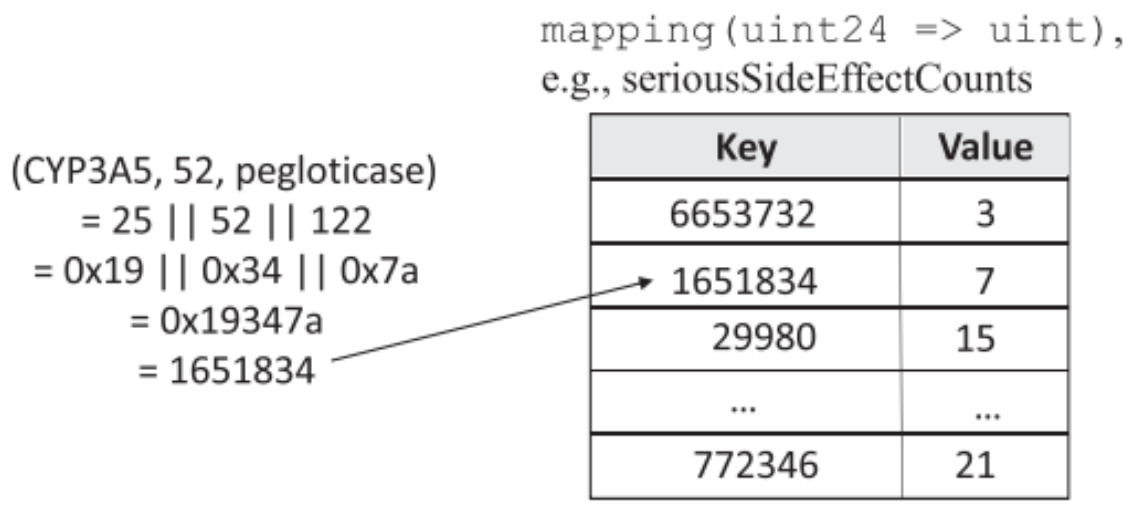


Figure 3.6: Visual depiction of the scheme of the Index Everything method (|| denotes integer concatenation) on the left, and an example mapping data structure counting side effects for each unique gene/variant/drug triple on the right. Structures like the one on the right exist for all observation categories: improved, unchanged, deteriorated, suspected relation, and side effect.

number B and drug name E. Therefore, the geneData map could easily support all queries with "ABE" inputs. For the wildcard search scenario, the team built a special mapping structure GeneDrugRelationKeyMapping with keys of wildcard search strings (e.g., "AB*") and values of the complete search strings (e.g., "ABE", the keys of the geneData data structure). The algorithm then pre-generated all possible combinations of geneData mapping keys for each wildcard input, and stored these combinations into the GeneDrugRelationKeyMapping data structure . For querying, the algorithm first searched GeneDrugRelationKeyMapping by "AB*" to get all geneData keys (e.g., "ABE" and others) that correspond to GeneDrugRelation items with A and B. Then, it searched geneData mapping to get the detailed GeneDrugRelation items. An example explaining how GeneDrugRelationKeyMapping supports wildcard query operations is shown in Fig. 3.7

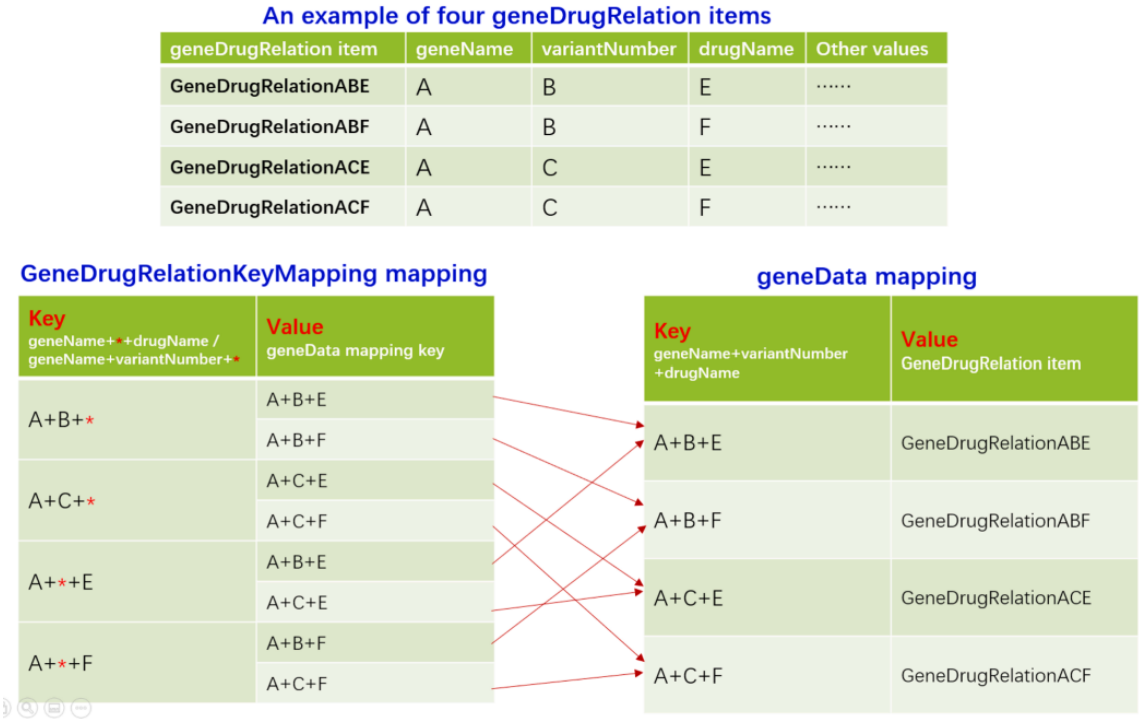


Figure 3.7: Key data store structure of the Dual-Scenario Indexing method.

3.1.4 Results

Evaluation

To evaluate the solutions, we inserted the two datasets (i.e., 200 and 1,000) to the blockchain either 1 or 200 records at a time to simulate different insertion speed and generated 60 queries to compute the query time required by each solution. Our evaluation criteria specified that: (a) a solution must complete the insertion of all records, (b) a solution must provide 100% correct query results, and (c) the speed of insertion and query is the most important feature, followed by storage and memory cost, and then scalability. Therefore, after checking the completeness and correctness of the solutions, we measured the insertion time, query time, disk storage, and memory usage, and then normalized these measurements to raw scores from 0 to 100. The raw scores were then weighted-summed to a subtotal score (insertion time = 35%, query time = 35%, disk usage = 15%, and memory usage = 15%). Next, the subtotal

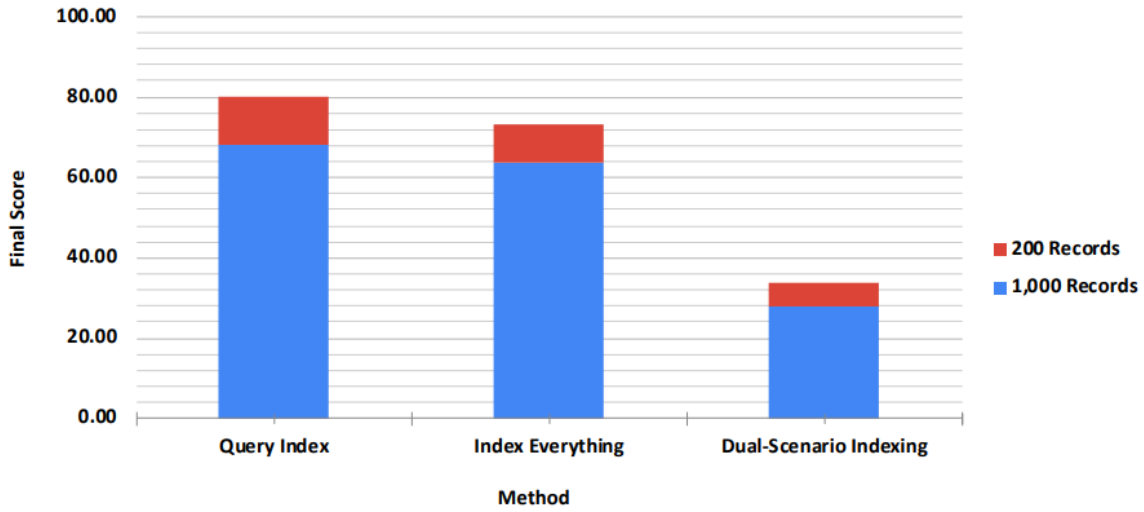


Figure 3.8: Final scores for each solution. The results were weighted based on the number of records in the test data (i.e., 200 records in red and 1,000 records in blue) and were averaged from the results of inserting 1 or 200 records at a time. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

scores were weighted-summed to an overall score, with the weights corresponding to the number of test records (i.e., 200 and 1,000) to account for scalability. Finally, the overall scores for inserting 1 and 200 records at a time were averaged to generate the final scores. We set up 24 Virtual Machines (VMs) to evaluate the solutions. Each VM had 2 CPU cores, 8 GB of RAM and 100 GB of storage; Ubuntu was the operating system.

Measurement results and final scores

Results and the scores are summarized in Table 3.1 and Fig. 3.8 respectively. As shown in the tables, inserting 200 records at a time reduced insertion time per record significantly. Also, while the insertion time increased linearly with the number of records in the test data, query times were more consistent, which could reflect the blockchain characteristic that writing is relatively slow (because it requires consensus block creation), while reading is fast (only local blocks are searched). The required disk space (<40 MB) and memory (<300 MB) were relatively small. In terms of

Solution	Query Index	Index Everything	Dual-Scenario Indexing
Number of Total Records per Site = 200, Number of Records per Insert = 1			
Complete	Yes	Yes	Yes
Correct	Yes	Yes	Yes
Runtime of Insertion (s)	212.75	226.75	203.00
Runtime of Query (s)	23.00	28.00	30.00
Storage Usage (MB)	21.18	21.23	21.19
Memory Usage (MB)	56.54	56.44	106.11
Number of Total Records per Site = 200, Number of Records per Insert = 200			
Complete	Yes	Yes	Yes
Correct	Yes	Yes	Yes
Runtime of Insertion (s)	13.25	4.75	11.25
Runtime of Query (s)	23.00	28.00	29.50
Storage Usage (MB)	19.49	19.46	19.50
Memory Usage (MB)	90.32	73.26	106.06
Number of Total Records per Site = 1000, Number of Records per Insert = 1			
Complete	Yes	Yes	Yes
Correct	Yes	Yes	Yes
Runtime of Insertion (s)	1006.75	1157.25	1003.50
Runtime of Query (s)	24.00	29.00	53.00
Storage Usage (MB)	31.44	32.98	31.36
Memory Usage (MB)	59.41	58.87	226.14
Number of Total Records per Site = 1000, Number of Records per Insert = 200			
Complete	Yes	Yes	Yes
Correct	Yes	Yes	Yes
Runtime of Insertion (s)	51.50	12.25	49.75
Runtime of Query (s)	24.00	29.00	53.00
Storage Usage (MB)	24.99	24.09	25.11
Memory Usage (MB)	110.10	103.72	225.73

Table 3.2: Results of each solution with different combination scenarios of records in test data (i.e., 200 versus 1,000) and number of records inserted at a time (i.e., 1 versus 200).

final scores, the Query Index method performed the best, followed by the Index Everything method. The Dual-Scenario Indexing method used more memory, and its insertion/query time and disk usage were comparable with those of other solutions.

Comparison of the three proposed methods

To further understand the differences between our three proposed methods, we analyzed the results in Table 3.1 for each of our proposed methods as follows. The storage usage for all solutions is similar (approximately 20–35 MB) and negligible when considering modern storage devices (e.g., 100 GB in our experiments). Therefore, our analysis focused on the other three measurements (i.e., runtime of insertion, runtime of query, and memory usage).

- **Query Index**. This method constructed a hash table for the queries and exhibited superior run time of query (23–24 s for 60 queries, or about 0.5 s per query, the fastest in all different scenarios regardless of the number of records per insertion). It also had relatively small memory usage (like the best solution, Index Everything, in all scenarios). For the runtime of insertion, it performed better when one record at a time was inserted, while it was comparatively slower when multiple records were inserted at a time.

- **Index Everything**. This approach indexed all possible queries ahead in a mapping table and performed extremely well when multiple records at a time were inserted (only 24–42% of the time used by the other two methods). It also used the least memory in all combination scenarios. However, this method required more insertion time when one record at a time was inserted. Also, the query time was slightly slower than that for the Query Index method.

- **Dual-Scenario Indexing**. This solution created two mapping structures to store the complete and wildcard queries and provided the shortest insertion time

when one record at a time was inserted. The runtimes of insertion for multiple records at a time were comparable to those for the Query Index method. It required more time to query and more memory usage when compared to the other two methods.

To summarize, different methods can be more suitable for different applications and scenarios. To achieve a fast insertion time, Index Everything (inserting multiple records at a time) and the Dual-Scenario Indexing (inserting one record at a time) would be more appropriate. To optimize query time, Query Index would be the best method. To preserve memory usage, both Index Everything and Query Index approaches could be considered.

3.2 Optimizing Logging and Querying for Blockchain-based Cross-site Genomic Dataset Access Audit

Genomic data have been collected by different institutions and companies and need to be shared for broader use. In a cross-site genomic data sharing system, a secure and transparent access control audit module plays an essential role in ensuring the accountability. A centralized access log audit system is vulnerable to the single point of attack and also lack transparency since the log could be tampered by a malicious system administrator or internal adversaries. Several studies have proposed blockchain-based access audit to solve this problem but without considering the efficiency of the audit queries. In this section, we propose a blockchain-based log system which can provide a light-weight and widely compatible module for existing blockchain platforms.

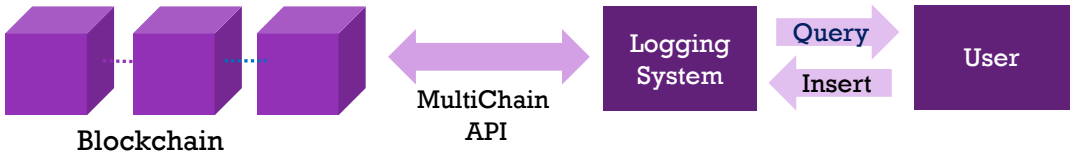
3.2.1 Cross-site Genomic Data Sharing System

With the rapid development of biomedical and computational technologies, a large amount of genomic data sets have been collected and analyzed in national and international projects such as Human Genome Project [28] , the HapMap project [29] and the Genotype-Tissue Expression (GTEx) project [46], which yielded invaluable research data and extended the boundary of human knowledge. Thanks to the advance of computer technology, the cost of genomic testing is dropping exponentially. Nowadays, the testing price ranges from under $100 to more than $2,000, depending on the nature and complexity of the test [83]. One can test her gene easily and cheaply by using services from DNA-testing companies such as Ancestry and 23andMe. Given the above, genomic data sets have been scattered around the world in different institutions and companies. On the other hand, the potential business value of genomic data and privacy concerns [51, 38, 62] hinder the sharing of cross-sites genomic data. Notably, the General Data Protection Regulation (GDPR) restricts the exchange of personal data. Under GDPR, such sensitive data only could be accessed after obtaining the consent of data subjects (i.e., the one who owns the data) and providing accountability audit. This requires that any cross-site genomic data sharing system should be equipped with a secure and transparent access control module.

3.2.2 Methods

We design a blockchain-based log system that is time/space efficient to store and retrieve genomic dataset access audit trail. Our method only leverages the Blockchain mechanism and is not limited to any specific Blockchain implementation, such as Bitcoin[60], Ethereum[84]. We introduce an on-chain indexing data structure which can be easily adapted to any blockchains that use a key-value database as their local storage. In our development, we use Multichain version 1.0.4 as an interface between Bitcoin Blockchain and our insertion and query method. Multichain is a Bitcoin

Figure 3.9: Overview of the logging system.



Blockchain fork. It conveniently provides a feature, data stream, to allow us to use Bitcoin Blockchain as an append-only key-value database.

System Overview

In Figure 3.9, we illustrate the overview of the logging system, which is built on top of Multichain APIs. The core task is to design space and time efficient methods for insertion and queries. There are three types of primitive queries: point query, AND query, and range query. There are seven fields in the given genomic dataset: Timestamp, Node, ID, Ref − ID, User, Activity, Resource as shown in Table 3.3.

Timestamp	Node	ID	Ref-ID	User	Activity	Resource
152202801	1	1	1	1	REQ_RESOURCE	MOD_FlyBase
152208352	1	2	1	1	VIEW_RESOURCE	MOD_FlyBase
152216966	1	3	3	6	FILE_ACCESS	GTEx
152237149	1	9	9	10	REQ_RESOURCE	MOD_SGD

Table 3.3: The Sample Logs.

For point query, the user can query on any field. For AND query, the user can query on any combination of fields. For range query, the user can query only on timestamp field with a start and end timestamp. See Table 3.3 & 3.4 as a running example.

Baseline method

We first describe a naive method as a baseline. The baseline method leverages only three Multichain APIs as shown in Table 3.5.

Insertion
Insert(“152202801 1 1 1 1 REQ_RESOURCE MOD_FlyBase”)
Queries
Point_Query(Activity=”VIEW_RESOURCE”)
AND_Query(ID=”2”, Node=”1”)
Range_Query(start=1522000000000, end=1522000100000)

Table 3.4: Insertion and Queries Examples.

Multichain APIs	Description
create(stream_name)	create a streamin database
publish(stream_name, key, value)	Insert key-value pair to specific stream
liststreamkeyitems(stream_name, key)	Retrieve all items with the given key

Table 3.5: Multichain APIs Used in Our Methods.

Insertion: First, we create K streams[1], where K is the number of fields. Multichain builds K tables in its back-end key-value database. Second, we build K key-value pairs, where the key is the attribute data and the value is entire record line. Finally, we convert those K pairs into one Blockchain transaction and publish it to Blockchain. The following Figure 3.10 is an example conversion from a log record to blockchain transaction. We will use this example log record in the remaining sections. After the transaction is confirmed by Blockchain, Multichain decodes the transaction and insert each key-value pairs to its corresponding table.

Timestamp	Node	ID	Ref-ID	User	Activity	Resource
111	1	3	3	6	FILE_ACCESS	GTEx

⇓

Transaction

Timestamp		Node		ID		Ref-ID		...	Resource	
Key	Value	Key	Value	Key	Value	Key	Value		Key	Value
111	V	1	V	3	V	3	V		GTEx	V

$V =$ ”111 1 3 3 6 $FILE\_ACCESS$ $GTEx$”.

Figure 3.10: Log Record to Transaction Conversion.

Point Query: The implementation of a point query is straightforward which

[1]same as table in database terminology

simply returns a list of records as shown in Algorithm 1. In this literature, we assume the run time complexity of all Multichain API is $O(1)$. The run time complexity of Point Query is $O(1)$.

Algorithm 1: Point Query

Input: A, K //attribute and key
Output: l_r //a list of record
1 $l_r \leftarrow liststreamkeyitems(A,K)$
2 **return** l_r

AND Query: AND query, Algorithm 2, enables a user to query with multiple keys. We convert AND query to multiple point queries and intersect the result of all point queries. The run time complexity is $O(K)$, where $K = number\ of\ keys$.

Algorithm 2: AND Query

Input: l_{AK} // a list of attribute and key pairs
Output: l_r //a list of record
1 $l_r \leftarrow point\_query(l_{AK}[0]_A, l_{AK}[0]_K)$
2 **foreach** $(A, K) \in l_{AK}$ **do**
3 $\quad l_r \leftarrow l_r \cap$ *point_query(A,K)*
4 **end**
5 **return** l_r

Timestamp Range Query: Given a start timestamp and an end timestamp, Timestamp Range Query, Algorithm 3, returns records whose timestamp is in this range. We convert Timestamp Range Query into R point queries, where R is the range of timestamp. The run time complexity is $O(R)$, where $R = range\ of\ timestamp$.

Algorithm 3: Timestamp Range Query

Input: t_s, t_e // start timestamp and end timestamp
Output: l_r //a list of record
1 $l_r \leftarrow \{\}$
2 **for** $t = t_s$ ***to*** t_e **do**
3 $\quad l_r \leftarrow l_r \cup point\_query$("Timestamp",t)
4 **end**
5 **return** l_r

Enhanced method

After testing the baseline solution, which will be discussed in the result section, we found that the retrieve speed heavily depends on the number of API calls. Therefore, the fewer API calls we use, the faster retrieve speed we get. More specifically, we found three non-optimal issues:

- The entire record is duplicated K times where K is the number of fields, which is insufficient in terms of storage overheads.
- Since we need to query all results and intersect them in local memory, AND query takes significant amount of memory when the number of AND operations increases.
- If the length of a given range query is n (typically, n is ranging from 10^6 to 10^8), the baseline method naively translate the range query into n point queries and concatenate the results.

The blockchain-based auditing system is an append-only structure, so a data structure that keeps the minimum amount of information while maintaining the efficiency is essential. The percentage of read(query) operations in the real-world auditing system is low [69], therefore we trade retrieval speed for storage cost. We redesign the key-value pairs in the blockchain transaction, modified the query algorithm accordingly and built a selectivity list based on data distribution. Most of all, we design a hierarchical timestamp structure which significantly reduces the number of queries(APIs) needed for the range query.

Insertion: To address these problems, we redesign the key-value pairs. The key part remained the same (attribute data), but we removed the entire entry from the value part. As a result, we removed all duplicated values in the baseline method as shown in Figure 3.11.

Transaction										
Timestamp		Node		ID		Ref-ID			Resource	
Key	Value	Key	Value	Key	Value	Key	Value	...	Key	Value
111	∅	1	∅	3	∅	3	∅		GTEx	∅

All duplicated Vs are removed.

Figure 3.11: Transaction with empty values.

Point Query: Since we now have an empty value in the key-value database, we cannot use the key to get original record directly. We now take advantage of Blockchain transaction ID which is included in the returning JSON file of *liststreamkeyitems* API. First, we get a list of *TXID* (transaction ID) with the given key. Second, we use another Multichain API, *getrawtransaction*, to get the matching transactions. Finally, we rebuild the original record from the transaction where all attribute data are included. It is worth mentioning that the point query now requires 1 + T times API calls to retrieve the records where T is the size of the *TXID* list. If Multichain nodes can perform the work from line 3 to line 6 in Algorithm 4, users can point query with just one API call. The run time complexity of our point query is $O(T)$, where T is the size of the *TXIDs* list.

Algorithm 4: Point Query with additional step

Input: A, K //attribute and key
Output: l_r //a list of record
1 $[TXIDs] \leftarrow liststreamkeyitems(A,K)$
2 $l_r \leftarrow []$
3 **foreach** $txid \in [TXIDs]$ **do**
4 $\quad T \leftarrow getrawtransaction\ (txid)$
5 $\quad R \leftarrow rebuild(T)$
6 $\quad append(l_r, R)$
7 **end**
8 **return** l_r

AND Query: In order to reduce the retrieval cost, we build a selectivity list for attributes based on the example test data in Algorithm 5. A selectivity list is based

on the rank of result size of each attribute. The attribute which has the smallest query result size is the most selective. For the enhanced AND query, we call point query only one time for the most selective key then filter the result in the memory. Since we only query once from Blockchain, the total memory usage is bounded by the largest query result. The run time complexity is $O(1)$.

Algorithm 5: AND Query with selectivity list

Input: l_{AK}, l_S // a list of attribute and key pairs and a selectivity list
Output: l_r //a list of record
1 $SK \leftarrow$ *findMostSelectiveKey*(l_{AK}, l_s)
2 $l_r \leftarrow$ *point_query*(SK_A, SK_K)
3 **foreach** $(A, K) \in l_{AK}$ **do**
4 $\quad l_r \leftarrow$ *filter*(l_r, A, K)
5 **end**

Timestamp Range Query: Since Blockchain is an immutable structure, the common indexing techniques, such as B-tree and R-tree, which require adjusting/balancing the entire data structure according to the data distribution, won't work. We introduce a hierarchical timestamp structure, which is an incremental data structure and matches the append-only characteristics of the blockchain system. Our design significantly reduces the number of queries(APIs) needed for a single range query.

The hierarchical timestamp structure consists of multiple levels. See Table 3.6 as an example. The range in the high level divides into multiple smaller range in the lower level. We denote each range part as LevelNumber:Starting Timestamp. A timestamp is recorded in the corresponding part at all levels. In our running example, a timestamp *111* will be recorded in L0:100, L1:110, and L2:111 in Table 3.6.

L0	[100,200)								
L1	[100,110)			[110,120)			[120,..)		
L2	100	...	109	110	...	119	120	...	...

Table 3.6: Simple Hierarchical Timestamp Structure

To build this structure, we need to slightly modify the insertion method by adding

L streams where L is the number of levels, and we need to add L key-value pairs to Blockchain transaction as well. See Figure 3.12 as an example.

Transaction										
Timestamp		...	Resource		L0		L1		L3	
Key	Value		Key	Value	Key	Value	Key	Value	Key	Value
111	∅		GTEx	∅	100	∅	110	∅	111	∅

Figure 3.12: Transaction with Hierarchical Timestamp Structure.

In our enhanced range query method, Algorithm 6, we recursively find the largest range in the hierarchical timestamp structure and use multiple point queries to retrieve the result.

Algorithm 6: Timestamp Range Query with hierarchical timestamp structure

Input: t_s, t_e // start timestamp and end timestamp
Output: l_r //a list of record
1 $l_r \leftarrow list$
2 $l, r \leftarrow findLargestRange(t_s, t_e)$
3 **while** $r \neq None$ **do**
4 $\quad append(l_r, point\_query(l, r))$
5 $\quad l, r \leftarrow findLargestRange(t_s, t_e)$
6 **end**
7 **return** l_r

In the following example, we show the number of queries(APIs) needed for our baseline range query and enhanced range query.

Range query from timestamp 109 to timestamp 120.

Baseline Method: $q('T', 109) \cup q('T', 110) \cup ... \cup q('T', 120) \rightarrow 11$ queries

Enhanced Method: $q('L2', 109) \cup q('L1', 110) \cup q('L2', 120) \rightarrow 3$ queries

We reduce the number of queries needed for range query from $R_{T_e - T_s}$ to $\sum_{i=0}^{L} \frac{R_i}{r_{L_i}}$, where $R_{i+1} = R_i \mod r_{L_i}$, $R_0 = R_{T_e - T_s}$ and r_{L_i} is the elemental range at level L_i.

The run time complexity of the enhanced range query is $O(\sum_{i=0}^{L} \frac{R_i}{r_{L_i}})$.

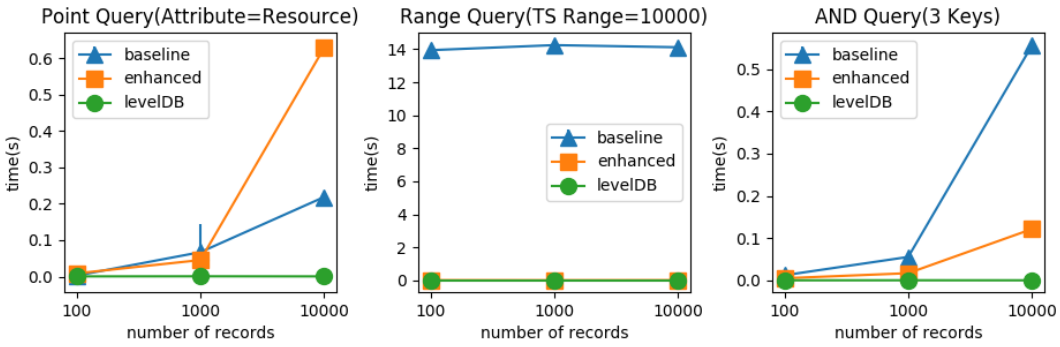


Figure 3.13: Scalability Test: Queries.

3.2.3 Results

We used Python3 as our main programming language to develop our solution, Savior [7] to interact with Multichain API and Docker [3] to simulate 4 Blockchain nodes. Additionally, we created some bash scripts to automatically setup Blockchain nodes and Multichain environment. We also wrote a benchmark program to compare our baseline method and enhanced method. Our code is available online [15]. The specifications of our testing machine are as follows: 6 cores CPU(i7 8700k), 32 GB of RAM and 6TB of HDD with Ubuntu 16.04 as the operating system.

We used the sample testing data supplied by the competition organizer to benchmark our implementation. The sample testing data consists of 4 files, one per node. Each file has 10^5 entries of log records which has 7 fields (Timestamp, Node, ID, Ref − ID, User, Activity, Resource). To illustrate, we provide a few sample data in Table 3.3.

To find the optimal number of levels and the step multiplier of two adjacent levels for the hierarchical timestamp structure (Table 3.6), we test all reasonable parameter combinations by brute-force. For the given sample data, the optimal parameter for the number of levels is 3 and the step multiplier of two adjacent levels is 100.

In our benchmark experiment, we show the scalability of our two methods alongside LevelDB[6] as a reference. Many blockchain systems[19][1][4] use LevelDB as a back-

end database to store the raw transaction data. It is worth mentioning that those systems only index the raw transactions, not the actual content inside the transactions. Database system and Blockchain do not share the same design goal: the former is usually administered by a centralized entity, and the latter intents to work in a trustless environment. Nevertheless, this comparison offers useful insights of Blockchain based log system which trades speed for data integrity. We simulate the enhanced insertion, the enhanced point query, and the enhanced AND query behavior in LevelDB. For range query, we use LevelDB native method so we can properly examine our hierarchical timestamp structure. In all tests, we run 10 rounds for each methods with respect to varying the number of records. We calculate the average and the standard deviation from the results. We notice that the standard deviation is extremely small which shows the little trace in all figures expect Figure 3.13(Point Query). This is due to the identical environment and the setup of our simulated blockchain nodes.

Scalability Test: Queries

Figure 3.13 shows query time with respect to the varying number of records for point query, range query, AND query. For the point query test, the response time is determined by the result size. As the number of records increases, the result size increases and the response time increases. The response time of the enhanced method is worse than the baseline method because of the addition API calls which we introduced in the enhanced point query. For the range query test, the performance is constant since the result size of certain time range is constant. It is worth mentioning that our enhanced range query method have very close performance comparing to the native LevelDB range query method. For AND query, since it consists of point query, the response time increases with the increasing number of records. It is worth mentioning that the selectivity list design in our enhanced AND query method offsets the drawback of the enhanced point query method when the number of keys is larger

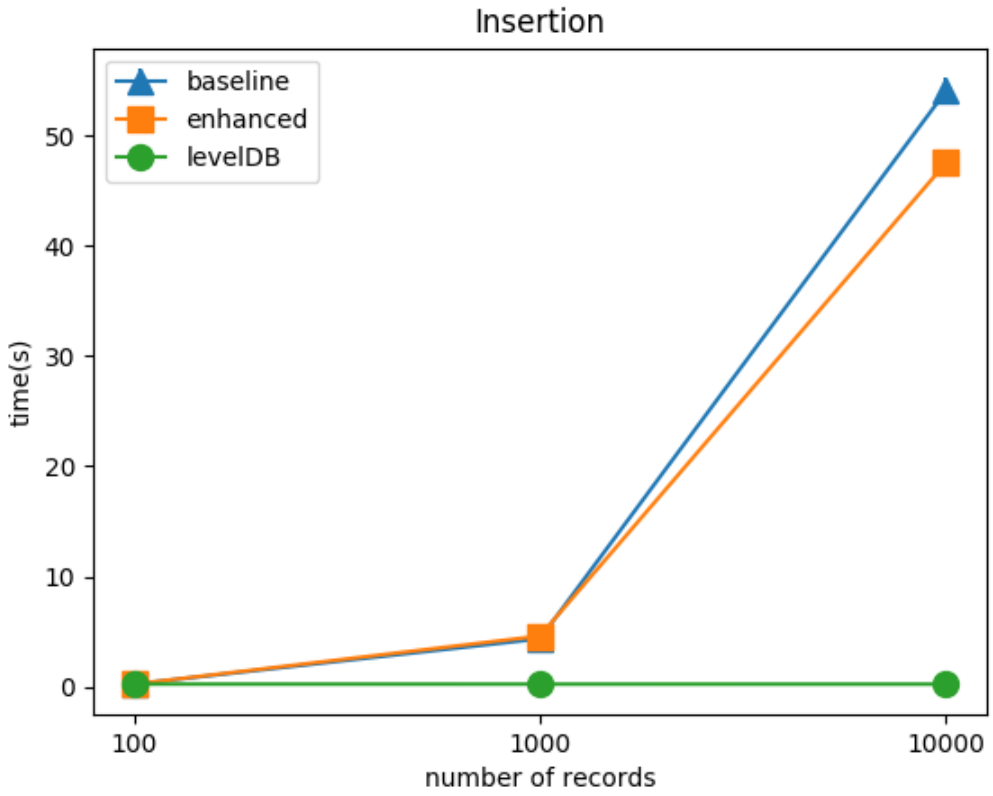


Figure 3.14: Scalability Test: Insertion.

than 2.

Scalability Test: Insertion

Figure 3.14 shows the completion time of insertion methods with respect to varying the number of records. The insertion time is depended on the transaction size. The insertion times of the two methods are approximately the same. The enhanced method needs more key-value pairs to support hierarchical timestamp indexing structure. However, the empty values in key-value pairs offset this transaction size increment.

Scalability Test: Storage

Figure 3.15 shows the total blockchain size in bytes with respect to varying the number of records. The blockchain size information is collected by calling Multichain

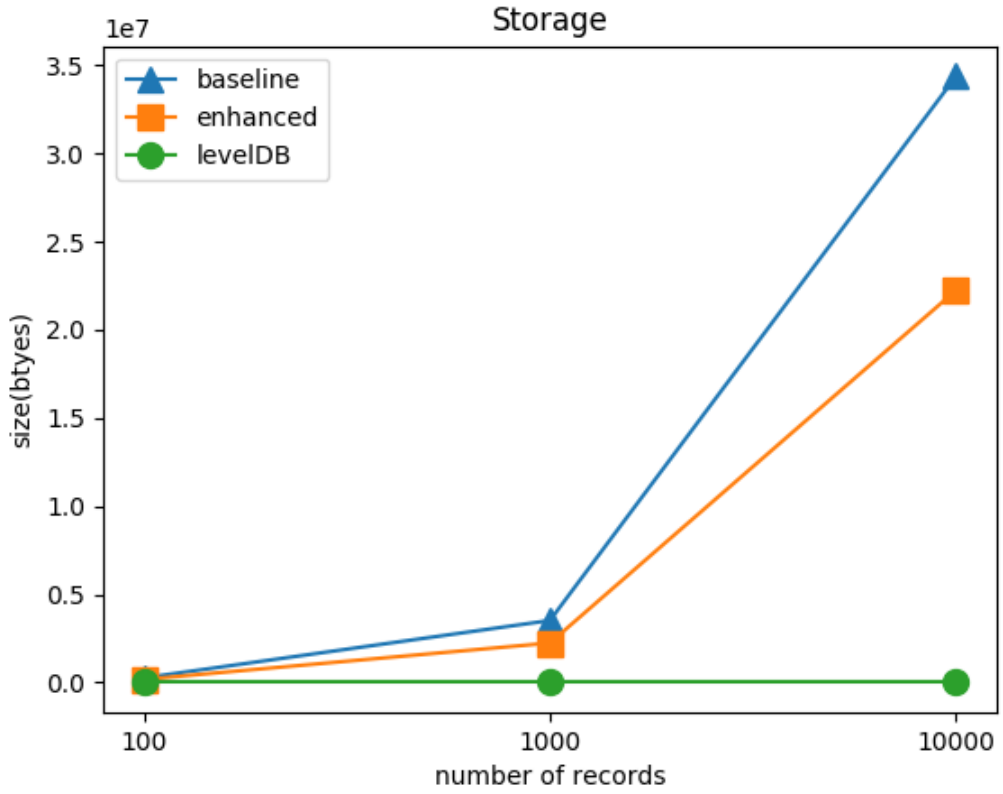


Figure 3.15: Scalability Test: Storage.

API. Since Blockchain and LevelDB measure their size in different ways, we exclude LevelDB in this test. The figure suggests that the enhanced method uses less storage than the baseline method. The duplication removal from the blockchain transaction in the enhanced method works as designed.

Detailed Comparison

In this section, we show a detailed performance difference of 3 query types in the baseline method, enhanced method, and LevelDB. We use the fixed 1000 records in the remaining tests.

Point Query: Figure 3.16 shows the query response time for different attributes. The enhanced method performance is worse than the baseline method, because of the additional API calls in the enhanced method. The rank in the result also matches the

rank in selectivity list which indicates the return record size. The return record size of *Activity* is the largest among the attributes. In other words, *Activity* has the lowest selective and need more API calls to get the result than other attributes, so it has the worst performance difference.

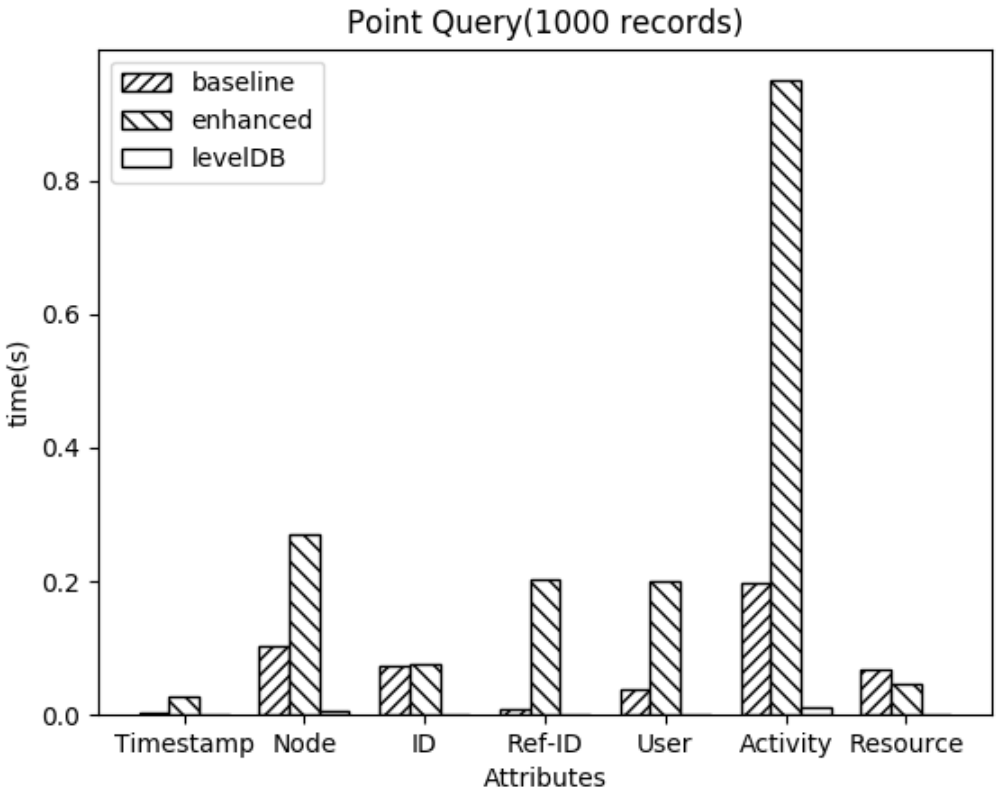


Figure 3.16: Point Query.

Range Query: Figure 3.17 shows the query response time with respect to varying the time range. The enhanced method is at least one order of magnitude better than the baseline method. It proves that our hierarchical timestamp structure can batch a large number of queries into a small (almost constant) number of queries. Hence, the enhanced method achieves almost constant time performance as LevelDB native range query method.

AND Query: Figure 3.18 shows the query time with respect to varying the number of keys. We test all combinations of keys. For example, for 2 keys test, we

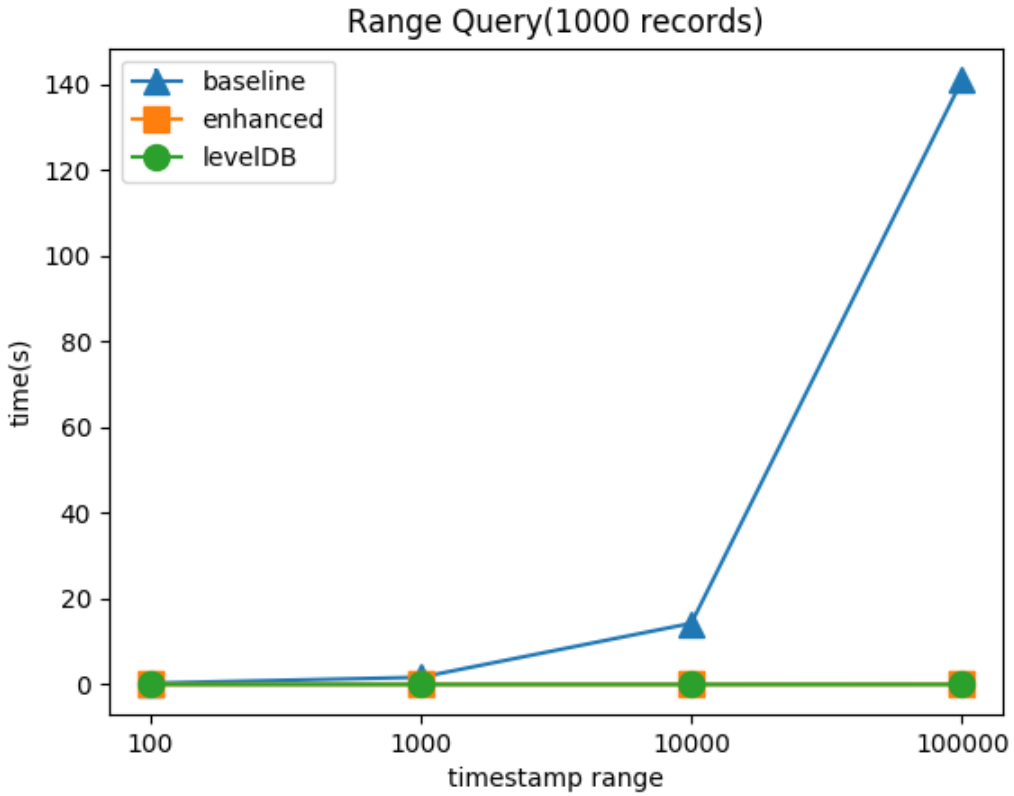


Figure 3.17: Range Query.

test all 21 combinations(7 choose 2) and average the result. It is much easier to find a more selective key when the number of keys is increasing. This is the reason why the enhanced method has a downward slope. When there are only 2 keys, the enhanced method has high possibility to find a low selective key. As a result, when AND query takes a low selective key, it requires a long response time.

3.3 Conclusion

In this study, we have presented innovative solutions for blockchain-based logging and querying of genomic dataset audit trails, and demonstrated the feasibility of sharing gene-drug interaction data using smart contracts on blockchain technology.

Our approach to the logging and querying of genomic dataset audit trails prioritizes

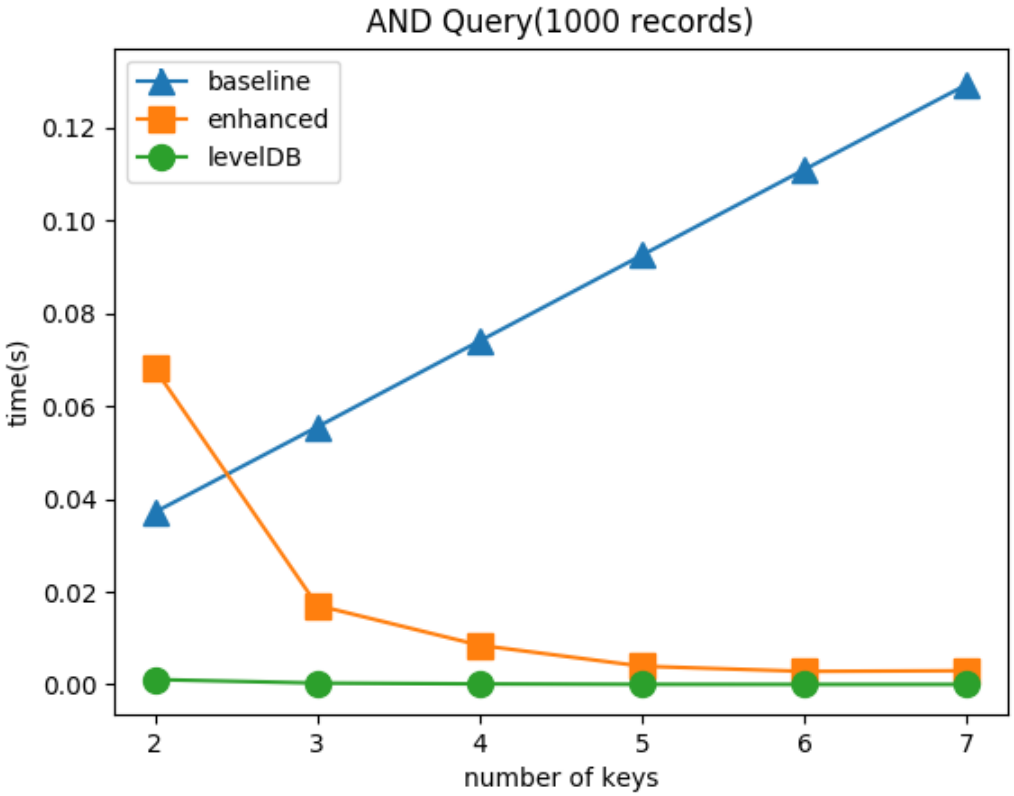


Figure 3.18: AND Query.

storage space over retrieval and insertion speed, reflecting the real-world characteristics of log systems where write operations significantly outnumber read operations. By implementing a hierarchical timestamp structure design, we have reduced storage costs by 25% and increased range query speed by at least an order of magnitude. Importantly, this design is blockchain implementation-independent and can be adapted to any blockchain with the aid of an intermediary.

In terms of sharing gene-drug interaction data, our system can store 4,000 gene-drug evaluation results from four sites within one minute and query all these pairs within 0.5 seconds. These results serve as benchmarks for future blockchain-based healthcare, genomic, and biomedical applications.

In conclusion, this study contributes to the growing body of knowledge on the adoption of blockchain technology for genomics and healthcare applications. Our

novel blockchain-based methods for sharing patient outcomes of gene-drug interactions could promote data sharing, thereby enabling personalized medicine. The results of this study can support future blockchain-based healthcare, genomic, and biomedical applications, and serve as a benchmark for future research in this field.

Chapter 4

Deterring Illicit Cryptocurrency Use

In the cryptocurrency ecosystem, tracking the flow of money in the event of a crime, fraud, or illicit transaction can be challenging due to the irreversible and pseudonymous nature of transactions. Our virtual taint system, inspired by the dye packs used by banks to disincentivize theft, can be used to overcome this challenge. In TaintCoin, we mark cryptocurrency transactions as "tainted" if they are known to have been involved in a crime, fraud, or other illicit activity and enforce collection of tainted coins at participating exchanges. Because tainted virtual coins are undesirable, much like physically dyed cash, monetizing or laundering the proceeds from criminal behavior becomes harder. As a second-order effect, the taint system deters potential fraudsters and criminals, helping to prevent rampant illicit activity that is today facilitated by cryptocurrency. Our vision is for the taint system to revitalize integrity within the cryptocurrency ecosystem by explicitly penalizing bad behavior. We show how our system is resilient to attacks, scalable to modern blockchains, and address deployment questions, such as how to conclude that a transaction should be tainted, explore different diffusion models for taint, present a fast-tracking algorithm, and discuss and

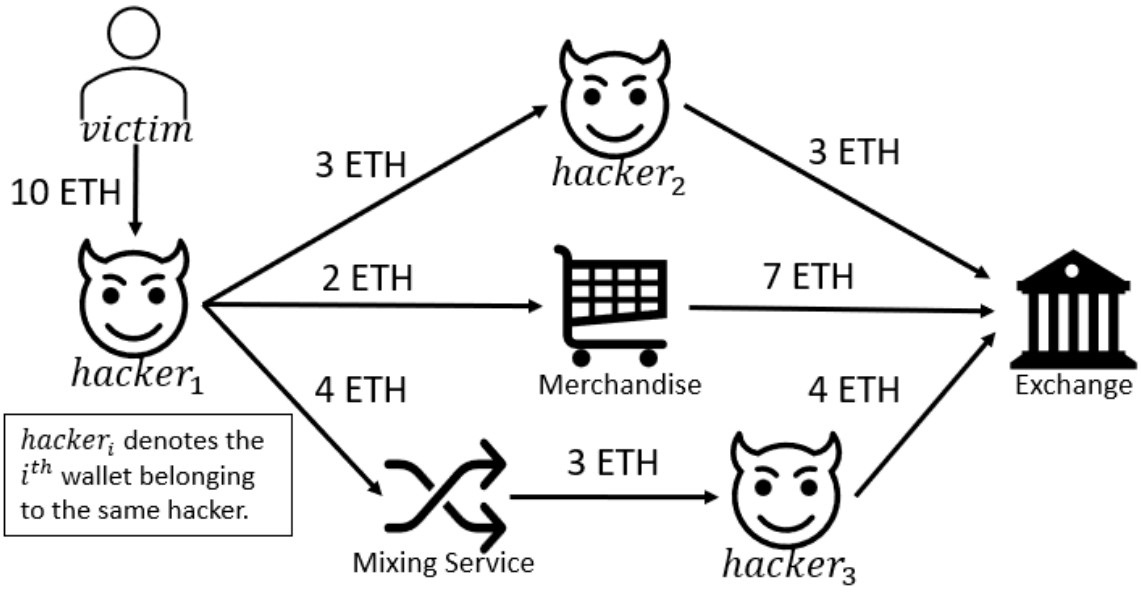


Figure 4.1: An Example of Crypto Money Laundering Layering

A ransomware victim pays 10 ETH as a ransom to a hacker's wallet, denoted by $hacker_1$. The hacker subsequently engages in a series of intricate crypto transactions to conceal the origin of these funds. Specifically, the hacker transfers 3 ETH to a disposable wallet $hacker_2$, and spends 2 ETH to buy goods from merchants who accept cryptocurrency directly. The hacker also deposits 4 ETH into crypto mixing services, then transfers the funds to a new wallet $hacker_3$. Ultimately, the cryptocurrency held in $hacker_2$, $hacker_3$, and the merchant's wallets is sent to exchanges for conversion into fiat currency. This complex web of transactions illustrates the challenges of tracking illicit activities within the blockchain ecosystem.

analyze possible implications from rolling out TaintCoin.

4.1 Cryptocurrency Money Laundering

Cryptocurrency is one of the most prominent applications of blockchain technology. Cryptocurrencies such as Bitcoin [60], Ethereum [84], and many others rely on blockchain to record transactions. However, cryptocurrencies can make it easier for fraudsters to obscure the source of criminal proceeds and are increasingly becoming the preferred currency of cybercriminals, from purchasing illicit goods using cryptocurrency as a payment method to ransomware attacks where payments by cryptocurrency are demanded. This trend is more prevalent because cryptocurrency offers a combi-

nation of anonymity, ease of use, and the ability to circumvent international borders and regulations, in essence, to launder the ill-gotten proceeds. When it comes to cryptocurrency, advanced fraudsters or money launderers have a plethora of techniques and strategies that they can use to conceal their activities. One of the most common methods is to use mixing services, which essentially blend their funds with those of other users in an attempt to make it difficult to trace the origin of the funds, disassociating them from criminal activities to cash out safely using a cryptocurrency exchange. As illustrated in Figure 4.1, it is straightforward for a legitimate exchange to trace funds from $hacker_2$ to $hacker_1$, as all transactions on the blockchain are public. However, linking the funds in $hacker_3$ and the partial funds used in the Merchandise to their origin in $hacker_1$ is much more complex. This complexity underscores the challenges of identifying and tracking the flow of illicit funds within the blockchain.

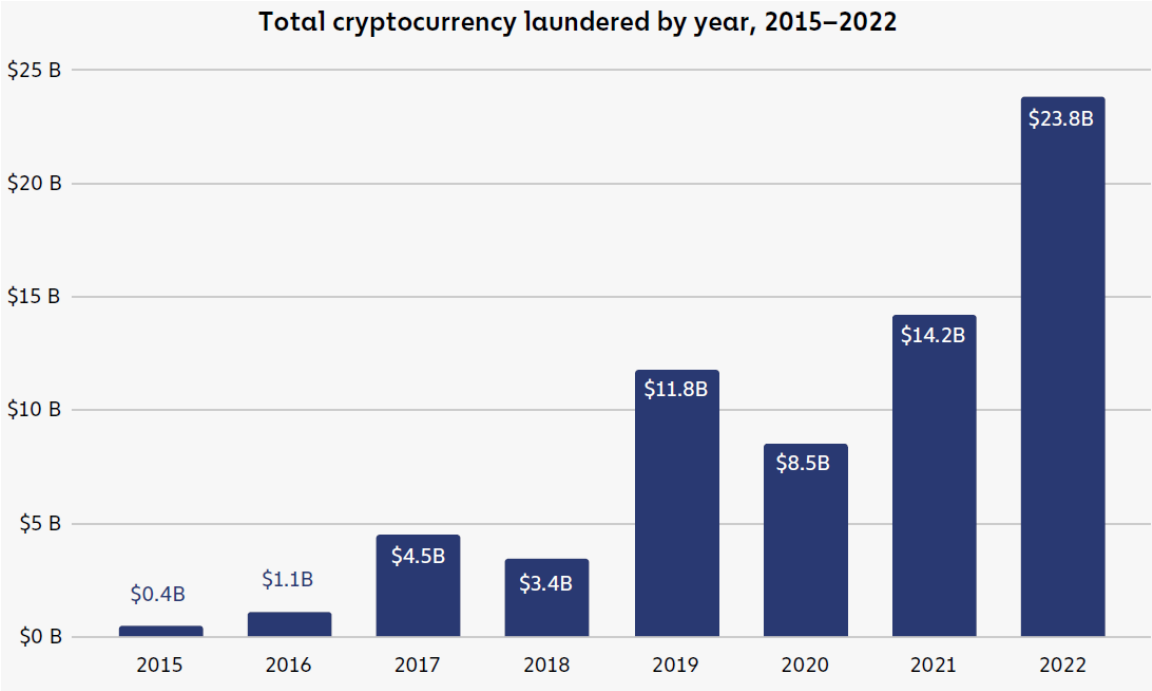


Figure 4.2: Total cryptocurrency Laundered by year, 2015-2022

Fig. 4.2 [16] shows that illicit addresses sent nearly $23.8 billion worth of cryptocurrency in 2022, a 68.0% increase over 2021. As is usually the case, mainstream

centralized exchanges were the biggest recipient of illicit cryptocurrency, taking in just under half of all funds sent from illicit addresses. That's notable not just because those exchanges generally have compliance measures in place to report this activity and take action against the users in question, but also because these exchanges are fiat off-ramps, where the illicit cryptocurrency can be converted into cash.

4.2 Design

Our taint system is a tool that can help in this effort by marking the tainted cryptocurrency and tracing its flow through the blockchain. It's an innovative solution that provides transparency and accountability in the cryptocurrency ecosystem, helping to prevent illicit activity and protect its users from fraudulent behavior.

TaintCoin is inspired by the concept of a dye pack, which will be familiar to anyone who has seen a bank heist movie. A dye pack is a small explosive device that is placed in a bag of money during a bank robbery. When the robber leaves the bank, the dye pack explodes, covering the money in ink and rendering it unusable. Similarly, in the cryptocurrency ecosystem, we can employ a concept known as a "taint system" or "digital dye pack" to track the flow of funds in the event of a crime, fraud, or illicit transaction.

By marking a particular amount of cryptocurrency as "tainted" if it has been involved in a crime, fraud, or illicit transaction, the taint system allows us to trace where the tainted money goes and identify the parties involved in the wrongdoing. This is achieved through the use of thoughtfully engineered algorithms and blockchain analysis techniques, which can monitor the movement of tainted funds across the network. Moreover, the taint system can also act as a deterrent to potential fraudsters and criminals. Knowing that their illicit transactions can be traced and potentially linked back to them, wrongdoers may be less likely to engage in fraudulent activity

within the cryptocurrency ecosystem. This is analogous to how traditional dye packs act as a deterrent by making stolen cash easily identifiable and therefore harder to use.

Furthermore, the taint system can be integrated with other security measures and regulatory compliance tools. For instance, cryptocurrency exchanges and wallet providers can implement the taint system to monitor transactions and ensure that they are not facilitating the movement of illicit funds. This not only helps in maintaining the integrity of their platforms, but also aids in adhering to anti-money laundering (AML) and know-your-customer (KYC) regulations. In addition to tracking and deterring illicit activities, the taint system can also be used for forensic analysis after a security breach or fraudulent activity. By analyzing the flow of tainted funds, investigators can gain insight into the modus operandi of criminals, which can be invaluable in legal proceedings and in developing strategies to prevent future incidents.

4.2.1 Architecture Overview

The proposed taint system, depicted in Fig. 4.3, is composed of four integral components, each playing a crucial role in ensuring the effective tracking and marking of the cryptocurrency involved in illicit activities. These components are:

- **Database for Cryptocurrency Transactions**: This component is essentially a repository that receives and stores cryptocurrency transactions. It is responsible for collecting data on all transactions occurring within the cryptocurrency network. The database is continuously updated in real-time to ensure that the most recent transactions are available for analysis. This is crucial for the taint system to accurately track the flow of funds.

- **Customizable Taint Algorithm**: The taint algorithm is at the heart of the taint system. It is responsible for analyzing transactions stored in the

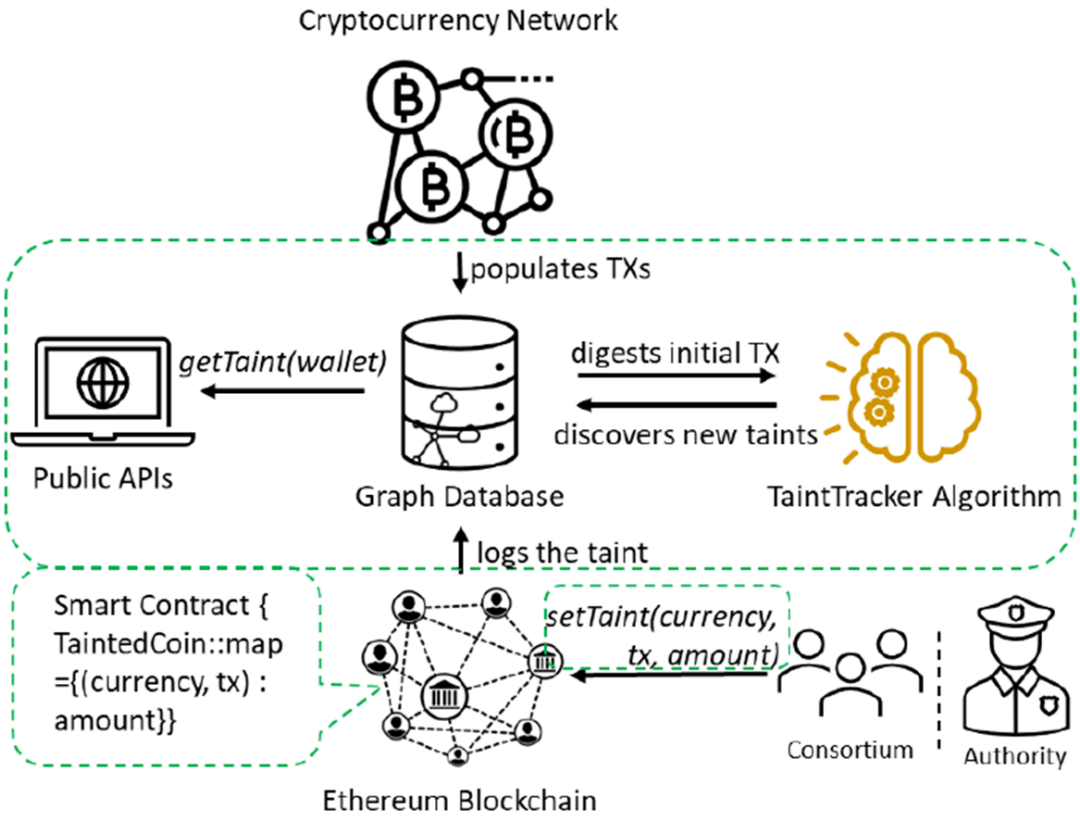


Figure 4.3: Taint System Architecture

database and identifying those involved in illicit activities. What makes this algorithm particularly powerful is its customizable nature. Depending on the specific requirements and criteria, the algorithm can be adjusted and fine-tuned to accurately mark tainted cryptocurrency. This flexibility allows the taint system to adapt to different types of illicit activities and evolving tactics used by wrongdoers.

- **Public Interface for Accessing Tainted Results**: To ensure transparency and accessibility, the taint system includes a public interface through which users and entities can access the results of the taint analysis. This interface is user-friendly and can be used by individuals, cryptocurrency exchanges, regulatory bodies, and law enforcement agencies to check whether a particular amount

of cryptocurrency is tainted. This information can be invaluable for making informed decisions regarding the handling of cryptocurrency.

- **Blockchain-based Registry for Initial Address and Taint Amount Setting**: This component is responsible for initializing the taint tracking process. It involves setting the initial address from which the tainted cryptocurrency originated and specifying the amount of cryptocurrency that is considered tainted. By leveraging blockchain technology, this registry ensures that the initial settings are immutable and transparent. This is crucial to establish a trustworthy starting point for the tracking process and to ensure the integrity of the taint system.

Together, these components work in tandem to create a robust and effective taint system capable of tracking the flow of illicit funds within the cryptocurrency ecosystem. To break this down, once a transaction is recorded on the blockchain, it cannot be altered or deleted. Similarly, the blockchain-based registry used to set the initial address and taint amount is also immutable. Once these initial settings are recorded on the blockchain, they are permanent and cannot be changed, providing a secure and tamper-proof starting point for taint tracking.

Additionally, the blockchain's transparent nature ensures that all transactions, including the initial address and taint amount settings, are visible to anyone with blockchain access. With the taint algorithm being widely known and the data (cryptocurrency transactions, initial address, and taint amount) being immutable and transparent, anyone can independently apply the taint algorithm to the data and reproduce the tainted result. Thus, any tampering would be noticeable as the independently reproduced result would not match the system-provided taint result. In essence, the combination of immutability, transparency, and reproducibility ensures the taint system's tamper-evident nature. Any alteration or manipulation attempts would be easily detectable, maintaining the integrity and reliability of the system.

The taint system operates on a hybrid structure that blends centralized and de-

centralized elements, facilitating a collaborative approach towards marking tainted cryptocurrencies. The central registry is managed by a consortium, authority, or blockchain core developers, and is responsible for officially setting taints based on substantial evidence. Simultaneously, individuals and organizations can set up secondary registries to contribute their own analysis and taint settings. The primary system is structured to integrate data from these secondary registries, thereby creating a comprehensive dataset to track tainted cryptocurrencies. This system leverages centralized oversight's rigor and decentralized contributions' diversity, integrating validation mechanisms to verify secondary registries' data credibility and protect against misuse. As such, this multifaceted approach fortifies the taint system, making it a resilient tool for monitoring the flow of tainted cryptocurrency.

4.2.2 Deployment

A taint tracking system, vital to ensure the transparency and security of digital transactions, can be deployed by various stakeholders within the digital asset landscape. Cryptocurrency exchanges stand to gain significantly from this deployment. The adoption of taint tracking mechanisms can identify potentially risky transactions, thereby strengthening their security and credibility. Furthermore, these systems aid in meeting the stringent regulatory requirements associated with anti-money laundering (AML) and countering the financing of terrorism (CFT). Regulatory authorities would also benefit from this technology. It provides a clear view of transactional flows and helps in spotting illicit activities or contraventions of regulatory norms. This essentially empowers regulatory bodies with a formidable tool for monitoring the health and integrity of the digital asset ecosystem. In the broader financial sector, institutions like banks that handle cryptocurrency transactions could harness the power of taint tracking to safeguard against fraudulent activities. This technology could bolster risk management strategies and provide essential support for compliance with the AML

and CFT regulations. Businesses that accept digital assets as a form of payment should consider taint tracking deployment as well. This system can shield them from the reputational and legal risks associated with receiving "tainted" coins tied to illicit activities. For agencies involved in law enforcement and criminal investigation, taint tracking can serve as a powerful tool in their arsenal. It allows for the tracing of illicit digital transactions, helping to prevent and investigate cryptocurrency-related crimes.

4.2.3 Algorithms

Taint analysis has been rigorously investigated within the Bitcoin ecosystem, wherein all Unspent Transaction Outputs (UTXOs) are inherently non-fungible [78, 18]. Motivated by these foundational studies, we endeavor to transpose the taint analysis paradigm onto the Ethereum platform. Ethereum's model diverges significantly from Bitcoin's, operating on an account-based system where tokens are intrinsically fungible. Analogous to the operations of traditional banking systems, Ethereum treats all ether as homogenous units; thus, when ether is aggregated within an account, it becomes a challenge to retain or trace the historical lineage of individual units. This homogeneity significantly complicates the discernment necessary for effective taint analysis. Given the fungible nature of Ethereum, our investigation addresses the novel complexities that arise in tracking and analyzing the movement of funds. To our knowledge, our work is the first study exploring taint analysis in the Ethereum landscape.

We present the pseudocode and logic for three taint propagation algorithms: Poison, Haircut, and FIFO (First-In, First-Out). These algorithms play a crucial role in tracking and propagating taint through cryptocurrency transactions. The poison algorithm assigns a tainted status to all output addresses when any input address is marked as tainted. It employs a straightforward approach, propagating taint uniformly across subsequent transactions. The Haircut algorithm calculates the proportion of taint based on the tainted inputs and distributes this proportion among

the output addresses in proportion to their input values. This algorithm introduces a more intricate method of tainting propagation. The FIFO algorithm, on the other hand, focuses on the order of address propagation. It adds the output addresses to the taint list in the same order they appear, considering the remaining taint amount after each propagation. By outlining the pseudocode for each algorithm and providing a summary of their logic, we gain a comprehensive understanding of the distinct methods employed in propagating taint and tracking tainted addresses within the blockchain.

Poison Method

Algorithm 7: Poison Taint Propagation Algorithm

Input : Blockchain transactions (TxList), Initial Tainted addresses (TaintList)
Output : Final Tainted addresses (TaintList)

```
1 Procedure POISON_TAINT_PROPAGATION(TxList, TaintList)
2     for each transaction in TxList do
3         if the transaction's input address is in TaintList then
4             Add the transaction's output addresses to TaintList;
5         end
6     end
7     return TaintList;
```

The Poison algorithm's core principle is succinctly demonstrated in Algo. 7. The concept is relatively straightforward: If any input of a transaction is identified as tainted, it then leads to all associated outputs being classified as tainted as well. This process involves cycling through every transaction and scrutinizing whether any of the input addresses coincide with the addresses cataloged in the TaintList. Upon detecting a match, all the corresponding output addresses from that transaction are incorporated into the TaintList. This practice extends across all transactions within the blockchain, facilitating an effective propagation of the taint. The Poison method's results, when applied to a model with two inputs and outputs, are visually represented

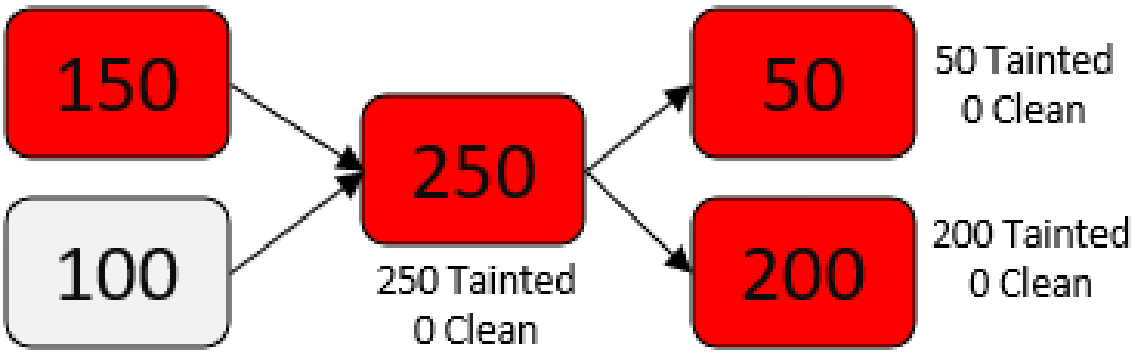


Figure 4.4: Poison Method
The white rectangles represent clean inputs, while the red rectangles represent fully tainted ones. For example, in a transaction with a 100 clean input and a 150 tainted input, both of the resulting 50 and 200 outputs will be classified as entirely tainted.

in Figure 4.4.

The justification for the Poison method is reinforced by real-world examples [17] often seen in criminal investigations led by law enforcement agencies. Specifically, when tracing the proceeds from illicit activities, they might adopt a "poison" approach where all funds originating from a criminal act are considered "tainted". For instance, if funds procured from illegal activities are co-mingled with legitimate funds within a bank account, the entire sum may be subjected to confiscation [17]. This action reflects the fundamental principle of the Poison algorithm.

Haircut Method

The Haircut algorithm, as described in Algorithm 8, implements taint propagation by determining the taint proportion, which is based on the tainted inputs involved in each transaction. This calculated proportion of taint is then proportionately allocated across the output addresses, corresponding to their respective input values. The operational procedure of the algorithm involves a thorough iteration through each transaction in the blockchain, verifying if any input address correlates with the addresses compiled in the TaintList. If a match arises, the algorithm computes the total input value

Algorithm 8: Haircut Taint Propagation Algorithm

Input : Blockchain transactions (TxList), Initial Tainted addresses (TaintList)

Output : Final Tainted addresses (TaintList)

```
1  Procedure HAIRCUT_TAINT_PROPAGATION(TxList, TaintList)
2      for each transaction in TxList do
3          if the transaction's input address is in TaintList then
4              Calculate the total input value and the proportion of taint;
5              for each output address in the transaction do
6                  Calculate the output value as the proportion of taint multiplied
                    by the output's input value;
7                  Add the output address to TaintList with the output value as
                    the new taint;
8              end
9          end
10     end
11     return TaintList;
```

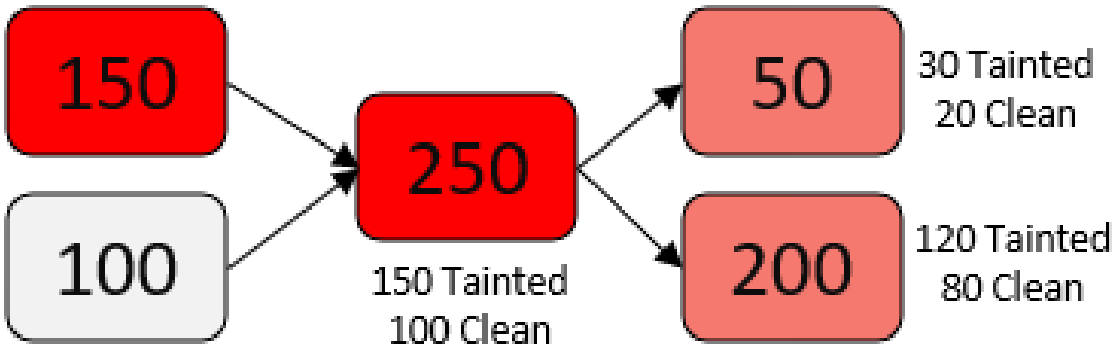


Figure 4.5: haircut Method

The white rectangles represent clean inputs, the darker red rectangles represent fully tainted ones and the light red rectangles represent partly tainted ones. Using the same example as in Fig. 4.4, instead of being tainted entirely, both outputs will receive the same proportion of the tainted coin to the total input value (3/5 proportion), which in this case would be 30 tainted for the 50 output and 120 tainted for the 200 output.

and the corresponding proportion of taint. This proportion of taint is then allocated among the output addresses in relation to their input values. The output addresses, along with their respective taint values, are subsequently appended to the TaintList. This practice extends to all transactions, thereby achieving effective taint propagation based on respective proportions. Figure 4.5 graphically presents the results of the

Haircut method applied to a model with two inputs and outputs.

The underpinning principle of the Haircut algorithm mirrors the approach employed in managing bankruptcy cases. A notable example can be found in the handling of the 2008 Lehman Brothers bankruptcy case [74]. Here, the remaining assets were proportionately divided amongst the creditors, an approach reminiscent of the Haircut algorithm's methodology.

FIFO Method

Algorithm 9: FIFO Taint Propagation Algorithm

Input : Blockchain transactions (TxList), Initial Tainted addresses and amounts (TaintList)
Output : Final Tainted addresses and amounts (TaintList)

```
1  Procedure FIFO_TAINT_PROPAGATION(TxList, TaintList)
2      for each transaction in TxList do
3          if the transaction's input address is in TaintList then
4              for each output address in the transaction do
5                  Calculate the remaining taint amount after propagation;
6                  Add the output address and the remaining taint amount to
                    TaintList;
7              end
8          end
9      end
10     return TaintList;
```

In summary, the FIFO algorithm (Algo. 9) propagates taint by calculating the remaining taint amount after propagation for each output address in a transaction. If the input address of a transaction is in the TaintList, the remaining taint amount is calculated and added to the TaintList along with the corresponding output address. This process continues for all transactions in the blockchain, resulting in the final TaintList containing the addresses and the remaining taint amounts propagated by the FIFO algorithm. Fig. 4.6 depicts the FIFO method result.

The principle of the FIFO method aligns with legal precedents in certain jurisdic-

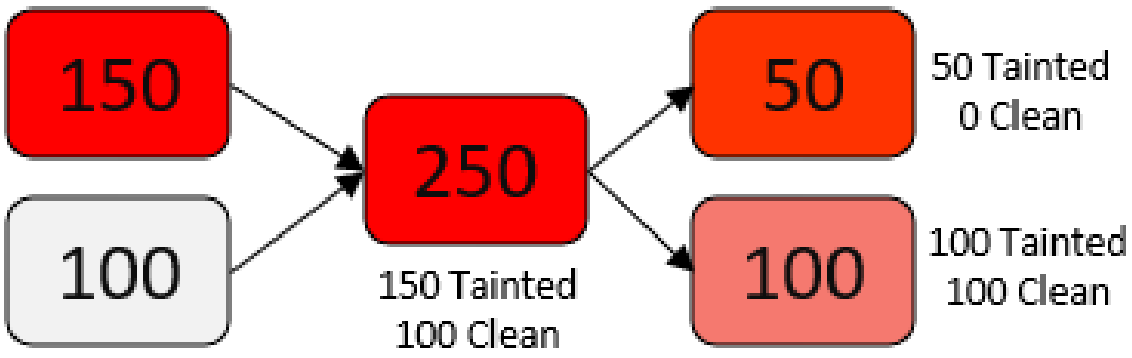


Figure 4.6: FIFO Method
The white rectangles represent clean inputs, the darker red rectangles represent fully tainted ones and the light red rectangles represent partly tainted ones. The FIFO method starts by allocating the first 150 tainted input in the first output and remaining 100 tainted in the second output. As a result, the first output are entirely tainted and the second output are partly tainted.

tions, notably in Clayton's case (Devaynes v Noble, UK, 1816) [73], often referenced in English common law. The principle of Clayton's Case, or the "first in, first out" rule, was established as a method of tracing funds in mixed bank accounts. This rule posits that withdrawals from an account are presumed to deplete the earliest deposited funds first. In the context of tainted assets, this rule could infer that the first assets received (and subsequently spent) are the tainted ones, which mirrors the functioning of the FIFO algorithm.

4.3 Experiments

In this section, we embark on a series of experiments designed to dive deeper into the workings and efficacy of the taint system in a cryptocurrency environment. We aim to understand the system's performance under different scenarios, investigate the propagation of taints over time, and analyze the diffusion of a taint based on the chosen algorithm. The insights gathered from these experiments will provide a comprehensive understanding of the taint system and its capabilities, paving the way for optimized utilization in tracking tainted cryptocurrencies. Each experiment targets

a specific aspect of the taint system, leveraging a variety of methods to accurately evaluate the system's operation and effectiveness in different contexts.

4.3.1 Questions & Evaluation

1. **Local System Performance Evaluation**: Investigate how the performance of the taint system, specifically the time taken to execute and the memory consumption, varies on a local machine with different taint algorithms and blockchain dataset sizes. Evaluate this by measuring the execution time and memory usage of each taint algorithm using profiling and system monitoring tools, and assess how performance scales with different sizes of blockchain datasets.

2. **Taint Propagation Duration**: Determine how long the taint system takes to propagate a taint from one transaction to all subsequent transactions involving the tainted coins and explore how this duration varies with different taint algorithms. Evaluate this by capturing the time-stamp when the taint starts and stops propagating to measure total propagation time, and apply different taint algorithms on the same set of data to compare their propagation durations.

3. **Taint Diffusion Analysis for Different Algorithms**: Analyze how the diffusion of a taint varies with different taint algorithms. Evaluate this by calculating the percentage of tainted coins for each transaction in the dataset and tracking how this percentage changes over subsequent transactions. Also, compare the diffusion patterns of different taint algorithms, and use visualization tools to illustrate the diffusion of the taint through the transaction network.

4.3.2 Test bed

The test bed for the experiments described in this paper includes an Ubuntu 22.04 operating system and an AMD Ryzen 7 5800x 8-core processor with 100GB of memory.

The experiments were conducted using data from Ethereum block 9000000 to block 14000000, which were mined on November 27, 2019, and January 23, 2022, respectively.

The current implementation of an Ethereum node stores data in a key-value format, which poses challenges for analysis. To address this, we begin by converting Ethereum transactions into a more analyzable format within a Clickhouse SQL database. In our experimental process, we initially extract data from the SQL database to construct a transaction list, also referred to as an "edges list" in graph database terminology. This list is assembled as a memory-mapped file. Subsequently, we execute each taint algorithm, all of which have been developed using C++. In essence, we create a simulation of a straightforward in-memory graph database, as illustrated in Fig. 4.3.

4.3.3 Result

We discuss the findings of our experiments conducted to assess the performance of the taint system, the speed of propagation of the taint and the diffusion based on different algorithms. These results provide us with a deeper understanding of the system's functionalities, its strengths, as well as areas of improvement. They also give us insight into how the taint system can be effectively utilized and optimized for tracking tainted cryptocurrencies in various scenarios. By shedding light on the system's operation in diverse contexts, these results form a foundation for further enhancement of the taint tracking methods in the realm of cryptocurrency transactions.

Performance Evaluation

Taint Algorithm	Poison	Haircut	FIFO
Execution Time (second)	3016	1742	104
Memory Usage (gigabyte)	6.59	6.47	0.12

Table 4.1: Local System Performance Evaluation

Our experimental results, derived from running different taint policies on the

data from Ethereum blocks 9000000 to 14000000, reveal significant disparities in the performance of the three evaluated taint algorithms - Poison, Haircut, and FIFO. These findings are illustrated in Table 4.1. The Poison algorithm shows substantial execution time and memory usage, clocking at about 3016 seconds and consuming 6.59GB of memory. Similarly, the Haircut algorithm requires a considerable amount of time and memory to execute, about 1742 seconds and 6.47GB, respectively. However, in stark contrast, the FIFO algorithm dramatically outperforms both, with an impressively low execution time of 104 seconds and minuscule memory usage of just 0.12GB.

The relatively poor performance of the Haircut algorithm can be ascribed to its intrinsic complexity and higher computational requirements. In detail, the Haircut algorithm works by distributing the taint proportionately across the transaction outputs, an operation that necessitates managing fractional taint values and maintaining the taint percentages for each cryptocurrency unit throughout all subsequent transactions. This process leads to an escalation in calculations and data management. As a result, despite being thorough, the Haircut algorithm might be less efficient in settings where there are constraints on execution time and memory resources, especially when compared to the more streamlined Poison or FIFO methods.

The Poison algorithm, despite its higher resource consumption when compared to the FIFO method, remains an important taint analysis method due to its unique approach. Rather than proportionally distributing taint across outputs, Poison assumes that the entire amount of any transaction output is tainted if any tainted inputs are used. This allows the Poison algorithm to track the flow of tainted units more directly, although at the cost of higher computational overhead. The increased execution time of 3016 seconds and memory usage of 6.59GB, as indicated in table 4.1, reflect this greater demand on resources. Nonetheless, the Poison method may prove more effective in situations where a more conservative taint tracking is preferred or necessary.

On the other hand, the FIFO (First In, First Out) stands out in terms of efficiency.

As evidenced by its performance in table 4.1, the FIFO method requires significantly less execution time, just 104 seconds, and minimal memory usage, a mere 0.12GB. The FIFO algorithm operates based on the assumption that the first units received are the first ones to be spent. This simplistic approach reduces computational demands, making FIFO particularly well suited for environments with resource constraints or for tasks requiring quick taint analysis. Despite this, the FIFO method might not provide as comprehensive a taint tracking as Poison or Haircut, and its effectiveness may be dependent on the specific circumstances and the nature of transactions being analyzed.

Analysis of Taint Results Across Various Methods

In the experiments that follow, we employ the Upbit hack [11] as a detailed case study to showcase how our system is capable of forensically analyzing hackers' transaction movements, contrasting them with typical user behavior. We designate the initial hacked address, 0xa0987[1], and the stolen sum of 342,000 ETH, transacted at block 9007863, as the starting point for tainted address and amount. For comparison, we establish a baseline by randomly selecting 100 addresses, using their respective account balances to define the initial tainted addresses and amounts.

[1] 0xa09871aeadf4994ca12f5c0b6056bbd1d343c029

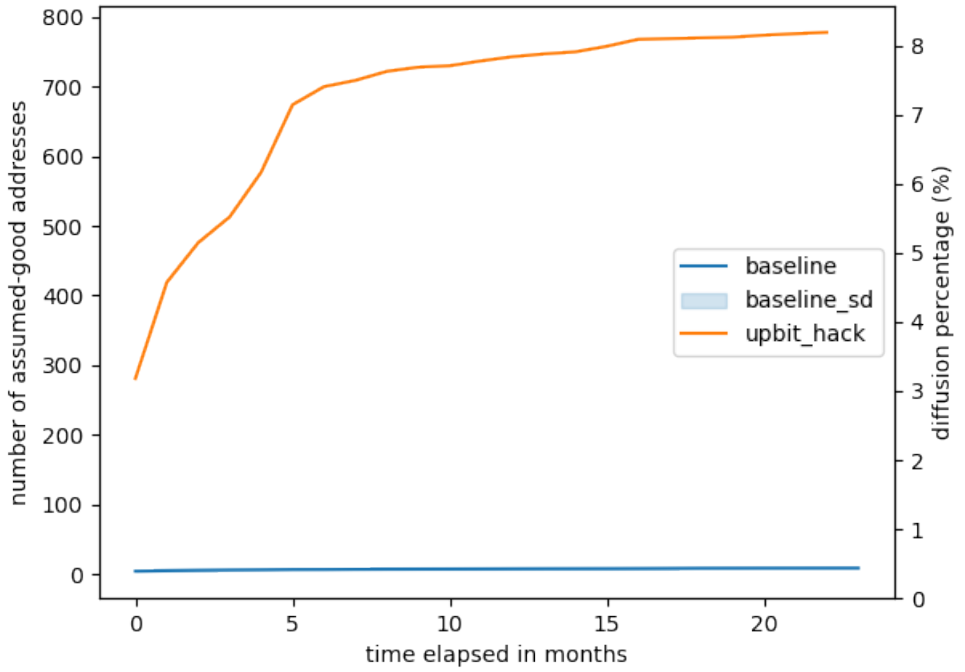


Figure 4.7: taint diffusion among the assumed-good addresses over time using FIFO method

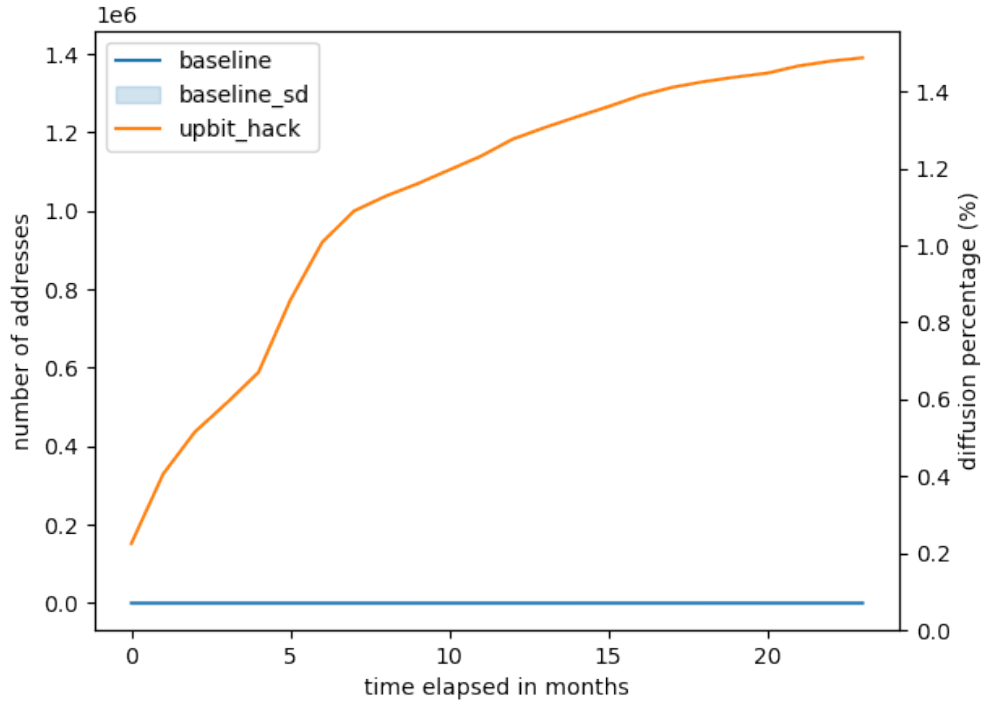


Figure 4.8: taint diffusion among all addresses over time using FIFO method

Fig. 4.7 provides a visual representation of the taint diffusion process among assumed-good addresses, utilizing the FIFO method, as it unfolds over time. In the baseline scenario, taint diffusion remains consistently below 1%, impacting fewer than 100 assumed-good addresses throughout the observed period. In stark contrast, the taint diffusion associated with the Upbit hack escalates rapidly, reaching 7% and affecting approximately 700 assumed-good addresses within a span of 5 months. This marked difference is likely attributable to the hacker's strategic transfers of funds to assumed-good addresses, such as exchanges and merchants, in an effort to convert the stolen cryptocurrency into fiat currency or tangible goods.

We observe similar findings in the taint diffusion process among all addresses, as depicted in Figure 4.8. Within the examined block range, there are 89,199,654 unique addresses. The diffusion grows readily in the first five months, a pattern we suspect is attributable to hackers moving funds to disposable addresses in an attempt to obfuscate the source of the funds.

Figure 4.9 illustrates the percentage of tainted amounts reaching Exchanges over time, analyzed using the FIFO method. It is important to note that the percentage fluctuates; this variation occurs as the tainted amount enters and leaves the exchanges in response to customer deposits and withdrawals. The patterns between the baseline and the Upbit hack case are highly distinguishable. In the baseline scenario, we observe that 50% of the tainted amount reaches exchanges within the first week. Our hypothesis, supported by Wu et al. [85], is that a large number of these addresses are exchange deposit addresses. These are utilized to receive funds from a user, and the Exchanges subsequently transfer them to their main address in a short time. In the Upbit hack case, the tainted amount does not reach Exchanges as quickly as in the baseline, likely because the hackers were moving funds to different addresses to layer the source in the fisrt a few weeks. There are some surges in the data that indicate the hackers were attempting to transfer cryptocurrency to Exchanges to cash out.

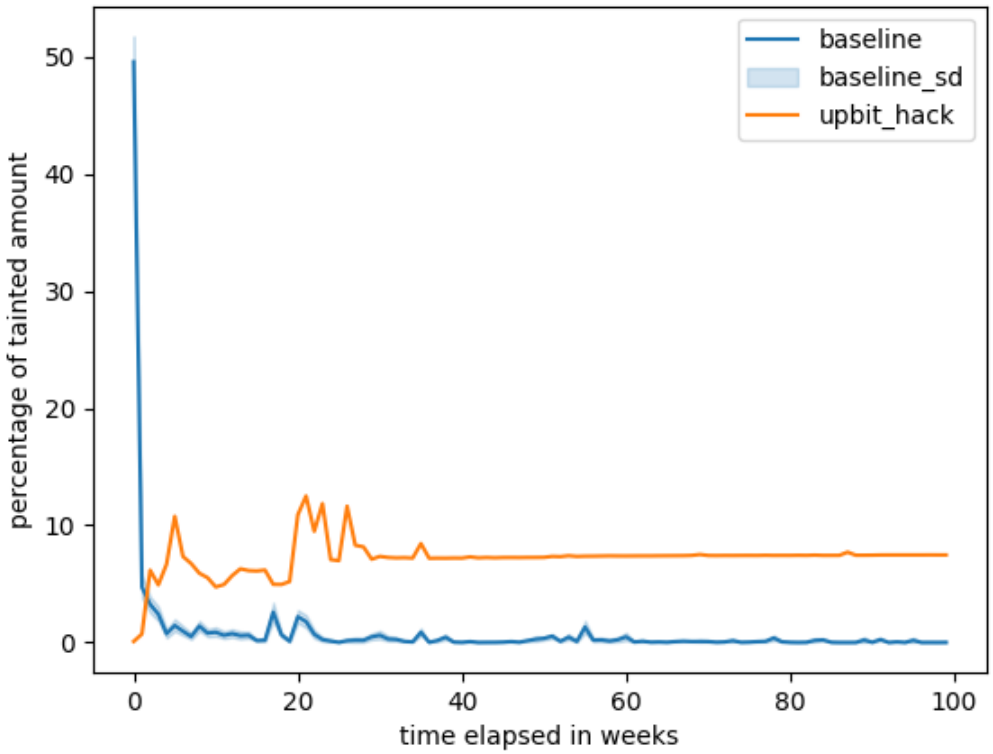


Figure 4.9: percentage of tainted amount reaches Exchanges over time using FIFO method

After 30 weeks, the percentage of tainted amount in Exchanges remains relatively flat, around 10%, suggesting that the amount is frozen and the tainted funds have ceased moving between addresses.

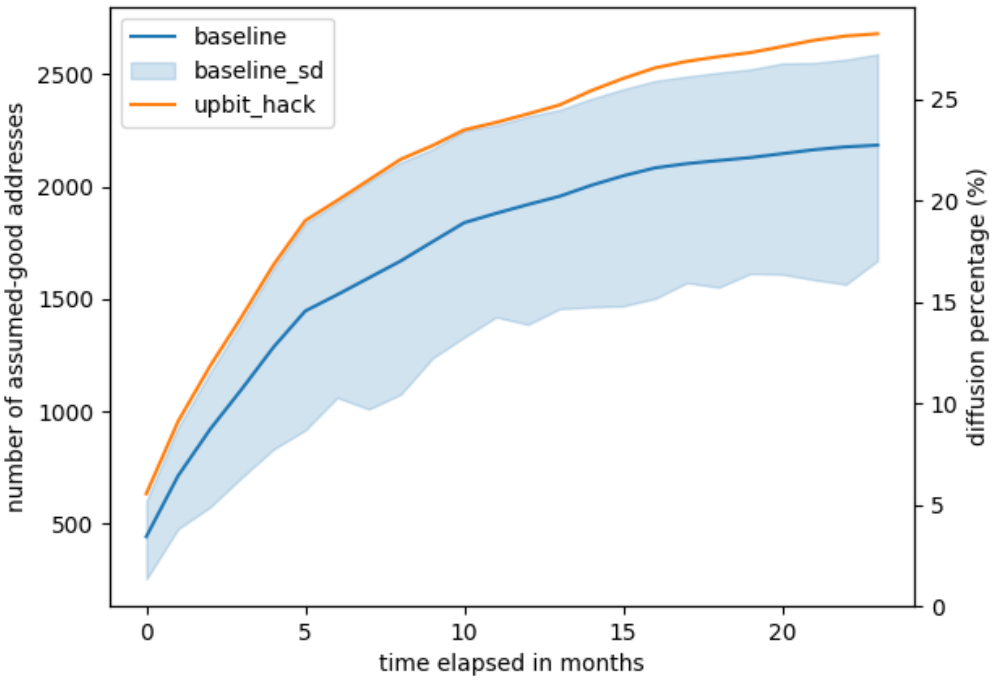


Figure 4.10: taint diffusion among the assumed-good addresses over time using haircut method

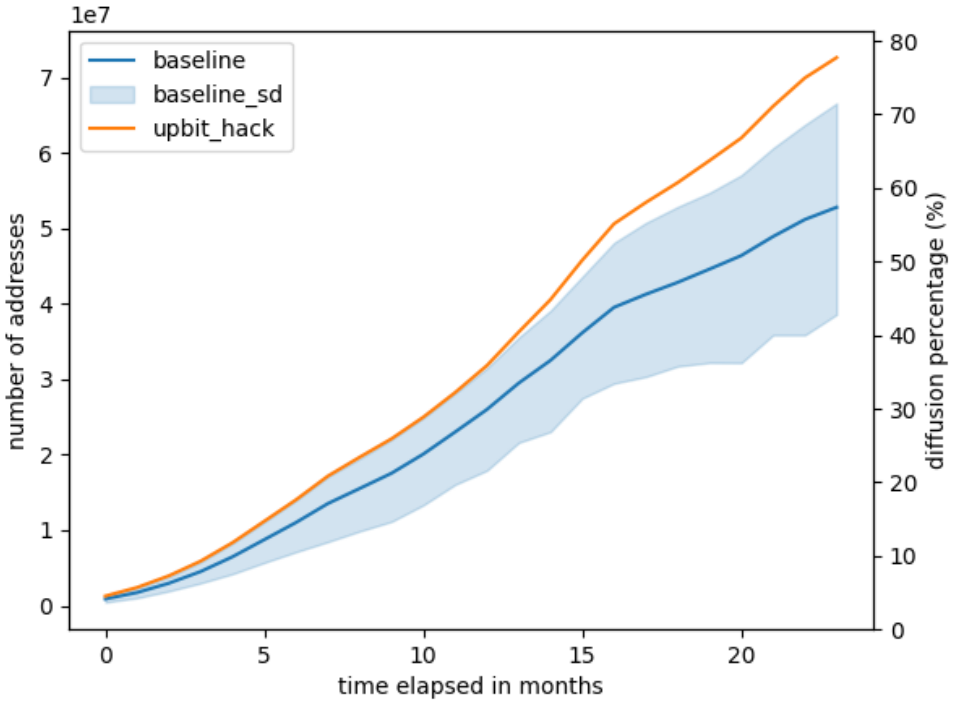


Figure 4.11: taint diffusion among all addresses over time using haircut method

Figure 4.10 illustrates the taint diffusion process among assume-good addresses, employing the haircut method to track this process as it unfolds over time. In the baseline scenario, the average taint diffusion escalates from 0% to approximately 24% of the assume-good addresses, encompassing around 2000 addresses in total. The light blue area within the plot serves as a visual representation of the error bar, reflecting 1 standard deviation from the mean, and thereby providing insight into the variability within the data. Notably, the Upbit hack case consistently hovers above the 1 standard deviation mark, underscoring its distinct behavior in the context of the study. This pattern is hypothesized to result from hackers' attempts to obfuscate the source of funds by distributing them across a greater number of addresses.

We observe similar behaviors in the diffusion process among all addresses using the haircut method as we do among assumed-good addresses, as depicted in Figure 4.11. In the case of the Upbit hack, the diffusion reaches 80% of all addresses, a value that is more than 1 standard deviation higher than the addresses in the baseline scenario.

The choice between FIFO and haircut methods will depend on the specific research objectives and the nature of the taint diffusion being studied. While the FIFO method excels in capturing temporal dynamics and specific case behaviors, the haircut method provides a more generalized view of taint distribution across a broader range of addresses. Our findings contribute to a deeper understanding of taint diffusion and offer valuable insights for forensic analysis, policy formulation, and the development of tools to trace illicit activities in the cryptocurrency domain.

Given that the poison method exhibits a pattern closely paralleling that of the haircut method, we have opted to forego a separate exposition of its results herein for the sake of brevity and focus.

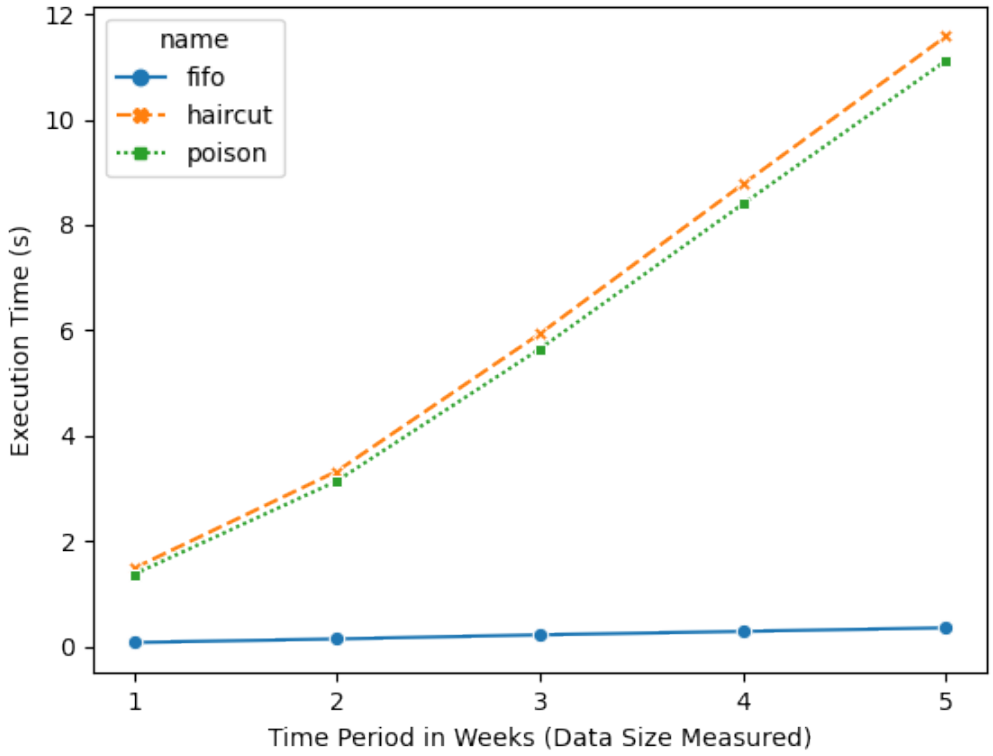


Figure 4.12: comparative execution times of FIFO, haircut, and poison methods over data size intervals

Method Scalability

Figure 4.12 presents the execution times of three analytical methods with respect to varying data sizes, quantified in weekly intervals. The execution time for the FIFO method remains relatively constant, whereas the execution times for both the haircut and poison methods demonstrate a linear increase in relation to the data size. This contrast is due to the different approaches to handling accounts with multiple outputs (withdrawals): The FIFO method affects only one output in each tainted transaction, while the Haircut and Poison methods require the allocation of tainted amounts to every output address. Furthermore, the haircut method exhibits a slightly longer execution time compared to the poison method. This is attributed to the haircut method's need to perform floating-point calculations to partially taint every output,

which is computationally more intensive.

4.4 Discussion

In the following discussion section, we explore the potential second-order effects of implementing a taint tracking system in the cryptocurrency ecosystem. These user concerns as well as responses and implications for various stakeholders, such as exchanges, regulatory bodies, and individual users. By considering these effects, we can better understand the potential impact of the taint system and work toward creating a balance between combating illicit activities and respecting user privacy.

4.4.1 Exchanges, Wallet Providers, and Regulatory Compliance

Cryptocurrency exchanges and wallet providers may develop policies to refuse or freeze assets identified as tainted and may be required to report such transactions to regulatory authorities. This could lead to the development of sophisticated screening tools to detect and handle tainted cryptocurrencies. Concurrently, regulatory bodies might revise their Anti-Money Laundering (AML) and Counter-Terrorist Financing (CTF) guidelines to include the handling of tainted cryptocurrencies, leading to stricter compliance requirements for businesses operating within the cryptocurrency space.

4.4.2 Individual Users and Market Dynamics

Regular users of cryptocurrency might become more vigilant about the sources of their cryptocurrency, possibly utilizing tools or services that allow them to verify if the cryptocurrency they are receiving is tainted. This could lead to a more informed and cautious user base. Additionally, tainted cryptocurrency might become devalued

as it becomes less liquid and harder to exchange, potentially creating a sub-market where tainted cryptocurrencies are traded at a discount compared to untainted ones.

4.4.3 Remedy Services Implications

Remedy Services are crucial for users to recover from receiving tainted funds, ensuring asset usability and supporting the integrity of the cryptocurrency system. Exchanges, at the heart of digital asset management, are in a prime position to offer these services, aligning with regulatory standards, and enhancing user convenience. By doing so, they can mitigate risks and educate users, positioning themselves as trusted entities.

Customer preferences could drive exchanges to adopt such services, similar to how credit card companies offer fraud protection. Exchanges that resist may lose customers to more secure platforms, indicating a market-driven push toward higher security standards.

Regulatory bodies could incentivize participation in the taint system through benefits for compliant exchanges and penalties for non-compliance, promoting a secure and responsible market.

In essence, the integration of a taint tracking system could reshape the cryptocurrency market, emphasizing security and trust, with stakeholders needing to adapt to maintain the ecosystem's integrity and security.

4.5 Conclusion

In our investigation, we have dissected the performance of different taint tracking algorithms in cryptocurrency environments, notably the Poison, Haircut, and FIFO methods. The Poison and Haircut algorithms were thorough but resource-intensive, whereas FIFO stood out for its efficiency.

The implementation of such taint tracking systems has significant implications for

stakeholders in the cryptocurrency ecosystem. Exchanges may need to adopt these systems for regulatory compliance and to maintain market trust, while users may become more discerning about the origins of their assets.

Our study also illustrated the diffusion of tainted funds, showcasing the potential of these algorithms to trace and mitigate the circulation of illicit funds in the blockchain network.

To sum up, the research provides a framework for enhancing taint tracking methods and highlights the need for a balanced approach that considers both the effectiveness of tracking and the impact on the cryptocurrency community. Future efforts should focus on optimizing these systems to address the challenges of financial crime while preserving user privacy and market integrity.

Chapter 5

Tracking IRSF Using Blockchain While Protecting Business Confidentiality

Decentralized delivery of physical or digital items via a sequence of handover actions is common in telecommunication, supply chains, snail mail, email, etc. In decentralized delivery systems, items are passed between carriers, from source to destination, without a central control, and often, by carriers that belong to different organizations. Delivery failures could be due to faults or the result of malicious actions like fraud, e.g., in International Revenue Share Fraud (IRSF), international phone calls are dropped by fraudulent telecommunication carriers. Tracking item delivery can help detect faults and fraudulent behavior. But the sequence of carriers used for delivery of a specific item is often business confidential, and should be revealed only in case of fraud.

In this chapter, we demonstrate a blockchain-based system, Fraud Buster, for confidential tracking of routes in a decentralized delivery system. In particular, we illustrate the ability to track handover of calls while preserving business confidentiality when detecting where calls were dropped. The paper makes the use of a permissioned

blockchain for tracking the required information yet revealing only the necessary information, when a fraud occurs.

Example. International phone calls go through several carriers. When a caller from the USA calls a number in Latvia, initially the service provider of the caller, say AT&T, would handle the call. A connection will be made with a company that can forward the call closer to the destination, say British Telecom (BT). BT will forward the call further, say to Orange Polska, which will deliver the call to Tet (Lattelecom) which will hand it over to Rigatta SIA, the carrier of the receiving number. Each carrier will make an independent business decision as to which carrier should be next in the chain, e.g., based on the requested handling fee. The customer who initiates the call pays its carrier, AT&T in this example. AT&T pays a fee to BT for handling the call, BT pays part of that to Orange Polska, which pays to Tet its share, and Tet pays to Rigatta SIA its fee for the connection with the call receiver.

Handling international calls, as presented in Example 5, is a typical case of decentralized delivery. Other examples are (1) international mail services, where a chain of handoffs between companies is established for delivering letters and packages to the destination address, (2) email in which mail servers forward the messages, (3) computer networks in which IP packets are transferred between servers, and (4) various supply chains, where delivery of physical items happens through a series of handovers between carriers.

A delivery may fail, due to a fault in the delivery system or a malicious action. Aborting the delivery maliciously is typically part of a fraud or an attack. If that happens, it is useful to be able to discover where the delivery failure occurred, e.g., which carrier dropped the call, dropped the IP packet, or failed to deliver the item to the next hop in the chain. This requires recording delivery information in a trusted way, and coping with the following three challenges. First, the distribution system is decentralized, so there is no entity that has all the information about deliveries and

carriers. Second, there might not be a single entity that all the involved organizations and individuals trust. Third, some of the information could be restricted by *business confidentiality*, as elaborated next.

Delivery chains in a decentralized system are often obscure or change frequently. For instance, the chain of carriers described in Example 5 could change if one of the carriers offered a cheaper price for handling the call. When BT needs to choose which company would be the next in the sequence and handle the call, it may prefer a different company to Orange Polska, e.g., Teo LT of Lithuania, if the fee requested by Teo LT is lower than the fee of Orange Polska.

In decentralized delivery systems, the carriers are often reluctant to reveal information about the route and the handoffs. For instance, business confidentiality may prevent revelation of information about handoffs. In Example 5, BT may not want to reveal its selection of Orange Polska and the incurred fee, to negotiate a lower fee with Teo LT. In the case of IP routing, the route can often be discovered using `traceroute` (or `tracert`). However, if VPN is used some information about the route would remain concealed.

5.1 International Revenue Share Fraud

International Revenue Share Fraud (IRSF) is the main motivating use case for our study. It is one of the most prevalent frauds plaguing the telecommunication industry [23, 41, 70]. A survey from 2017, conducted by the Communications Fraud Control Association[1] (CFCA), estimated that the revenue losses due to IRSF, for telecommunication companies worldwide, exceeded $10B yearly.[2]

IRSF occurs when a fraudulent international phone call is made and the fraudster, or an associate of the fraudster, is paid a portion of the cost of terminating the call.

[1] `https://www.cfca.org/`

[2] `https://gdpr.report/news/2017/05/29/telecommunications-battle-fraud/`

IRSF often entails an artificial inflation of traffic, i.e., traffic-pumping to international premium rate numbers (IPRN), or switching international calls to a fraudster carrier who drops the call, yet gets paid. The revenues of the premium number holder or of the carrier are shared with the fraudster. For instance, a PBX box can be hacked to issue many calls to the premium number, or via a carrier that participates in the fraud. In the first case, the fraudulent calls are often very long, because calls to a premium rate number are billed by duration of the call. In the second case, there is typically a high volume of very short calls. These are calls that are short stopped by the fraudster carrier. The fraudster carrier receives a small fee for each call and profits from "handling" many short calls that are dropped, i.e., not delivered to any end user.

When fraud is discovered, the records that are related to the fraud need to be identified. This is costly and today requires human labor, in particular, contacting other carriers to get the data related to the calls involved in the fraud. That data is stored in the form of Call Detail Records (CDRs). Since each carrier stores the CDRs of calls it handles in its own proprietary database, there is no single database that provides an overview of how calls were handled end-to-end, along the entire route from the initiator of the call to the end receiver. Furthermore, there is no mutually agreed upon trusted central entity that maintains such a comprehensive database across all carriers.

Example. As an example of IRSF, consider a fraudster that hacks into a private telephone system (PBX) of a customer of telecom company T, and issues calls to premium numbers in Latveria[3]. Since the T customer has not made the call, company T adjusts the customer's bill by refunding the customer for this call. However, since T forwarded the call to other carriers, it has to pay them for their service. A fraction of the payment goes to the fraudster carrier who terminates (or claims to terminate) the calls. The customer whose PBX has been hacked is not paying for the fraudulent

[3]Fictional nation appearing in American comic books.

calls, but the carrier of this customer (company T) loses money. All the other involved carriers, and in particular the one who terminates the call or provides the premium number, make a profit. The fraudster carrier shares the revenues from such fraudulent calls with the hacker. Note that international calls to some countries are quite expensive.

To reveal the fraudster carrier, suspicious calls are tracked, in order to discover the carriers that were part of the route, and if a call was dropped, find the carrier that short-stopped it. This requires access to information about the call. However, the information about each handover of the call is only stored in the databases of the involved carriers. Discovering the involved carriers, therefore, is an iterative and costly process. It would have been easier if there had been a central storage of all the information. But the different carriers are competitors and may not fully trust each other or let other carriers manage such information for them.

For coping with IRSF when carriers do not fully trust one another, we present a blockchain-based decentralized system that tracks dubious international calls, e.g., calls to suspicious numbers or a burst of calls to particular countries. The goal is to help track the termination of calls, and to mitigate fraud by making fraudulent behavior (or assistance to fraudulent behavior) traceable in an automatic way.

Example. Consider the IRSF case described in Example 5.1. The first carrier, T, only knows the next carrier to which it forwarded the call, say BT. To discover the carrier to which BT transferred the call, company T needs to approach BT and ask for the information. BT would need to look for the relevant *Call Detail Record* (CDR) in its records and provide that. Suppose that from the CDR, company T learns that the next carrier in the chain is Orange Polska. This requires approaching Orange Polska and getting the CDR from them, to know who is the next carrier in the chain. This continues until the fraudster carrier is revealed or when a carrier refuses to cooperate. It is a slow and expensive process, which in some cases could cost more than the losses

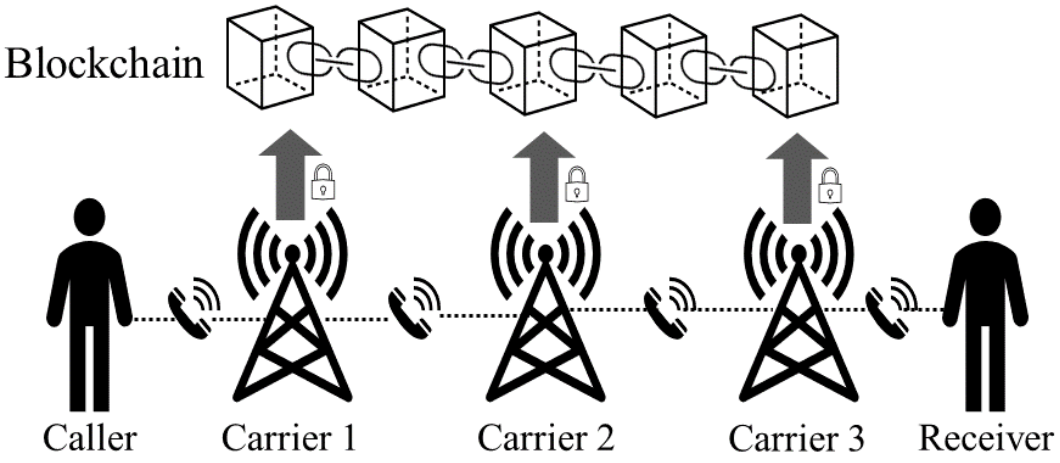


Figure 5.1: Handoff recordings on a blockchain

incurred by the fraud. If, however, all the information were already on a blockchain protected from tampering attempts of the fraudster, it would be easier to extract it and identify the carriers that handled the fraudulent calls.

In this chapter we consider four undesirable behaviors of carriers: (1) short stopping a call and not recording the call on the blockchain, (2) short stopping the call and adding a fake record to the blockchain as if a successful handoff has been executed, (3) handling the call properly but adding a false record to the blockchain, (4) handling the call properly without recording that on the blockchain. Our goal is to use a blockchain for revealing the shortstops while ensuring confidentiality in non-fraudulent cases.

5.2 Framework

We now present our framework, including terminology, notations and problem definition.

Decentralized delivery system. A *decentralized delivery system* consists of a

network of *carriers* and dispatched items. The items can be physical (e.g., parcel) or virtual (e.g., email, international call). Each dispatch (call) is between a pair of end users (sender-receiver/caller-callee).

Let C be the set of carriers and L be the set of links between the carriers. Let U be a set of *end users*, which could be senders (callers) and receivers (callees). A link between carriers c_1 and c_2 enables handover of items from carrier c_1 to carrier c_2. The set of carriers C and links L yields a directed graph $G = (C, L)$ where C are the nodes and L are the edges. Each end user can also handover an item to a carrier or receive an item from a carrier, but end users do not serve as intermediary carriers.

A *delivery task* is the duty to deliver an item or establish a phone call connection from end user $u_s \in U$ (sender) to end user $u_r \in U$ (receiver). This is executed using a set of carriers, which could belong to different organizations. A successful delivery is a sequence of handoffs that creates a path in G from a carrier connected to u_s to a carrier connected to u_r. Different methods can be used for the link selection per delivery task, e.g., based on estimations of the shortest path to the destination, the load on carriers or the handling fee. For instance, when forwarding a call, BT may select between Orange France, Orange Polska and Deutsche Telekom based on the fee each company charges for handling the call to the desired destination.

Successful delivery is a sequence $u_s, c_1, c_2, \ldots, c_n, u_r$ where

- end users $u_s \in U$ and $u_r \in U$ are sender and receiver;
- each pair of consecutive carriers c_i and c_{i+1} are linked, i.e., $(c_i, c_{i+1}) \in L$, and there is a successful handoff between c_i and c_{i+1};
- the u_s to c_1 and c_n to u_r handoffs are successful.

A *failed delivery* from sender u_s to receiver u_r is a sequence of handoffs $u_s, c_1, c_2, \ldots, c_i$ that starts in u_s but does not reach u_r. We refer to c_1 and c_i as the *first carrier* and *last carrier* of the failed delivery, respectively.

In the case of IRSF, only the first carrier loses money because it needs to compensate the sender and pay c_2 for the service. (Note that c_2 also needs to pay c_3 for the service, but it still makes a small profit, and the same is true for the other carriers c_j, $2 \leq j < i$.) Carrier i, the fraudster, receives a fee for handling the call but drops it and does not pay carrier c_{i+1} because there is no handoff of the call. The fraudster shares the revenue with the hacker that initiated the call. Carrier i could also be a carrier who makes the connection to a fraudster IPRN.

Example. Consider a successful delivery through four carriers $u_s, c_1, c_2, c_3, c_4, u_r$. User u_s pays \$1.8 to the home carrier c_1, for the call. Carrier c_1 pays \$1.2 to carrier c_2 for handling the call and gains \$0.6, carrier c_2 pays \$0.8 to c_3 and profits \$0.4, carrier c_3 pays \$0.4 to c_4 and keeps \$0.4, and c_4 delivers the call to u_r.

Now, consider a case where c_3 short stopped the call. The sequence is u_s, c_1, c_2, c_3. The carrier c_1 does not charge u_s for the call because it is not a genuine call of u_s. Carrier c_1 still pays \$1.2 to c_2 for handling the call, and c_2 pays \$0.8 to c_3. Since c_3 dropped the call, it does not need to pay to any other carrier. In this case, c_1 loses \$1.2, c_2 gets \$0.4 and c_3 gains \$0.8. A burst of 1000 such calls would lead to the case where c_1 loses \$1200 and c_3 gains \$800. A portion of the revenue of c_3 is shared with the hacker who hacked the PBX phone system and initiated the calls.

Registry using blockchain. To mitigate IRSF, the last carrier of suspected fraud calls (failed delivery) should be identified and made known to the first carrier, e.g., in the case of IRSF it could spare paying the fraudster carrier, and in the case of a technical failure, discover the carrier responsible for it.

A registry system that records all the handoffs could help in detecting the failure point. But managing such a system is challenging given that the delivery system comprises of carriers that may not fully trust each other. The registry should be trustworthy and should not be controlled by any single carrier. In a registry system that is controlled by a single organization, the controlling organization can deny

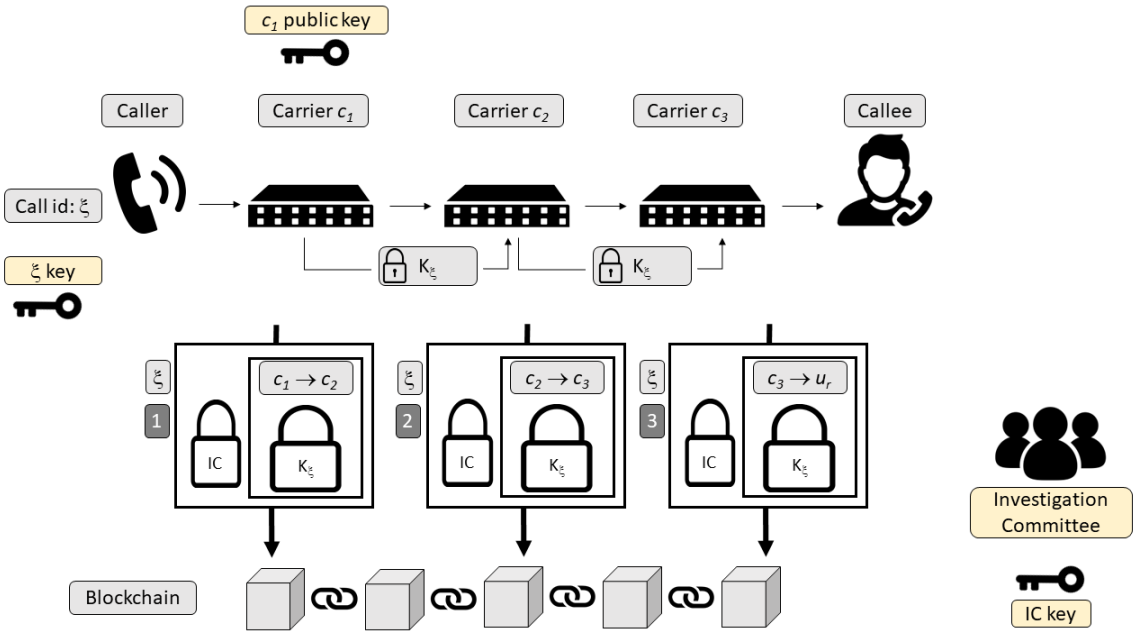


Figure 5.2: Recording handoffs under the All-to-All confidentiality model, where the handoffs are encrypted using a unique key created for each call, and the key is securely transferred to the carrier handling the call. The handoffs are encrypted twice—first with the unique key of the call and then with the key of the IC. The Last-to-All model is similar except that if the call is dropped, the IC only decrypts the last two records.

access to the information from other parties. In a registry system that has replicas stored on different nodes independently, it is difficult to guarantee that exactly the same information appears in all replicas. Therefore, we propose a solution based on blockchain.

Blockchain is a decentralized, tamper-proof and transparent ledger, managed by peers that are part of a peer-to-peer network [61]. The peers create blocks of transactions and add them to the chain in a way that guarantees consensus regarding valid transactions and their order. Blockchain was initially developed to prevent the "double spending" problem in cryptocurrencies [61], but recently, there has been a growing interest in using blockchains for a variety of applications where there is a need to reach consensus in a decentralized environment [22, 31, 53, 72, 80]. In the case of a permissioned blockchain, the set of peers is known and a consensus mechanism is used to decide which blocks are valid and can be added to the blockchain [35].

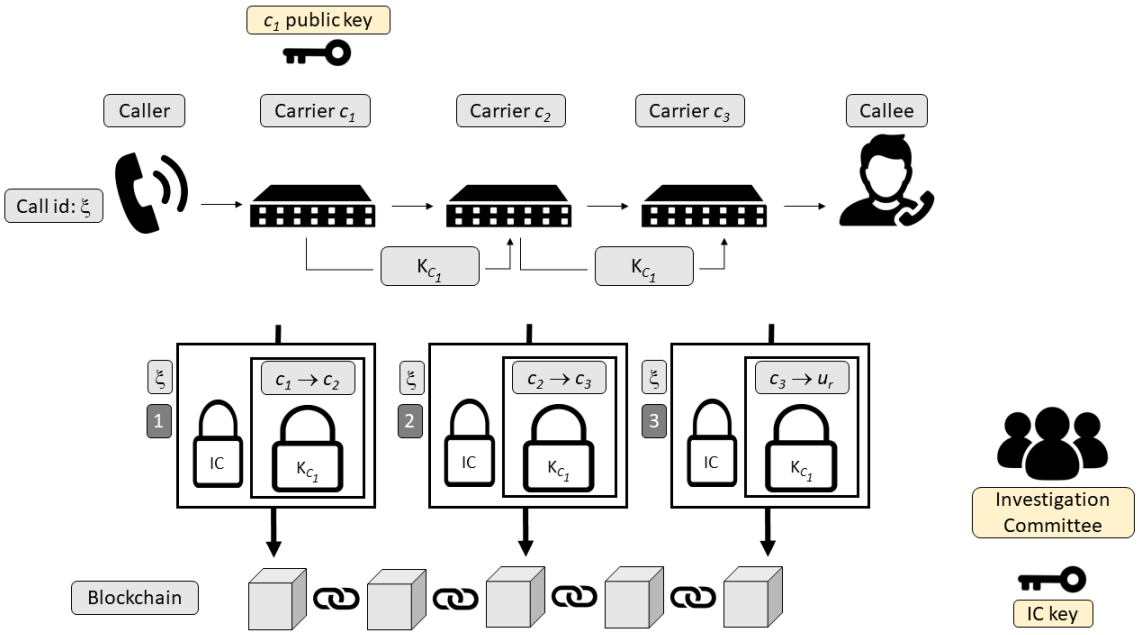


Figure 5.3: Recording handoffs under the All-to-First confidentiality model, where the handoffs are encrypted twice—first with the key of the first carrier and then with the key of the IC. The Last-to-First model is imlemented in the same way, except that if the call is dropped, the IC only decrypts the last two records.

In our system, a blockchain is used as a ledger to record delivery tasks and handoffs, see Fig. 5.1. It provides a tamper-proof decentralized log of the delivery tasks and their execution, and these records can be used to track handoffs and discover points of failure, i.e., the carrier that dropped the call (or failed to handover an item). In this paper we assume that the blockchain peers are entities that were assigned to manage the blockchain. They can be carriers, but do not have to be carriers or end users.

Confidentiality. Blockchains provide transparency, where all the peers that manage the blockchain see the stored transactions. This is an advantage in many applications. However, there is a conflict between transparency and business confidentiality. Carriers may not want information about their handoffs to be revealed to their competitors.

To achieve business confidentially, the following two general guidelines should be applied:

1. *minimize the exposed information*, and

2. *minimize the number of access permissions.*

Information should only be revealed when necessary and only to entities that need to see the information.

When minimizing the exposed information, the goal is to reduce the number of handoffs that are revealed to a third party. An example of such a restriction is to only reveal transactions that are part of a failed delivery. When minimizing the number of viewers, the set of carriers (or other entities) that are privy to the disclosed information should be as small as possible. For example, if a carrier does not participate in a delivery, it should not be exposed to information about that delivery.

We consider four confidentiality models, for a failed delivery $u_s, c_1, c_2, \ldots, c_i$.

- **All-to-All:** The handoffs of the delivery are revealed to the carriers $c_1, c_2, \ldots, c_i$.
- **All-to-First:** The handoffs of the delivery are revealed to the first carrier c_1.
- **Last-to-All:** Only the handoffs associated with the last two carriers c_{i-1} and c_i are revealed to the carriers $c_1, c_2, \ldots, c_i$.
- **Last-to-First:** Only the handoffs associated with the last two carriers c_{i-1} and c_i are revealed and only to the first carrier c_1.

Disclosing the handoffs to all the carriers on the path allows all of them to know that they are part of a failed delivery, so that they collectively could be responsible for the prevention of future failures. Revealing the information just to the first carrier provides stronger confidentiality. Similarly, revealing the entire set of handoffs on a route provides information that could be used to prevent reoccurring failures. Revealing just the last two carriers limits the exposure to only a small set of carriers and provides stronger confidentiality. Note that in a case of a malicious action, the carrier that drops a call or fails to deliver an item, say c_{i-1}, may try to conceal that by recording a handoff to c_i on the blockchain. In this case, information about both

c_{i-1} and c_i should be revealed, to examine which one is responsible for the short stop. We consider four behaviors of the carriers.

- **Honest** (but curious): the carrier transfers the call and correctly records that on the blockchain.
- **Fraudulent**: the carrier drops the call but records a fake transfer on the blockchain.
- **Sloppy**: the carrier transfers the call but fails to properly record that on the blockchain.
- **Malicious**: the carrier drops the call and records nothing on the blockchain.

A fraudulent carrier c_{i-1} may add to the blockchain a fake record that the call was forwarded to c_i. This is the reason for revealing the last two nodes of a failed delivery. Typically, the majority of the carriers are honest, but even the honest carriers should not be privy to confidential information (handoffs of other carriers).

Goal. Demonstrate a system that implements the four confidentiality models on top of a blockchain, in the presence of all four carrier behaviors.

5.3 Confidentiality Models

In this section, we discuss the implementation of the confidentiality models. In all the models, information about handoffs is initially concealed. We assume that there is a committee that is responsible for deciding when information should be revealed, and we refer to it as the *investigation committee* (IC). The IC consists of several independent entities. It is honest but curious. That is, the IC can be trusted to decide when information should be revealed but should not see any confidential information. It can be a proxy that executes a court order, or can be some other group of semi-trusted

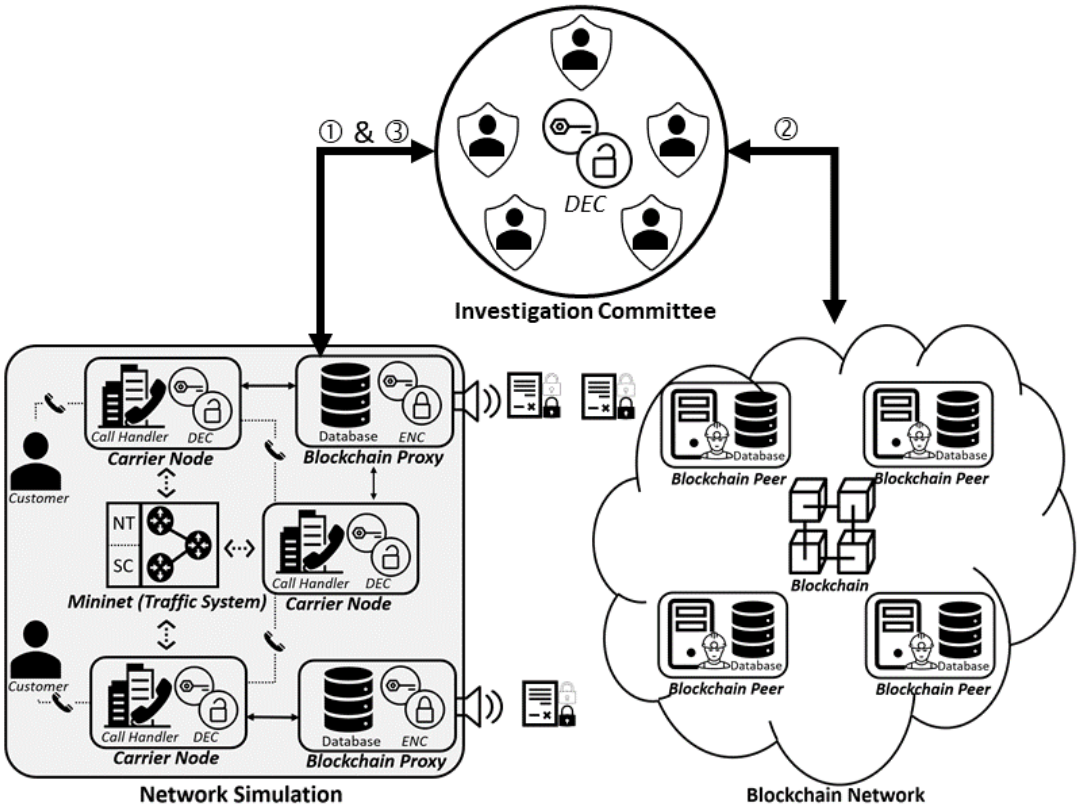


Figure 5.4: System architecture

entities. It can be implemented as a single node, as a distributed system with a consensus protocol, by the blockchain peers, etc.

The protocols use the RSA [67] public key cryptosystem [40], and each carrier c has a pair (K^c_{priv}, K^c_{pub}) of private and public keys. The public key can be used for encrypting of short texts, e.g., using optimal asymmetric encryption padding [24]. Given a message m, we denote by $\text{Enc}(m; K)$ and $\text{Dec}(m; K)$ the encryption and decryption of m using key K. Given a message m, a carrier c can sign m by applying to m a cryptographic hash function $h()$ like SHA2 [66], and encrypting the result using its private key, $s = \text{Sign}(m) = \text{Enc}(h(m); K^c_{priv})$. Any carrier or blockchain peer could validate a signature by checking that $h(m) = \text{Dec}(s; K^c_{pub})$. The IC $\mathcal{C}$ also has a pair $K^{\mathcal{C}}_{priv}$, $K^{\mathcal{C}}_{pub}$ of private and public keys.

Encrypting a long plaintext using a public key is inefficient. So, often for a message

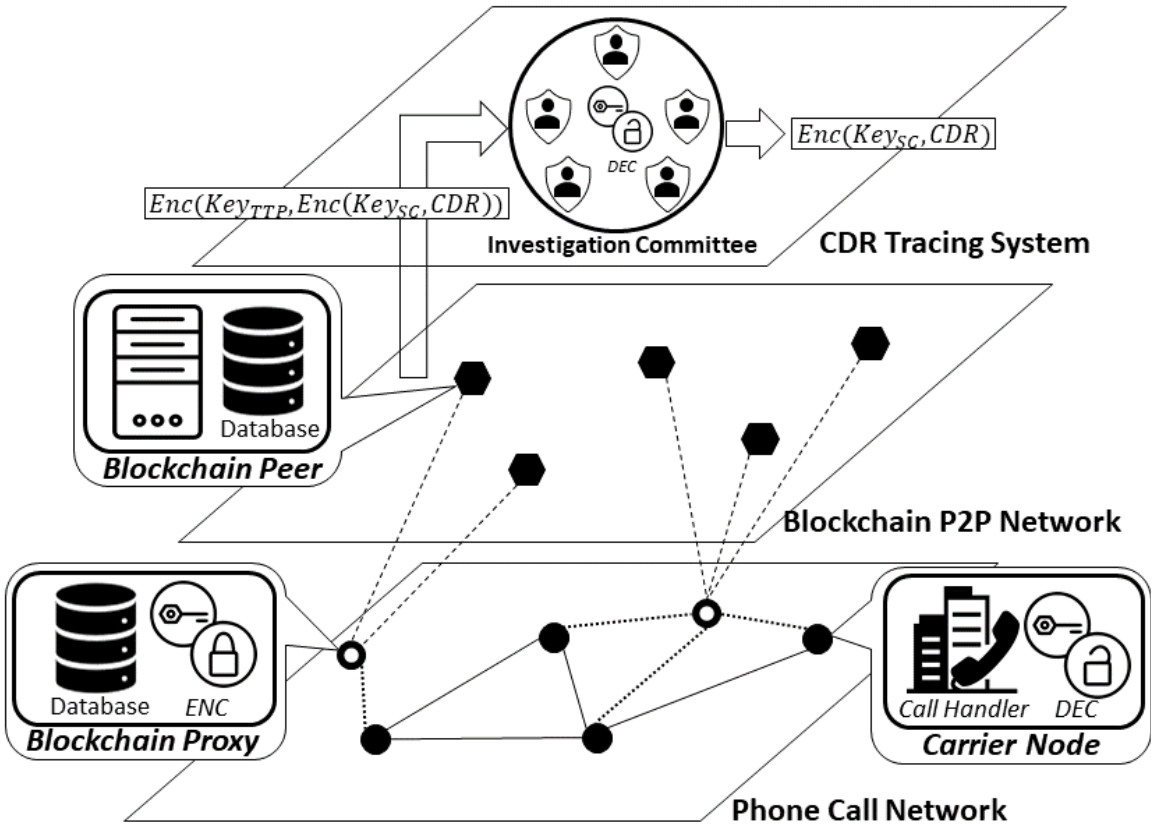


Figure 5.5: Layered network

m and a public key K_{pub}, the encryption is a two step process using a symmetric key like AES [30]. The symmetric key is used for both the encryption and the decryption [64]. First, a new secret symmetric key K_{sym} is created. The symmetric key is used to encrypt the message and the public key is used to encrypt the symmetric key. The two records $\text{Enc}(m; K_{sym})$ and $\text{Enc}(K_{sym}; K_{pub})$ are added to the blockchain. We denote this 2-step encryption by $\text{Enc2}(m, K_{pub})$.

All-to-All. Given a call with call id ξ, from sender u_s to receiver u_r, the route is recorded by storing handoffs on the blockchain. The handoffs are encrypted by a key that is shared with the relevant carriers, to prevent exposing the information to carriers that are not part of the route of call ξ. The encryption is by using a symmetric key K^{ξ}_{sym}, uniquely created for call ξ, e.g., using AES [30].

```
mininet@mininet-vm:mn$ sudo python Simulation.py
Building the route...
Unable to contact the remote controller at 127.0.0.1:6633
*** Creating network
*** Adding controller
*** Adding hosts:
agent h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 nat
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (h4, s4) (h5, s5) (h6, s6) (h7, s7) (h8, s8) (h9,
s9) (h10, s10) (s1, s2) (s1, s5) (s1, s6) (s2, s4) (s3, s5) (s3, s9) (s4, s9)
 (s5, s7) (s6, s8) (s6, s10) (s7, s8) (s8, s9) (s9, s10) (s11, agent) (s11, n
at) (s11, s1) (s11, s2) (s11, s3) (s11, s4) (s11, s5) (s11, s6) (s11, s7) (s1
1, s8) (s11, s9) (s11, s10)
*** Configuring hosts
agent h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 nat
*** Starting controller
ryu
*** Starting 11 switches
s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11 ...
*** Starting ryu.apps...
Calculating the shortest path
Setting the IP address for each switch
Setting the link interface address for each switch
Setting the routing table for each switch
(100.00000% loss) (100.00000% loss)
h1 -> h2 h3 h4 h5 h6 h7 X X X agent
h2 -> h1 h3 h4 h5 h6 h7 X h9 X agent
h3 -> h1 h2 h4 h5 h6 h7 X h9 h10 agent
h4 -> h1 h2 h3 h5 h6 h7 h8 h9 h10 agent
h5 -> h1 h2 h3 h4 h6 h7 X X X agent
h6 -> h1 h2 h3 h4 h5 X X X X agent
h7 -> h1 h2 h3 h4 h5 X h8 h9 h10 agent
h8 -> X X X h4 X X h7 h9 h10 agent
h9 -> X h2 h3 h4 X X h7 h8 h10 agent
h10 -> X X h3 h4 X X h7 h8 h9 agent
agent -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10
*** Results: 23% dropped (84/110 received)
*** Starting CLI:
Continue?

* Simulation Begins
 12%|████                                  | 14/120 [00:14<01:46,  1.00s/it]
```

Figure 5.6: Mininet control: the simulator creates the network nodes, the links, i.e., the connections between nodes, and the routing tables for the nodes. In the simulation, calls are simulated by IP packets, which are routed according to the routing tables. Fraudulent or malicious nodes may drop the packets, as an execution of a short stop.

Carrier c_1 creates the key K^{ξ}_{sym}. In step i, carrier c_i records on the blockchain the encrypted handoff information $(\xi, \text{Enc2}(r; K^{\mathcal{C}}_{pub}))$, where $r = \text{Enc}((c_i, c_{i+1}, \text{CDR}^i_{\xi}); K^{\xi}_{sym})$ and CDR^i_{ξ} is the call detail record of call ξ when handled by carrier c_i. The CDR contains details about the call such as time, duration, completion status, source and

destination numbers, etc. Carrier c_i hands over the encrypted key K^{ξ}_{sym}, to c_{i+1}. That is, c_i encrypts the key using the public key of c_{i+1} and sends the encrypted key $\text{Enc}(K^{\xi}_{sym}; K^{c_{i+1}}_{pub})$ to c_{i+1}. Carrier c_{i+1} can decrypt the message using its private key to discover K^{ξ}_{sym}. Note that the handoff information is encrypted twice, first using the key created by the first carrier and then by the public key of the IC. By the end of this process, the carriers involved in handling the call ξ store the encrypted handoffs on the blockchain. An example of a call and the records that are stored on the blockchain are depicted in Fig. 5.2.

When the handoffs of call ξ need to be disclosed due to an indication that ξ was short stopped, the IC $\mathcal{C}$ decrypts all the transactions with call identifier ξ and adds the decrypted transactions to the blockchain. The decrypted transactions are still encrypted by K^{ξ}_{sym}, so $\mathcal{C}$ and all the carriers that are not part of call ξ cannot see the handoffs. Carriers that have K^{ξ}_{sym} can decrypt the handoff records of ξ.

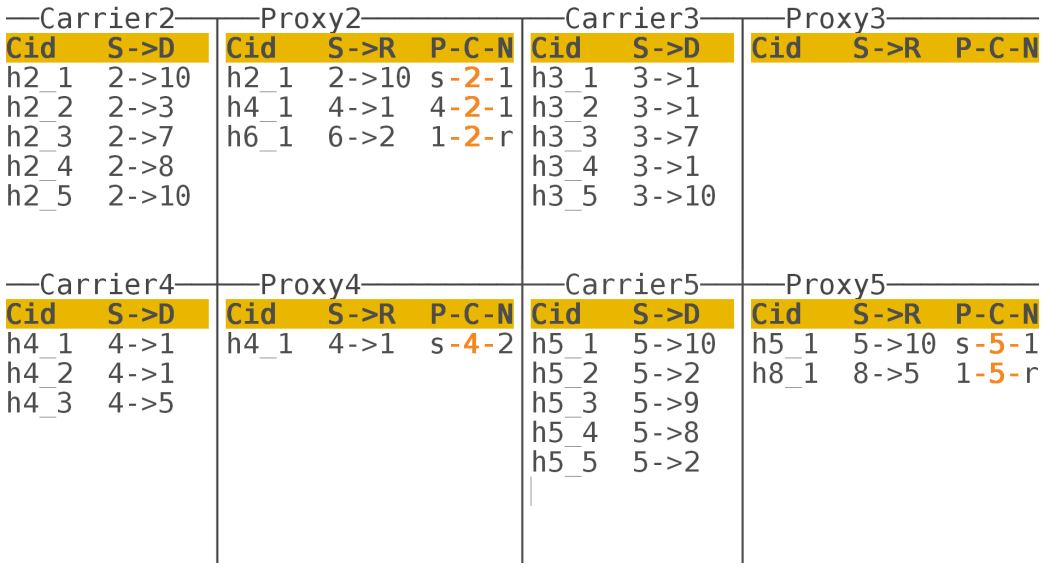
There is no single carrier that stores all the information, can change stored transactions or can deny access to stored transactions. Moreover, the IC does not need to store or see any information on calls.

All-to-First. To change All-to-All into All-to-First, the public key $K^{c_1}_{pub}$ of the first carrier c_1 is used instead of the key K^{ξ}_{sym}. In the handoffs, each carrier c_i instructs the next carrier c_{i+1} to use $K^{c_1}_{pub}$. The record stored by c_i on the blockchain has the form

$$(\xi, \text{Enc2}(\text{Enc2}((c_i, c_{i+1}, \text{CDR}^i_{\xi}); K^{c_1}_{pub}); K^{\mathcal{C}}_{pub})).$$

An example of a call and the records that are stored on the blockchain are depicted in Fig. 5.3. Only c_1 has the private key to decrypt the transactions, so only c_1 can see the handoffs. However, since transactions are also encrypted using the key of the IC, carrier c_1 can see the handoffs only after the decryption of the information by IC.

Last-to-All. In this model, the last two handoffs of ξ should be identified. This is done by adding a handoff number to the records, where carrier c_i records on the



Carrier2

Cid	S->D
h2_1	2->10
h2_2	2->3
h2_3	2->7
h2_4	2->8
h2_5	2->10

Proxy2

Cid	S->R	P-C-N
h2_1	2->10	s-2-1
h4_1	4->1	4-2-1
h6_1	6->2	1-2-r

Carrier3

Cid	S->D
h3_1	3->1
h3_2	3->1
h3_3	3->7
h3_4	3->1
h3_5	3->10

Proxy3

Cid	S->R	P-C-N

Carrier4

Cid	S->D
h4_1	4->1
h4_2	4->1
h4_3	4->5

Proxy4

Cid	S->R	P-C-N
h4_1	4->1	s-4-2

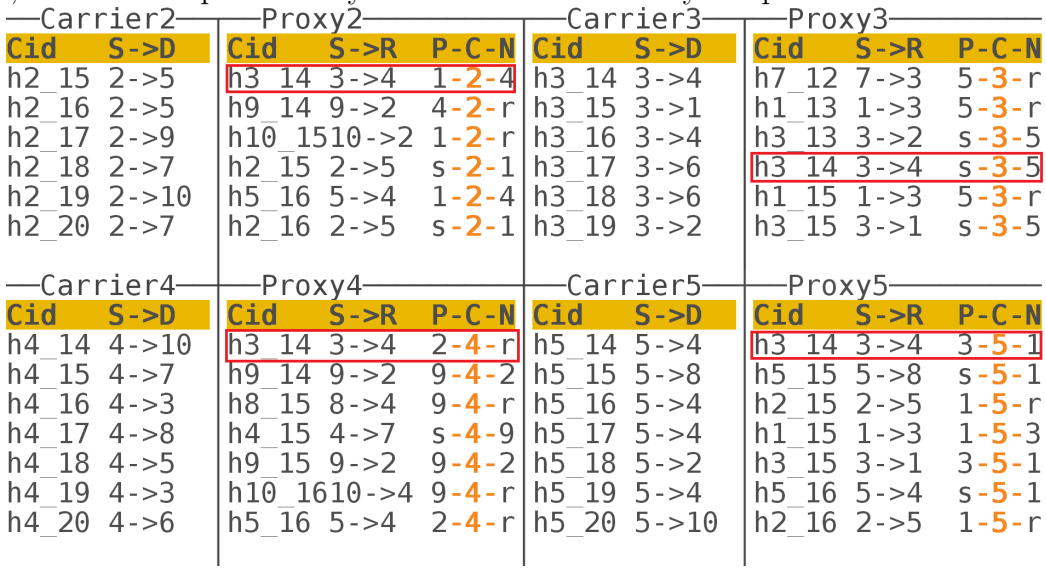
Carrier5

Cid	S->D
h5_1	5->10
h5_2	5->2
h5_3	5->9
h5_4	5->8
h5_5	5->2

Proxy5

Cid	S->R	P-C-N
h5_1	5->10	s-5-1
h8_1	8->5	1-5-r

Figure 5.7: Handover transactions that four carriers and their proxies see. In real time, carriers and proxies only see handovers that they are part of.



Carrier2

Cid	S->D
h2_15	2->5
h2_16	2->5
h2_17	2->9
h2_18	2->7
h2_19	2->10
h2_20	2->7

Proxy2

Cid	S->R	P-C-N
h3_14	3->4	1-2-4
h9_14	9->2	4-2-r
h10_15	10->2	1-2-r
h2_15	2->5	s-2-1
h5_16	5->4	1-2-4
h2_16	2->5	s-2-1

Carrier3

Cid	S->D
h3_14	3->4
h3_15	3->1
h3_16	3->4
h3_17	3->6
h3_18	3->6
h3_19	3->2

Proxy3

Cid	S->R	P-C-N
h7_12	7->3	5-3-r
h1_13	1->3	5-3-r
h3_13	3->2	s-3-5
h3_14	3->4	s-3-5
h1_15	1->3	5-3-r
h3_15	3->1	s-3-5

Carrier4

Cid	S->D
h4_14	4->10
h4_15	4->7
h4_16	4->3
h4_17	4->8
h4_18	4->5
h4_19	4->3
h4_20	4->6

Proxy4

Cid	S->R	P-C-N
h3_14	3->4	2-4-r
h9_14	9->2	9-4-2
h8_15	8->4	9-4-r
h4_15	4->7	s-4-9
h9_15	9->2	9-4-2
h10_16	10->4	9-4-r
h5_16	5->4	2-4-r

Carrier5

Cid	S->D
h5_14	5->4
h5_15	5->8
h5_16	5->4
h5_17	5->4
h5_18	5->2
h5_19	5->4
h5_20	5->10

Proxy5

Cid	S->R	P-C-N
h3_14	3->4	3-5-1
h5_15	5->8	s-5-1
h2_15	2->5	1-5-r
h1_15	1->3	1-5-3
h3_15	3->1	3-5-1
h5_16	5->4	s-5-1
h2_16	2->5	1-5-r

Figure 5.8: The setting of Fig. 5.7 after a while. The handover transactions of call `h3_14` are circled, to show the route of this call.

blockchain

$$\hat{\xi}_i = (\xi, i, \mathrm{Enc2}(\mathrm{Enc}((c_i, c_{i+1}, \mathrm{CDR}^i_\xi); K^\xi_{sym}); K^{\mathcal{C}}_{pub})).$$

As integrity constraint, the blockchain peers verify when adding a record (ξ, i, m) to the blockchain that it contains a record of the form $(\xi, i-1, m')$ and does not contain a record of the form (ξ, i, m''), for any cipher texts (encrypted content) m, m' and m''.

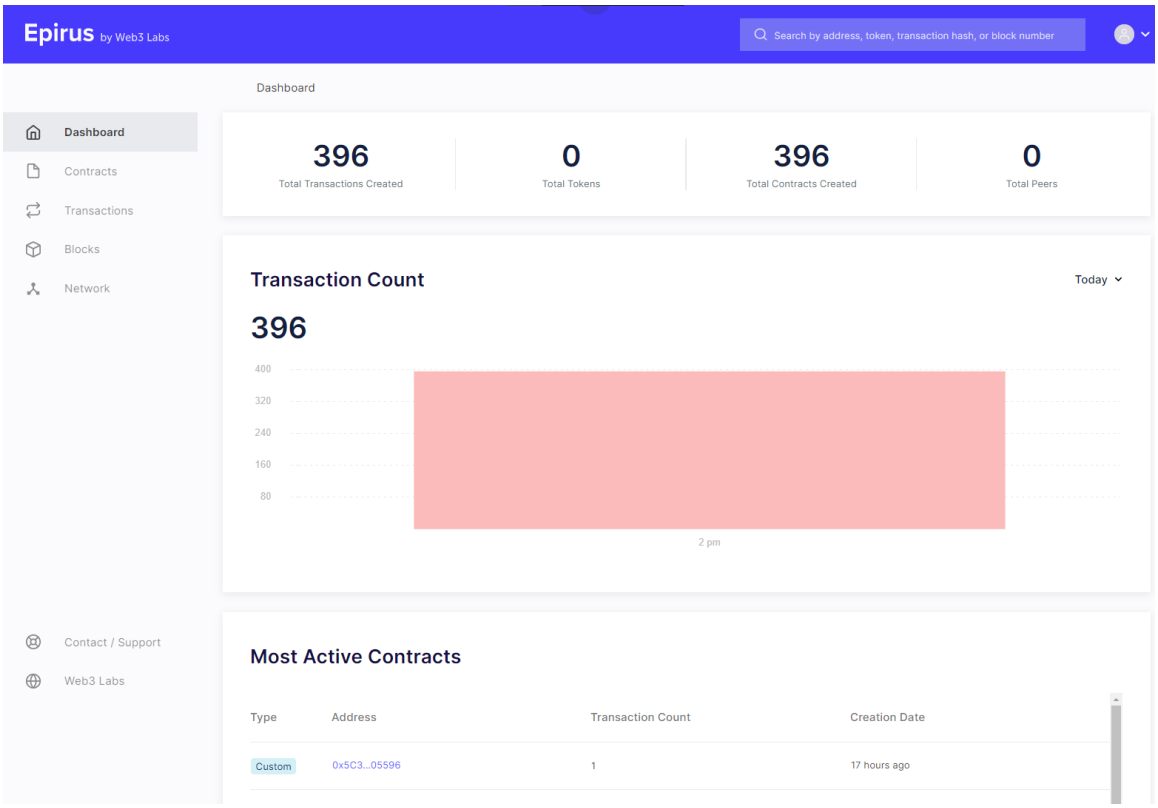


Figure 5.9: Blockchain dashboard.

The transfer of the shared key K^{ξ}_{sym} is the same as the key transfer in All-to-All.

When a call ξ seems to be part of a fraud (the sender u_s claims it is the result of a hack and the first carrier c_1 needs to investigate it), the IC $\mathcal{C}$ decrypts the last two records of ξ and sends to c_1 the records $\text{Dec}(\hat{\xi_{i-1}}; K^{\mathcal{C}}_{priv})$ and $\text{Dec}(\hat{\xi_i}; K^{\mathcal{C}}_{priv})$. The carrier c_1 would still need to investigate which one of the two carriers c_i and c_{i+1} is the fraudster. It could be that c_i is a fraudster that added a fake transfer record ξ_i, but in this case the record ξ_{i-1} could provide the information about the handoff of the call to c_i. Another case is that c_{i+1} is malicious and it will not add any record to the blockchain. In this case, the record ξ_i will provide information about the handoff of the call to c_{i+1}. Carrier c_1 cannot distinguish between these cases without an investigation. In all these cases, only records decrypted by $\mathcal{C}$ are visible and only to the carriers that hold the key K^{ξ}_{sym}.

Last-to-First. Implementing the Last-to-First model is the same as the imple-

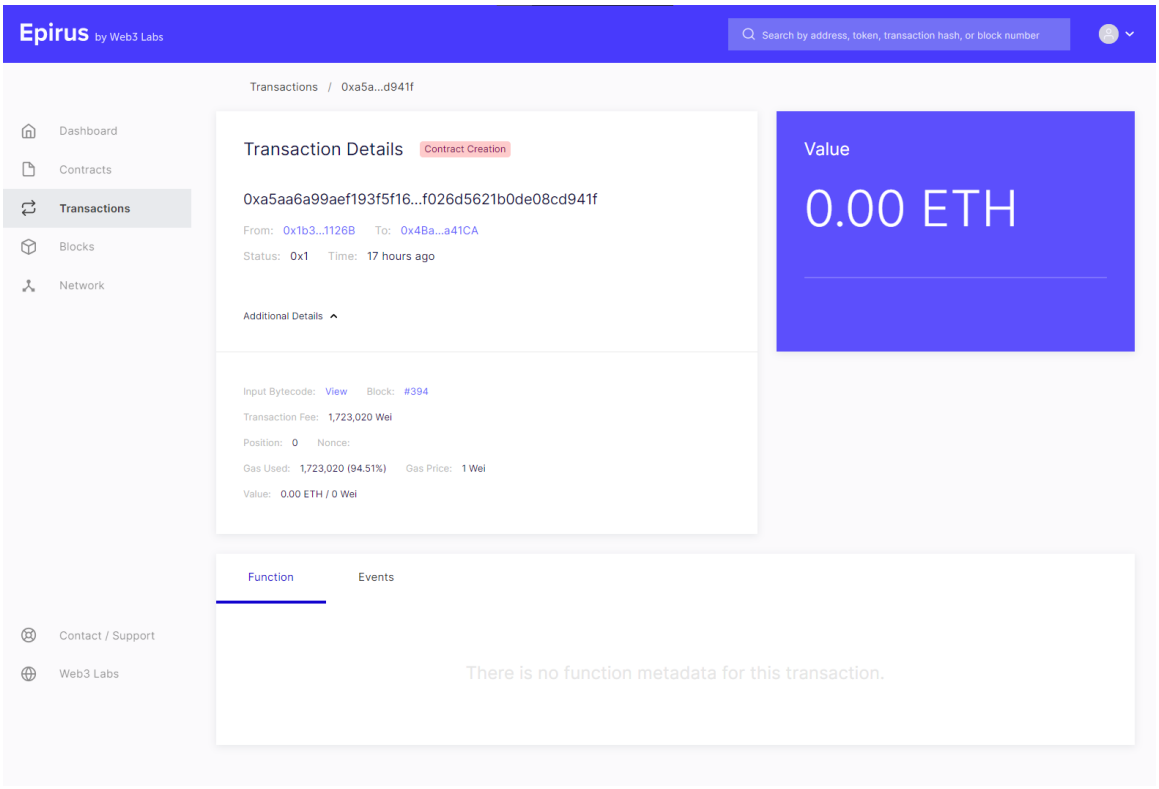


Figure 5.10: Transaction details.

mentation of the Last-to-All model except that the public key of the first carrier $K_{pub}^{c_1}$ is used instead of the shared key of the call K_{sym}^{ξ}.

Using Secret Sharing for the IC. When the IC consists of several entities, deciding when to decrypt records should be based on a consensus mechanism, e.g., for an IC of size n and some $0 < k \leq n$, requiring that any subset of k members of the committee could decrypt records on the blockchain, but a smaller subset would not be able to do so. This can be implemented using *secret sharing*. In FRAUD BUSTER, we are using Shamir's secret sharing [71].

In Shamir's secret sharing, first a large prime number p is selected and all the computations are executed modulo p, that is, with respect to the finite field $\mathbb{Z}_p = \mathbb{Z}/p\mathbb{Z}$. The secret is a number $s \in \mathbb{Z}_p$. The secret holder creates $k-1$ random numbers $a_1, \ldots a_{k-1}$, where $a_i \in \mathbb{Z}_p$ for $1 \leq i < k$. These numbers define a polynomial $f(x) = s + \sum_{i=1}^{k-1} a_i x^i$ of degree $k-1$. Then it sends to each peer j the encrypted secret

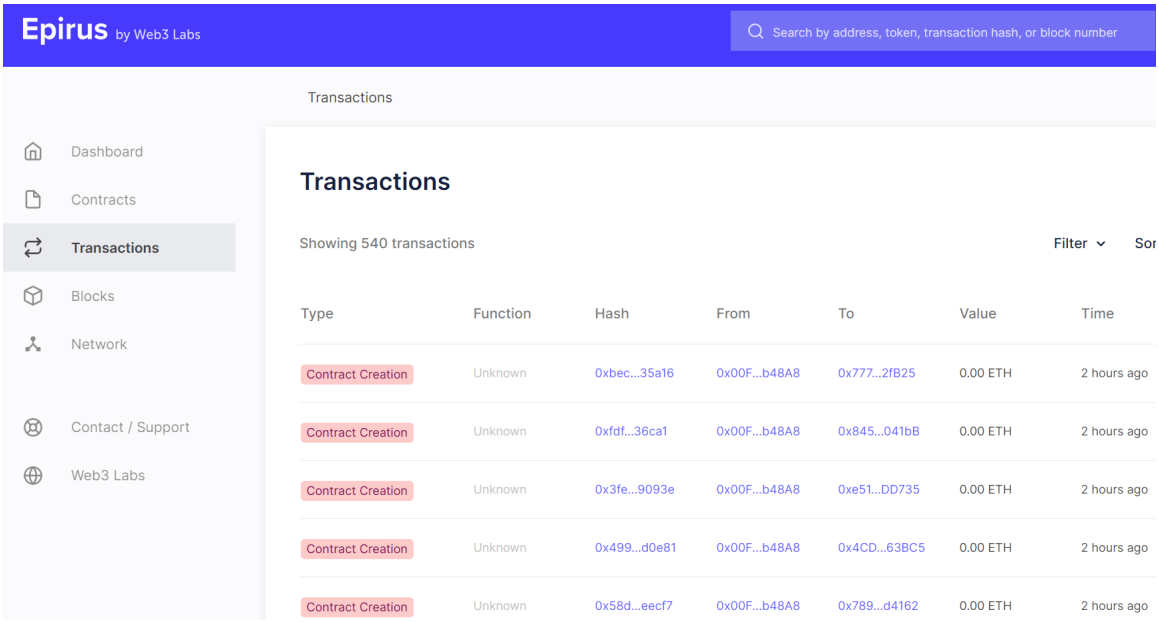


Figure 5.11: Transaction explorer.

point $\mathrm{Enc}((j, f(j)); K_{pub}^{peer_j})$. Any set of k or more peers can reveal their points to each other, and using interpolation the polynomial $f(x)$ can be computed, to discover the secret s. Any group of less than k peers cannot discover the secret s even if they reveal to each other the points they received.

The goal is to create and disseminate a shared secret for the committee members in $\mathcal{C}$. This is done as follows. Consider the delivery $u_s, c_1, c_2, \ldots, c_i, \ldots$ of call ξ, where in each handover, the carrier records encrypted CDR and delivery details, on the blockchain. When carrier c_i encrypts the information, it uses the following protocol.

1. Carrier c_i generates a new pair of matching public and private keys $(K_{pub}^{c_i,\xi}, K_{priv}^{c_i,\xi})$.

2. Carrier c_i generates a secret sharing scheme where the secret is $s = K_{priv}^{c_i,\xi}$. It generates a polynomial $f_i(x)$ of degree k over $\mathbb{Z}_p$ where $f_i(0) = s$, and sends to each peer j the encrypted pair $\mathrm{Enc}((j, f(j)); K_{pub}^{peer_j})$.

3. Carrier c_i adds to the blockchain the record

$$(\xi, i, \mathrm{Enc2}(\mathrm{Enc2}((c_i, c_{i+1}, \mathrm{CDR}_\xi^i); K_{pub}^{c_1}); K_{pub}^{c_i,\xi}))$$

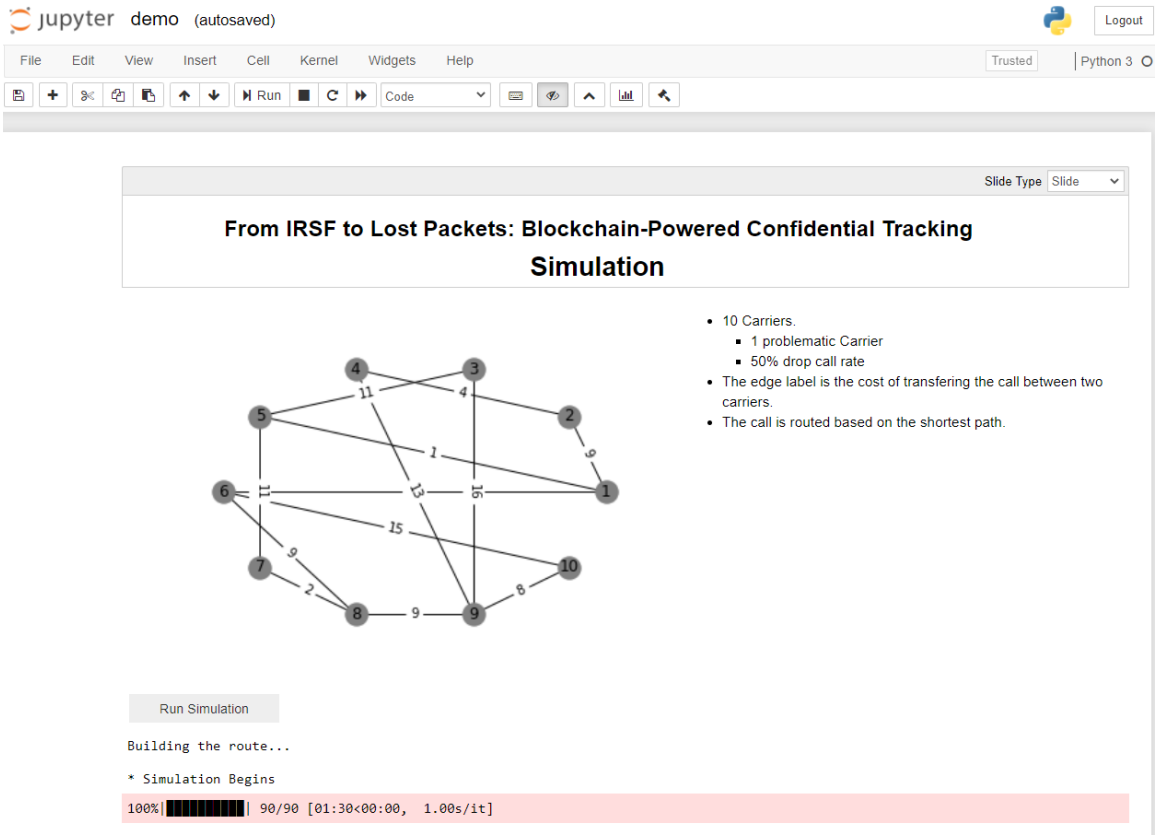


Figure 5.12: Step 1 of the simulation: the network is created, by defining nodes and links, and node behaviors (e.g., fraudster or malicious) are selected.

under the models All-to-First and Last-to-First; and it adds to the blockchain the record

$$(\xi, i, \mathrm{Enc2}(\mathrm{Enc}((c_i, c_{i+1}, \mathrm{CDR}^i_\xi); K^\xi_{sym}); K^{c_i,\xi}_{pub}))$$

under the models All-to-All and Last-to-All.

Decryption requires k or more members of the IC, to reveal their share of the secret, compute the decryption key $K^{c_i,\xi}_{priv}$ and use it to decrypt the record on the blockchain. The decrypted records are still encrypted by key $K^{c_1}_{pub}$ in the All-to-First and Last-to-First models, or by K^ξ_{sym} in the All-to-All and Last-to-All models. In the first case, only the carrier c_1 can decrypt them using its private key. In the second case, all the carriers on the route, who received K^ξ_{sym}, can decrypt the information. The information is not revealed to any other entities, including members of the IC.

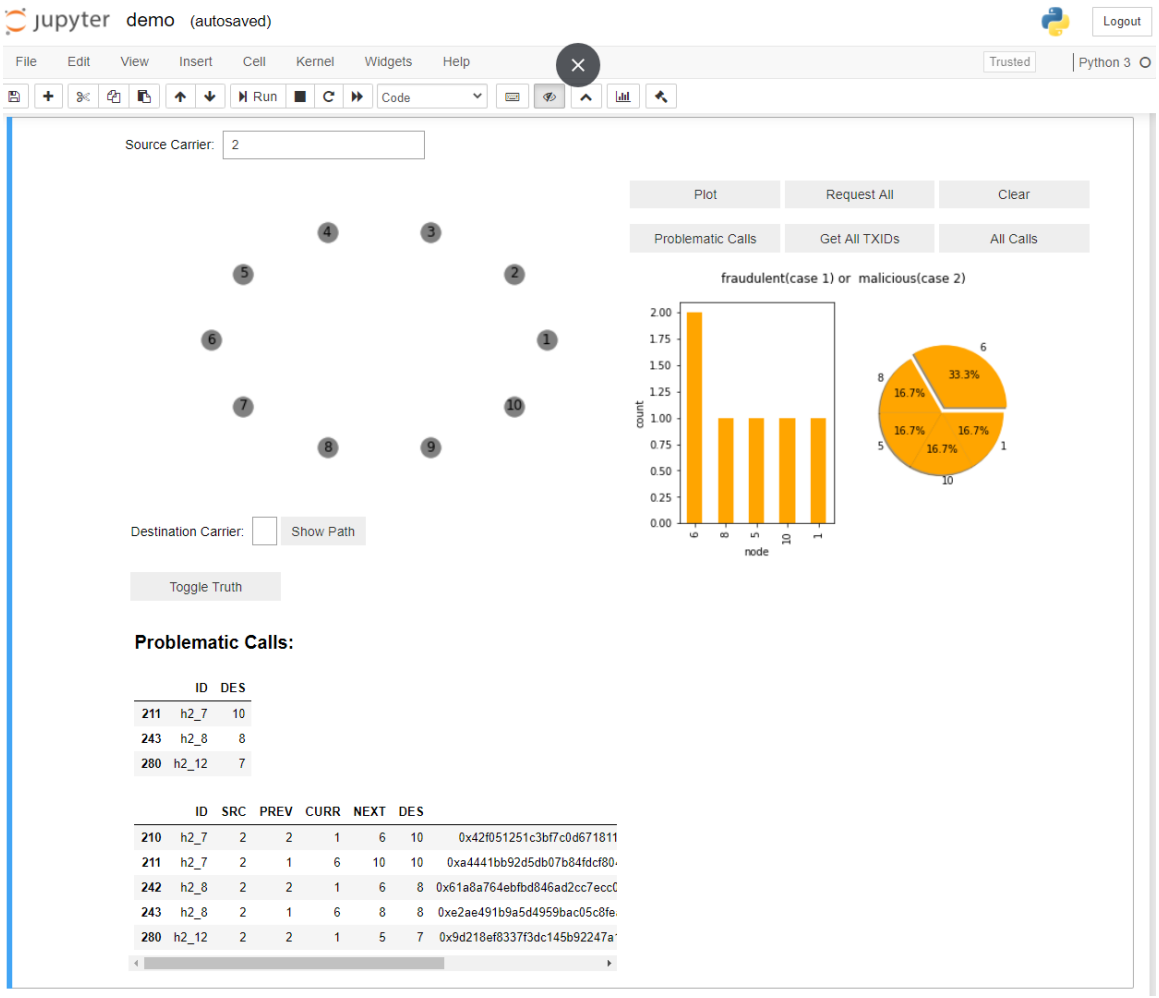


Figure 5.13: Step 2: calls are dispatched and the handover transactions are recorded on the blockchain. Carriers only see handovers they were part of.

5.4 Fraud Buster System

We now describe the Fraud Buster system. The system architecture is depicted in Fig. 5.4. It has three parts: (1) a network simulator, (2) the blockchain managed by the blockchain peers, and (3) the investigation committee (IC) component. This can be seen as a layered network where the lowermost layer consists of the network and the transfer of calls, the middle layer is the recording of call handoffs on the blockchain. The top layer is the application of tracking call routes confidentially. The layers are depicted in Fig. 5.5.

Network simulator. Network simulation is implemented using Mininet (`http:`

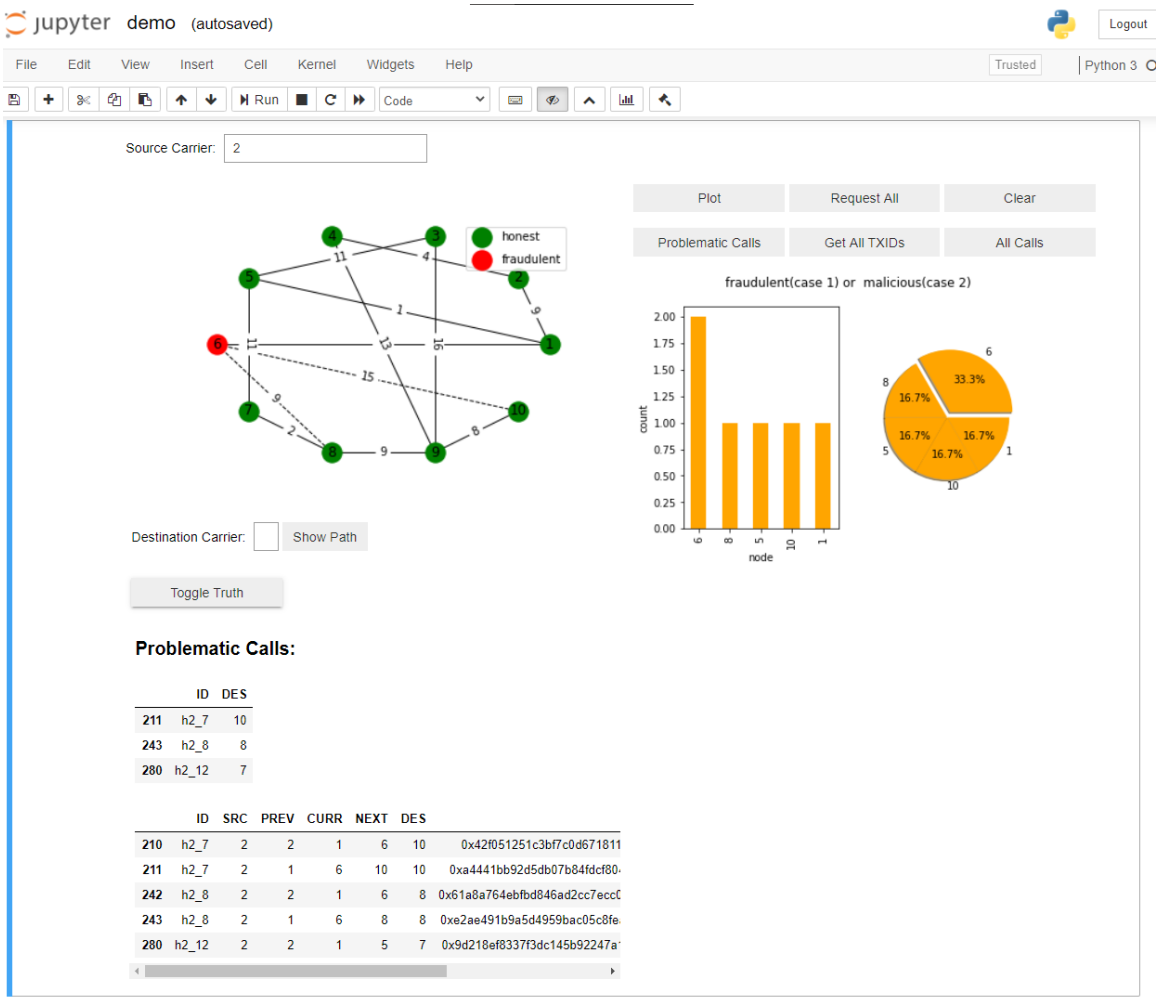


Figure 5.14: Step 3: information on calls that are short-stopped is collected, but it is still unknown to the first carrier where calls were dropped.

`//mininet.org/`), a simulator of virtual networks with Software Defined Networking (SDN) capabilities. The SDN capabilities provide control over the network traffic at the packet level, using OpenFlow commands executed on Open vSwitches (`https://www.openvswitch.org/`). Dispatched IP packets simulate calls. The Open vSwitches represent carriers and different behaviors (honest, fraudulent, sloppy, malicious) are assigned to them. Ryu (`https://ryu.readthedocs.io/`) is used for the SDN controller, to control the traffic flow according to the provided specifications. See illustration of the Mininet control screen in Fig. 5.6. A Network Topology Specification and Configuration (NTSC) was implemented, to control different network parameters,

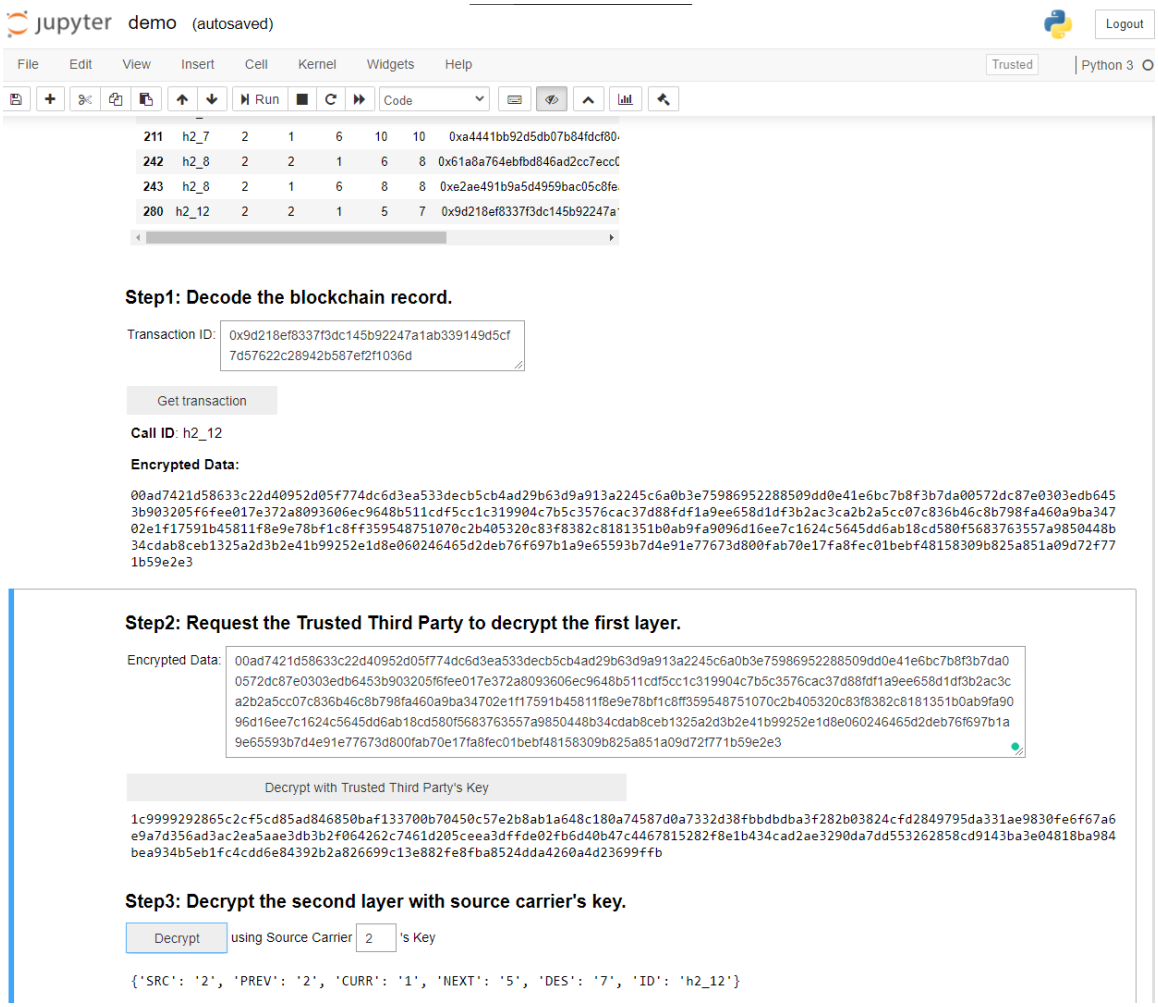


Figure 5.15: Step 4: the blockchain records of a dropped call are decrypted and revealed to the first carrier of that call.

including the network topology. For scalable storage and exploration of the network, a graph database may be used [39].

Each carrier is connected to a blockchain proxy and has an encryption/decryption module. The blockchain proxy manages the connection of the carrier with the blockchain peers and supports access to the blockchain. To facilitate data extraction and evaluation of queries over the information in the blockchain, a PostgreSQL RDBMS is used by each proxy, to manage local data. Note that the blockchain proxies only deliver encrypted data, so the same proxy can serve more than one carrier.

Blockchain and P2P network. The system uses a permissioned version of

Ethereum (`https://ethereum.org/`) with a Proof-of-Authority (PoA) consensus protocol. However, any other blockchain system or any other consensus protocol can be used for the decentralized trusted storage of handoffs. In PoA, a small set of trusted validators create the blocks. In each round, a leader is selected randomly from the set of validators. The leader suggests a block. If the block is approved by the majority of the validators, it is added to the blockchain. Otherwise, the leader is considered malicious and removed from the set of validators. PoA has a high throughput (transaction rate) in comparison to common consensus protocols like proof of work (PoW), however, it is considered less secure than PoW. Improving scalability, e.g., by chain partitioning as suggested in [32], is an ongoing research direction.

Figures 5.7 and 5.8 depict the handoff transactions that four carriers see. As time passes, more handoffs are conducted and the carriers see more transactions. To illustrate that, we present in Figures 5.7 and 5.8 the transactions at two different times. The table of each carrier contains the columns `Cid` and `S->D`, where `Cid` is the call identifier and `S->D` is the pair of source and destination carriers of the call. For example, in the first row of the table of Carrier 2 in Fig. 5.7, `h2_1` is the call id, and `2->10` specifies that the source and destination of the call are carriers 2 and 10. The Proxy tables show transactions that go via the proxy of the Carrier. In the proxy table there are three columns. The `Cid` column is the call identifier, the `S->R` column specifies the sender and receiver nodes of the call, and `P-C-N` refers to the previous, current and next nodes in the path to the destination. The current node appears in orange. For example, the first row of Proxy 2 in Fig. 5.7 contains the call id `h2_1`, the source and destination pair 2 and 10, as `2->10`, and the 3-tuple `s-2-1`, which specifies that the previous node is the sender, the current node is Carrier 2 and the next node is Carrier 1. Note that having the value '`s`' for the previous node refers to the sender user (the caller), and the value '`r`' as the next node refers to the receiver user (the callee).

In Fig. 5.8, the transactions of call `h3_14` are circles. This call is routed from Carrier 3 to Carrier 4, based on the value `3->4` in column `S->R`. Initially, in Proxy 3, we see `s-3-5` for `P-C-N`, which means that from Carrier 3, the call is forwarded to Carrier 5. In Proxy 5 we see `3-5-1`, which means that the previous node was Carrier 3, the current node is Carrier 5 and the next node is Carrier 1. Note that Proxy 3 and Carrier 3 do not see the forwarding of the call from Carrier 5 to Carrier 1. The records `1-2-4` in Proxy 2 and `2-4-r` in Proxy 4 show that the call was forwarded from Carrier 1 to Carrier 4 and from Carrier 4 to the receiver user. All the records together reveal the route `s->3->5->1->2->4->r` of the call, however, each carrier and proxy only see a local view of the handovers they where involved in. In other words, each carrier has a different view of the information on the blockchain because a carrier can only see its own records. Only when the records of a short stopped call are decrypted by the IC, the first carrier of the call can view them.

A blockchain dashboard allows tracking the information on the blockchain, including statistics (Fig. 5.9), detailed information on selected transactions (Fig. 5.10) and the list of transactions in the blockchain (Fig. 5.11).

IC. The IC and the four confidentiality models are implemented in Python. The packages Cryptography and PyCryptodome are used for the cryptographic functions. When CDRs of a call ξ should be revealed, (1) the IC receives the call id from the proxy, (2) it retrieves the records of ξ from the blockchain and decrypts the relevant records, according to the confidentiality model, and (3) the decrypted records are shared with the relevant carriers through the proxy.

5.5 Demonstration

The demonstration will focus on showing two things—usability and confidentiality. Usability requires that in the case of a failed delivery, the records on the blockchain

and the IC decryption are sufficient for revealing the place where the fraud occurred (indicating the last two nodes in the route). The demo will illustrate the usability for different behaviors of carriers (honest, fraudster, sloppy and malicious), under the assumption that most of the carriers are honest, like in the real world.

To illustrate confidentiality, the information that each carrier sees will be presented. We show that the view of each carrier does not lead to a confidentiality breach and carriers see in an unencrypted form only information they are allowed to see according to the model.

Figures 5.12–5.15 present Jupyter Notebook screenshots that depict a demonstrated scenario. These screenshots are updated in real time while the system runs and they provide information about the status of different components. Initially, the network is created based on given parameters, and the behavior of each node (honest, fraudster, sloppy or malicious) is selected. Fig. 5.12 presents a network topology that is created in Step 1, and a case where one of the carriers drops 50% of the calls through it. Initially, the identity and type of behavior of this carrier are not known to the other carriers. The simulation dispatches calls and routes them to the specified destination over the simulated network. Fig. 5.13 and Fig. 5.14 show the information that Carrier 2 sees at two times. This is a limited view of the handovers due to confidentiality. The carrier only sees handovers it is part of. The encrypted handoffs of all the honest carriers are recorded on the blockchain. Fig. 5.15 presents the decryption process and the revealed path of a dropped call.

This demonstration focuses on a specific but large problem, mitigation of IRSF—a problem at the scale of billions of dollars. However, the ability to track delivery failures and frauds in decentralized delivery systems has many additional applications in different supply chains and delivery systems. Our main contribution is showing that tracking can be done in a decentralized system, with limited trust between organizations, while maintaining business confidentiality. Future work includes adding

economic intensives for carriers to record information on the blockchain, and incentives not to misuse the system, e.g., by requiring that carriers would pay for information.

5.6 Conclusion

This demonstration focuses on a specific but large problem, mitigation of IRSF—a problem at the scale of billions of dollars. However, the ability to track delivery failures and frauds in decentralized delivery systems has many additional applications in different supply chains and delivery systems. Our main contribution is showing that tracking can be done in a decentralized system, with limited trust between organizations, while maintaining business confidentiality. Future work includes adding economic intensives for carriers to record information on the blockchain, and incentives not to misuse the system, e.g., by requiring that carriers would pay for information.

Chapter 6

Conclusion

This dissertation has provided an in-depth exploration of the application of blockchain technology in various domains, with a particular focus on medical data storage, deterring illicit activities such as cryptocurrency money laundering, and enhancing privacy through cryptographic schemes.

Our research has significantly contributed to the existing body of knowledge by proposing efficient data storage and retrieval strategies tailored for blockchain systems. These strategies strike a balance between data replication and the performance of data retrieval, enhancing the practicality and feasibility of employing blockchain technology for medical data storage. Consequently, our work contributes to the advancement of secure and efficient data management solutions in the healthcare sector, paving the way for more robust and reliable medical data systems.

In addressing the issue of cryptocurrency money laundering, we designed and implemented mechanisms that monitor transaction traffic, aiming to deter users from engaging in illicit activities. Our work plays a crucial role in creating a safer and more secure digital environment, where the advantages of technological advancements can be fully leveraged without the risk of misuse.

Furthermore, we proposed multiple cryptographic schemes that balance the need

for transparency with privacy and confidentiality. These schemes employ various cryptographic techniques to control and limit access to data specifics to authorized individuals, while maintaining the integrity and traceability of the data. Our contributions in this area enhance the security and privacy of blockchain technology, making it more suitable for a wider range of applications.

Despite the significant insights gained from this research, there are limitations that must be acknowledged. Future studies could further explore the potential of blockchain technology in other domains, and devise more efficient strategies for data storage and retrieval, as well as more robust mechanisms to deter illicit activities.

6.1 Future Directions

Despite the significant insights gained from this research, there are limitations that must be acknowledged. Future studies could further explore the potential of blockchain technology in other domains, such as supply chain management, intellectual property protection, and decentralized finance. Additionally, research could focus on devising more efficient strategies for data storage and retrieval, as well as more robust mechanisms to deter illicit activities. The exploration of new cryptographic techniques and consensus algorithms may also provide avenues for enhancing the scalability, security, and efficiency of blockchain systems.

6.2 Final Remarks

In conclusion, this dissertation has significantly contributed to the field of blockchain technology by proposing efficient strategies for data storage and retrieval, designing mechanisms to deter illicit activities, and enhancing privacy through cryptographic schemes. The findings of this study pave the way for the advancement of secure and efficient data management solutions in various sectors, and contribute to creating a

safer and more secure digital environment.

Bibliography

[1] Bitcoin. URL https://bitcoin.org/en/.

[2] Deloitte's 2018 global blockchain survey. https://www2.deloitte.com/content/dam/Deloitte/cz/Documents/financial-services/cz-2018-deloitte-global-blockchain-survey.pdf. (Accessed on 08/21/2023).

[3] Docker. URL https://www.docker.com/.

[4] Ethereum. URL https://www.ethereum.org/.

[5] Genes-drugs – cpic. https://cpicpgx.org/genes-drugs/. (Accessed on 08/01/2023).

[6] Leveldb. URL https://github.com/google/leveldb.

[7] A python wrapper for multichain json-rpc api. URL https://github.com/DXMarkets/Savoir.

[8] Solidity — solidity 0.8.21 documentation. https://docs.soliditylang.org/en/v0.8.21/#solidity. (Accessed on 08/21/2023).

[9] Table of pharmacogenomic biomarkers in drug labeling — fda. https://www.fda.gov/drugs/science-and-research-drugs/table-pharmacogenomic-biomarkers-drug-labeling. (Accessed on 08/01/2023).

[10] The real story of how the internet became so vulnerable — the washington post. https://www.washingtonpost.com/sf/business/2015/05/30/net-of-insecurity-part-1/. (Accessed on 07/29/2023).

[11] Tracing the trail of the upbit hack - cylynx. https://www.cylynx.io/blog/tracing-the-trail-of-the-upbit-hack/. (Accessed on 08/21/2023).

[12] U.s. treasury issues first-ever sanctions on a virtual currency mixer, targets dprk cyber threats — u.s. department of the treasury. https://home.treasury.gov/news/press-releases/jy0768, . (Accessed on 07/31/2023).

[13] U.s. treasury sanctions notorious virtual currency mixer tornado cash — u.s. department of the treasury. https://home.treasury.gov/news/press-releases/jy0916, . (Accessed on 07/31/2023).

[14] Warfarindosing. http://warfarindosing.org/Source/Home.aspx. (Accessed on 08/01/2023).

[15] Our code at github. URL https://github.com/mshuaic/Blockchain_med.

[16] The chainalysis 2023 crypto crime report. https://go.chainalysis.com/2023-crypto-crime-report.html, 2003. (Accessed on 08/01/2023).

[17] United states v. evans. F.3d, 2018 WL 2926807 (5th Cir.), 2018. Decided June 12, 2018.

[18] Ross Anderson. Making bitcoin legal (transcript of discussion). In *Security Protocols XXVI: 26th International Workshop, Cambridge, UK, March 19–21, 2018, Revised Selected Papers 26*, pages 254–265. Springer, 2018.

[19] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, and Yacov Manevich. Hyperledger fabric: a distributed operating

system for permissioned blockchains. In *Proceedings of the Thirteenth EuroSys Conference*, page 30, 2018.

[20] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, Srinivasan Muralidharan, Chet Murthy, Binh Nguyen, Manish Sethi, Gari Singh, Keith Smith, Alessandro Sorniotti, Chrysoula Stathakopoulou, Marko Vukolić, Sharon Weed Cocco, and Jason Yellick. Hyperledger fabric: A distributed operating system for permissioned blockchains. In *Proceedings of the Thirteenth EuroSys Conference*, EuroSys '18, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450355841. doi: 10.1145/3190508.3190538. URL `https://doi.org/10.1145/3190508.3190538`.

[21] Asaph Azaria, Ariel Ekblaw, Thiago Vieira, and Andrew Lippman. Medrec: Using blockchain for medical data access and permission management. In *2016 2nd international conference on open and big data (OBD)*, pages 25–30. IEEE, 2016.

[22] Arshdeep Bahga and Vijay Madisetti. *Blockchain applications: a hands-on approach*. VPT, 2017.

[23] Dan Baker. International revenue share fraud: are we winning the battle against telecom pirates? *Black Swan Telecom Journal*, 2012.

[24] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 92–111. Springer, 1994.

[25] Alex Biryukov, Dmitry Khovratovich, and Ivan Pustogarov. Deanonymisation of clients in bitcoin p2p network. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 15–29, 2014.

[26] Joseph Bonneau, Arvind Narayanan, Andrew Miller, Jeremy Clark, Joshua A Kroll, and Edward W Felten. Mixcoin: Anonymity for bitcoin with accountable mixes. In *International Conference on Financial Cryptography and Data Security*, pages 486–504. Springer, 2014.

[27] Nicolas Christin. Silk road and the darknet: A new market for illegal drugs. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, pages 189–202. ACM, 2013. doi: 10.1145/2504730.2504747.

[28] Francis S. Collins, Michael Morgan, and Aristides Patrinos. The human genome project: lessons from large-scale biology. *Science*, 300(5617):286–290, 2003.

[29] International HapMap Consortium. The international HapMap project. *Nature*, 426(6968):789, 2003.

[30] Joan Daemen and Vincent Rijmen. *The design of Rijndael: AES-the advanced encryption standard*. Springer Science & Business Media, 2013.

[31] Tamraparni Dasu, Yaron Kanza, and Divesh Srivastava. Geofences in the sky: herding drones with blockchains and 5G. In *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2018.

[32] Tamraparni Dasu, Yaron Kanza, and Divesh Srivastava. Unchain your blockchain. In *Proc. Symposium on Foundations and Applications of Blockchain*, pages 16–23, 2018.

[33] Laura Dean. Warfarin therapy and vkorc1 and cyp genotype. 2018.

[34] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. https://www.torproject.org/, 2004. Online; accessed [07/11/2023].

[35] Tien Tuan Anh Dinh, Ji Wang, Gang Chen, Rui Liu, Beng Chin Ooi, and Kian-Lee Tan. BLOCKBENCH: a framework for analyzing private blockchains. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17. ACM, 2017. ISBN 9781450341974.

[36] Ittay Eyal, Adem Efe Gencer, Emin Gün Sirer, and Robbert Van Renesse. {Bitcoin-NG}: A scalable blockchain protocol. In *13th USENIX symposium on networked systems design and implementation (NSDI 16)*, pages 45–59, 2016.

[37] Andreas Fichtner and Christoph Meinel. Ransomware: A comprehensive review. *Journal of Cybersecurity and Privacy*, 3(1):1–19, 2020. doi: 10.2478/jcsp-2020-0001.

[38] Aris Gkoulalas-Divanis, Grigorios Loukides, and Jimeng Sun. Publishing data from electronic health records while preserving privacy: A survey of algorithms. *Journal of Biomedical Informatics*, 50:4–19, 2014.

[39] Pramod Jamkhedkar, Theodore Johnson, Yaron Kanza, Aman Shaikh, N. K. Shankaranarayanan, and Vladislav Shkapenyuk. A graph database for a virtualized network infrastructure. In *Proceedings of the 2018 ACM International Conference on Management of Data*, SIGMOD '18. ACM, 2018.

[40] Neal Koblitz and Alfred J Menezes. A survey of public-key cryptosystems. *SIAM review*, 46(4):599–634, 2004.

[41] Godfred Yaw Koi-Akrofi, Joyce Koi-Akrofi, Daniel Adjei Odai, and Eric Okyere Twum. Global telecommunications fraud trend analysis. *International Journal of Innovation and Applied Studies*, 25(3):940–947, 2019.

[42] Ahmed Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving

smart contracts. In *2016 IEEE symposium on security and privacy (SP)*, pages 839–858. IEEE, 2016.

[43] Tsung-Ting Kuo, Hyeon-Eui Kim, and Lucila Ohno-Machado. Blockchain distributed ledger technologies for biomedical and health care applications. *Journal of the American Medical Informatics Association*, 24(6):1211–1220, 2017.

[44] Tsung-Ting Kuo, Xiaoqian Jiang, Haixu Tang, XiaoFeng Wang, Tyler Bath, Diyue Bu, Lei Wang, Arif Harmanci, Shaojie Zhang, Degui Zhi, et al. idash secure genome analysis competition 2018: blockchain genomic data access logging, homomorphic encryption on gwas, and dna segment searching, 2020.

[45] Tsung-Ting Kuo, Tyler Bath, Shuaicheng Ma, Nicholas Pattengale, Meng Yang, Yang Cao, Corey M Hudson, Jihoon Kim, Kai Post, Li Xiong, et al. Benchmarking blockchain-based gene-drug interaction data sharing methods: a case study from the idash 2019 secure genome analysis competition blockchain track. *International journal of medical informatics*, 154:104559, 2021.

[46] John Lonsdale, Jeffrey Thomas, Mike Salvatore, Rebecca Phillips, Edmund Lo, Saboor Shad, Richard Hasz, Gary Walters, Fernando Garcia, Nancy Young, Barbara Foster, Mike Moser, Ellen Karasik, Bryan Gillard, Kimberley Ramsey, Susan Sullivan, Jason Bridge, Harold Magazine, John Syron, Johnelle Fleming, Laura Siminoff, Heather Traino, Maghboeba Mosavel, Laura Barker, Scott Jewell, Dan Rohrer, Dan Maxim, Dana Filkins, Philip Harbach, Eddie Cortadillo, Bree Berghuis, Lisa Turner, Eric Hudson, Kristin Feenstra, Leslie Sobin, James Robb, Phillip Branton, Greg Korzeniewski, Charles Shive, David Tabor, Liqun Qi, Kevin Groch, Sreenath Nampally, Steve Buia, Angela Zimmerman, Anna Smith, Robin Burges, Karna Robinson, Kim Valentino, Deborah Bradbury, Mark Cosentino, Norma Diaz-Mayoral, Mary Kennedy, Theresa Engel, Penelope Williams, Kenyon Erickson, Kristin Ardlie, Wendy Winckler, Gad Getz, David DeLuca, Daniel

MacArthur, Manolis Kellis, Alexander Thomson, Taylor Young, Ellen Gelfand, Molly Donovan, Yan Meng, George Grant, Deborah Mash, Yvonne Marcus, Margaret Basile, Jun Liu, Jun Zhu, Zhidong Tu, Nancy J. Cox, Dan L. Nicolae, Eric R. Gamazon, Hae Kyung Im, Anuar Konkashbaev, Jonathan Pritchard, Matthew Stevens, Timothèe Flutre, Xiaoquan Wen, Emmanouil T. Dermitzakis, Tuuli Lappalainen, Roderic Guigo, Jean Monlong, Michael Sammeth, Daphne Koller, Alexis Battle, Sara Mostafavi, Mark McCarthy, Manual Rivas, Julian Maller, Ivan Rusyn, Andrew Nobel, Fred Wright, Andrey Shabalin, Mike Feolo, Nataliya Sharopova, Anne Sturcke, Justin Paschal, James M. Anderson, Elizabeth L. Wilder, Leslie K. Derr, Eric D. Green, Jeffery P. Struewing, Gary Temple, Simona Volpi, Joy T. Boyer, Elizabeth J. Thomson, Mark S. Guyer, Cathy Ng, Assya Abdallah, Deborah Colantuoni, Thomas R. Insel, Susan E. Koester, A. Roger Little, Patrick K. Bender, Thomas Lehner, Yin Yao, Carolyn C. Compton, Jimmie B. Vaught, Sherilyn Sawyer, Nicole C. Lockhart, Joanne Demchok, and Helen F. Moore. The genotype-tissue expression (GTEx) project. *Nature Genetics*, 45: 580–585, 2013.

[47] Shuaicheng Ma, Yang Cao, and Li Xiong. Efficient logging and querying for blockchain-based cross-site genomic dataset access audit. *BMC medical genomics*, 13(7):1–13, 2020.

[48] Shuaicheng Ma, Yang Cao, and Li Xiong. Transparent contribution evaluation for secure federated learning on blockchain. In *2021 IEEE 37th International Conference on Data Engineering Workshops (ICDEW)*, pages 88–91, 2021. doi: 10.1109/ICDEW53142.2021.00023.

[49] Shuaicheng Ma, Tamraparni Dasu, Yaron Kanza, Divesh Srivastava, and Li Xiong. Fraud buster: Tracking irsf using blockchain while protecting business confidentiality. In *CIDR*, 2021.

[50] Tim K Mackey, Tsung-Ting Kuo, Basker Gummadi, Kevin A Clauson, George Church, Dennis Grishin, Kamal Obbad, Robert Barkovich, and Maria Palombini. 'fit-for-purpose?'–challenges and opportunities for applications of blockchain technology in the future of healthcare. *BMC medicine*, 17(1):1–17, 2019.

[51] Bradley A. Malin, Khaled El Emam, and Christine M. O'Keefe. Biomedical data privacy: problems, perspectives, and recent advances. *Journal of the American Medical Informatics Association*, 20(1):2–6, 2013.

[52] Nicholas Mamo, Gillian M Martin, Maria Desira, Bridget Ellul, and Jean-Paul Ebejer. Dwarna: a blockchain solution for dynamic consent in biobanking. *European Journal of Human Genetics*, 28(5):609–626, 2020.

[53] Thomas McGhin, Kim-Kwang Raymond Choo, Charles Zhechao Liu, and Debiao He. Blockchain in healthcare applications: Research challenges and opportunities. *Journal of Network and Computer Applications*, 135:62–75, 2019.

[54] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M Voelker, and Stefan Savage. A fistful of bitcoins: characterizing payments among men with no names. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 127–140, 2013.

[55] Ian Miers, Christina Garman, Matthew Green, and Aviel D Rubin. Zerocoin: Anonymous distributed e-cash from bitcoin. In *2013 IEEE Symposium on Security and Privacy*, pages 397–411. IEEE, 2013.

[56] P. Mockapetris. Domain names - concepts and facilities. Technical report, RFC 882, Internet Engineering Task Force, 1983.

[57] Daniel Moore and Thomas Rid. Cryptopolitik and the darknet. *Survival*, 58(1): 7–38, 2016.

[58] Massimo Morini. From'blockchain hype'to a real business case for financial markets. *Available at SSRN 2760184*, 2016.

[59] Malte Moser. Anonymity of bitcoin transactions. 2013.

[60] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Bitcoin.org*, 2008. URL `https://bitcoin.org/bitcoin.pdf`. Online; accessed [07/11/2023].

[61] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot, 2019.

[62] Muhammad Naveed, Erman Ayday, Ellen W. Clayton, Jacques Fellay, Carl A. Gunter, Jean-Pierre Hubaux, Bradley A. Malin, and Xiaofeng Wang. Privacy in the genomic era. *ACM Comput. Surv.*, 48(1):6:1–6:44, 2015.

[63] Charles Noyes. Bitav: Fast anti-malware by distributed blockchain consensus and feedforward scanning. *arXiv preprint arXiv:1601.01405*, 2016.

[64] Wouter Penard and Tim van Werkhoven. On the secure hash algorithm family. *Cryptography in Context*, pages 1–18, 2008.

[65] Gareth W Peters and Efstathios Panayi. Understanding modern banking ledgers through blockchain technologies: Future of transaction processing and smart contracts on the internet of money. In *Banking beyond banks and money*, pages 239–278. Springer, 2016.

[66] Dian Rachmawati, JT Tarigan, and ABC Ginting. A comparative study of Message Digest 5 (MD5) and SHA256 algorithm. In *Journal of Physics: Conference Series*, volume 978. IOP Publishing, 2018.

[67] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[68] Alex Roehrs, Cristiano André da Costa, Rodrigo da Rosa Righi, Valter Ferreira da Silva, José Roberto Goldim, and Douglas C Schmidt. Analyzing the performance of a blockchain-based personal health record implementation. *Journal of biomedical informatics*, 92:103140, 2019.

[69] Drew Roselli and Thomas E Anderson. Characteristics of file system workloads. 1998.

[70] Merve Sahin, Aurélien Francillon, Payas Gupta, and Mustaque Ahamad. Sok: Fraud in telephony networks. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 235–250. IEEE, 2017.

[71] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, November 1979. ISSN 0001-0782. doi: 10.1145/359168.359176. URL `https://doi.org/10.1145/359168.359176`.

[72] Rakesh Shrestha, Rojeena Bajracharya, and Seung Yeob Nam. Blockchain-based message dissemination in vanet. In *2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS)*, pages 161–166. IEEE, 2018.

[73] John Smith. *Asset Tracing in the Common Law*. Legal Publishing House, 2010. See discussion of Clayton's Case (Devaynes v Noble, UK, 1816).

[74] John Smith. *The Bankruptcy of Lehman Brothers: An Insider's View*. Financial Publishing House, 2010.

[75] John Smith and Emily Johnson. Deepfake: A comprehensive survey. *Journal of Artificial Intelligence*, 10(2):123–145, 2021. doi: 10.1234/ai.2021.10.2.123.

[76] Sebastian Stein, Matthew Palmer, Nicolas Gerber, and Alice Hutchings. Ransomware: An exploratory study of the negotiation process. *International Journal of Information Security*, 17(3):301–317, 2018. doi: 10.1007/s10207-017-0362-8.

[77] Patrick Sylim, Fang Liu, Alvin Marcelo, Paul Fontelo, et al. Blockchain technology for detecting falsified and substandard drugs in distribution: pharmaceutical supply chain intervention. *JMIR research protocols*, 7(9):e10163, 2018.

[78] Tin Tironsakkul, Manuel Maarek, Andrea Eross, and Mike Just. Probing the mystery of cryptocurrency theft: An investigation into methods for taint analysis. *arXiv preprint arXiv:1906.05754*, 2019.

[79] Ray Tomlinson. The first email system. *IEEE Annals of the History of Computing*, 22(2):41–43, 2000.

[80] Sarah Underwood. Blockchain beyond bitcoin. *Commun. ACM*, 59(11):15–17, October 2016. ISSN 0001-0782. doi: 10.1145/2994581. URL `https://doi.org/10.1145/2994581`.

[81] Jelle van den Hooff, M Frans Kaashoek, and Nickolai Zeldovich. Versum: Verifiable computations over large public logs. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 1304–1316, 2014.

[82] Anuraag A Vazirani, Odhran O'Donoghue, David Brindley, and Edward Meinert. Blockchain vehicles for efficient medical record management. *NPJ digital medicine*, 3(1):1, 2020.

[83] Kris A Wetterstrand. Dna sequencing costs: data from the nhgri genome sequencing program (gsp). `www.genome.gov/sequencingcostsdata`. Accessed: 2020-06-01.

[84] Gavin Wood. Ethereum: A secure decentralised generalised transaction ledger. `https://ethereum.github.io/yellowpaper/paper.pdf`, 2014.

[85] Mike Wu, Will McTighe, Kaili Wang, Istvan A Seres, Nick Bax, Manuel Puebla, Mariano Mendez, Federico Carrone, Tomás De Mattey, Herman O Demaestri,

et al. Tutela: An open-source tool for assessing user-privacy on ethereum and tornado cash. *arXiv preprint arXiv:2201.06811*, 2022.

[86] Anatoly Yakovenko. Solana: A new architecture for a high performance blockchain v0. 8.13. *Whitepaper*, 2018.

[87] Peter Yeoh. Regulatory issues in blockchain technology. *Journal of Financial Regulation and Compliance*, 2017.

[88] Xiao Yue, Huiju Wang, Dawei Jin, Mingqiang Li, and Wei Jiang. Healthcare data gateways: found healthcare intelligence on blockchain with novel privacy risk control. *Journal of medical systems*, 40:1–8, 2016.