DIFFERENTIAL PRIVACY IN FINANCIAL DISTRIBUTED LEDGER APPLICATIONS

by
MERVE CAN KUŞ

Submitted to the Graduate School of Engineering
and Natural Sciences in partial fulfilment of
the requirements for the degree of Doctor of Philosophy

Sabancı University
December 2022

ABSTRACT

DIFFERENTIAL PRIVACY IN FINANCIAL DISTRIBUTED LEDGER APPLICATIONS

MERVE CAN KUŞ

COMPUTER SCIENCE AND ENGINEERING Ph.D. DISSERTATION, DECEMBER 2022

Dissertation Supervisor: Prof. Albert Levi

Keywords: Bitcoin, blockchain, distributed ledger, differential privacy, smart metering.

Bitcoin is the pioneering financial distributed ledger system, which captivated researchers with its innovative public blockchain structure. Examinations of this public blockchain resulted in many proposals for improvement in terms of anonymity and privacy. Generally used methods include mixing protocols, ring signatures, zero-knowledge proofs, homomorphic commitments, and off-chain storage systems. On the other hand, differential privacy is a privacy notion coming up with mechanisms that enable running statistical queries without leaking any private information. To the best of our knowledge, in the literature, there is no study examining Bitcoin's public blockchain in terms of differential privacy. However, public blockchain structure can benefit from differential privacy mechanisms for improved privacy, by hiding actual values, and preserving checkability of the integrity of the blockchain. In this dissertation, first, we provide a theoretical examination of differential privacy in Bitcoin public blockchain. We examine the current Bitcoin blockchain structure using the differential privacy formulation. Then, we present feasibility of utilization of two differential privacy mechanisms to be applied to the blockchain of Bitcoin: (i) noise addition to the transaction amounts, and (ii) user graph perturbation. Moreover, we implement noise addition to the transaction amounts by using a public software library. We compare four differential privacy mechanisms using varying parameter values in order to determine the feasible ones. As another contribution of this dissertation, we propose a blockchain-based differentially-private federated smart utility metering framework. We utilize noise addition approach to hide the actual utility consumptions while providing fair settlement among the clients and the utility providers. To sum up, in this dissertation we show that noise addition and graph

perturbation methods decrease the fraction of the cases violating differential privacy. Therefore, they can be used for improving privacy in financial distributed ledger applications.

ÖZET

FİNANSAL DAĞITIK DEFTER UYGULAMALARINDA
DİFERANSİYEL MAHREMİYET

MERVE CAN KUŞ

PROGRAM ADI DOKTORA TEZİ, ARALIK 2022

Tez Danışmanı: Prof. Dr. Albert Levi

Anahtar Kelimeler: Bitcoin, blok zinciri, dağıtık defter, diferansiyel mahremiyet, akıllı ölçüm.

Bitcoin, yenilikçi ve açık blok zinciri yapısıyla araştırmacıları büyüleyen öncü finansal dağıtık defter sistemidir. Bu açık dağıtık defterin incelemesi ile anonimlik ve mahremiyet açısından karıştırma protokolleri, halka imzalar, sıfır bilgi kanıtları, homomorfik taahhütler ve zincir dışı depolama sistemlerini kullanan pek çok iyileştirme önerisi yapılmıştır. Diğer yandan diferansiyel mahremiyet, mahrem bilgi sızdırmadan istatistiksel sorgulamaların yapılmasını sağlayan mekanizmalar ile ortaya çıkan bir gizlilik kavramıdır. Bildiğimiz kadarıyla literatürde diferansiyel mahremiyet açısından Bitcoin'in açık defterini inceleyen bir çalışma yoktur. Bununla birlikte, açık blok zinciri yapısı, gerçek değerleri gizleyecek ve dağıtık defter bütünlüğünün kontrol edilebilirliğini koruyacak diferansiyel mahremiyet mekanizmalarından yararlanabilir. Bu tezde öncelikle, Bitcoin açık blok zinciri için diferansiyel mahremiyetin teorik bir incelemesi sunulmaktadır. Diferansiyel gizlilik formülasyonu kullanılarak mevcut Bitcoin blok zinciri yapısı incelenmektedir. Ardından, Bitcoin blok zincirinde uygulamak için iki farklı gizlilik mekanizmasının fizibilitesi sunulmaktadır: (i) işlem miktarlarına gürültü eklenmesi, ve (ii) kullanıcı grafiğinin pertürbasyonu. Ayrıca, bir açık yazılım kütüphanesi kullanılarak işlem miktarlarına gürültü ekleme uygulanmıştır. Uygulanabilir mekanizmalar ile parametreleri tespit edebilmek için değişken parametre değerleri için dört farklı gizlilik mekanizmasının karşılaştırması yapılmıştır. Bu tezin diğer bir katkısı olarak blok zinciri tabanlı diferansiyel mahremiyeti sağlayan federe bir akıllı hizmet ölçüm çerçevesi önerilmektedir. Müşteriler ve hizmet sağlayıcılar arasında adil bir çözüm sunarken gerçek hizmet tüketimini gizlemek için gürültü ekleme yaklaşımı

kullanılmaktadır. Özetle bu tezde, gürültü ekleme ve kullanıcı grafiğinin pertürbasyonu yöntemlerinin diferansiyel mahremiyeti ihlal eden vaka oranını azalttığı gösterilmektedir. Dolayısıyla finansal dağıtık defter uygulamalarında mahremiyeti geliştirmek için kullanılabilecekleri önerilmektedir.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my dissertation advisor, Prof. Albert Levi, for his worthwhile guidance, continuous support, and invaluable patience throughout my Ph.D. studies. It has been a privilege to study under his guidance. I would also like to thank my dissertation committee members, Prof. Erkay Savaş and Prof. Cem Güneri, for their valuable feedback and contributions. I am also thankful to the remaining members of my dissertation jury, Assoc. Prof. Ali İnan and Assoc. Prof. Muhammed Ali Bingöl, for reviewing my dissertation and providing valuable suggestions.

I am deeply grateful to all of my instructors in Sabancı University, who have provided their immense knowledge and plentiful experience. Special thanks to all my friends from Sabancı University, who supported me with their invaluable companionship and experience.

I am grateful to Kuveyt Turk, Kuveyt Turk Research & Development Center and Architecht for supporting and rewarding my studies. I appreciate support and encouragements of my colleagues throughout my studies.

I am deeply indebted to my family, who did not spare their physical and moral support throughout my studies. I would have never made it without them. And my beloved daughter Mina who was born during my Ph. D. studies, I owe her a debt of gratitude for her tremendous understanding and patience.

I would also like to thank all my friends, who have unwavering support and belief in me.

To my beloved
family and my little Mina

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATONS

P2P Peer-to-Peer 2
TOR The Onion Router 9
UTXO Unspent Transaction Output 12
ECDSA Elliptic Curve Digital Signature Algorithm 12
PoW Proof-of-Work 15
DoS Denial-of-Service 15
ISP Internet Service Provider 21
SMC Secure Multiparty Computation 21
AMI Advanced Metering Infrastructures 23
SM Smart Meter 23
HAN Home Area Network 23
WAN Wide Area Network 23
ZK-SNARK Zero Knowledge Succinct Non-interactive ARguments of Knowledge 25
SNARK Succinct Non-interactive ARgument of Knowledge 25
I2P The Invisible Internet Project 37
TRR Transaction Remote Release 37
ECC Elliptic Curve Cryptography 40
DC-net Dining Cryptographers Network 41
XOR Exclusive Or 41
ZKCP Zero Knowledge Contingent Payments 43
CT Confidential Transactions 44
NIZKP Non-Interactive Zero-Knowledge Proof 46
MAE Mean Absolute Error 49
MRO Mean Ranking Offset 82
PoS Proof-of-Stake 93

LIST OF SYMBOLS

ε	Differential privacy parameter
$\mathrm{P}[x]$	Probability of x
$S(f)$	Sensitivity of function f
D_1, D_2	Databases differing in a single row
S	Subset
δ	Differential privacy parameter for Gaussian and Uniform mechanisms
F	Functions used for differential privacy evaluation
n	Number of Bitcoin transactions or measurement periods
A_1, A_2	Bitcoin addresses
a	Bitcoin transaction amount
ce	Actual consumption measurement value for electricity
cw	Actual consumption measurement value for water
cg	Actual consumption measurement value for gas
ne	Noise value added for electricity
nw	Noise value added for water
ng	Noise value added for gas
nce	Noisy consumption measurement value for electricity
ncw	Noisy consumption measurement value for water
ncg	Noisy consumption measurement value for gas
ane	Aggregate noise value for electricity
anw	Aggregate noise value for water
ang	Aggregate noise value for gas
r_e	Rate of electricity utility
r_w	Rate of water utility
r_g	Rate of gas utility
SH	List of smart homes

E	Electricity utility provider
W	Water utility provider
G	Gas utility provider
be	Electricity utility bill amount
bw	Water utility bill amount
bg	Gas utility bill amount
U	Utility list
P	List of utilities that have positive value for the sum of aggregate noises multiplied by the utility rate
N	List of utilities that have negative value for the sum of aggregate noises multiplied by the utility rate

1. INTRODUCTION

Payments, money transfers and commerce are widely preferred to be made on the Internet for a long while, especially in the last decade, as with many other things in this digital age. This preference comes from the speed, which is provided by the digitalization and needed in busy daily life, as well as increasing global connectivity with the rise of digital businesses and social networks. Commerce on the Internet is done assured that financial institutions and banks serve as trusted authorities. This model can be called trust-based; buyers and merchants may not trust each other; however, they trust well-known banks and banks act as trust entities managing transactions and keeping records. However, there are some disadvantages with this trust-based model (Nakamoto, 2008). First, financial institutions act as mediators between merchants and buyers, and there exists a cost for mediation. This limits the minimum practical transaction size. Second, there is a possibility of reversal of transactions. Transactions can be reversed by banks if there is a dispute between the trading parties, e.g., the buyer transfers the money, but the seller does not send goods or provide services to the buyer. However, this possibility of reversal compels that merchants get information about their customers. On the contrary, merchants do not have to get extra information about their customers like billing address, name, etc. when transactions are irreversible. In addition, irreversible transactions protect merchants from *chargeback fraud*, i.e., if a dishonest buyer says that he did not make the purchase. If dishonest merchants are considered, using escrow services may be a method for protecting buyers in the case of irreversible transactions. Finally, especially international transactions in the regular banking, e.g., money transfers, are slow due to procedural delays.

An electronic payment system, which is not based on trust, can be realized using cryptographic mechanisms. Digital cash concept, which utilizes cryptography, was first introduced by Chaum (1983) in 1982 and evolved from trust-based model to decentralized networks in the subsequent decades. In a decentralized system, where there is not any

trusted authority, parties, also called peers, transact directly with each other forming a Peer-to-Peer (P2P) network. This kind of digital cash systems, which use virtual assets and utilize cryptography, are also called cryptocurrency. Bitcoin (Nakamoto, 2008) which is introduced by Satoshi Nakamoto in 2008, is the first digital currency with a decentralized ledger. Although a huge number of alternative proposals emerged following the ideas of Bitcoin, Bitcoin is still the most widely accepted and used one. There can be found a total number of 8958 cryptocurrencies listed in CoinMarketCap web site (https://coinmarketcap.com) with $856 billion total market cap, where the number of circulating bitcoins exceeded 19.2 million, and the total Bitcoin market cap exceeded $330 billion, resulting Bitcoin dominance over 38% as of December 2022.

Bitcoin is defined as “an innovative payment network and a new kind of money” in Bitcoin website (https://bitcoin.org). Essentially, it is an open source P2P money. In Bitcoin, transactions are recorded in a publicly distributed ledger, which is called *blockchain*. The base unit of account is *bitcoin*, and the lowest-valued unit is *satoshi*. There is no central authority or a bank, managing and verifying transactions. These operations and issuing of bitcoins are performed collectively by the network, which consists of communicating nodes (peers) running Bitcoin software. Blockchain structure provides a single and shared history for all users, which also provides integrity. Issuing bitcoins is achieved through *mining* process. Mining is the activity of adding transaction records to the blockchain. Users spend their computing power to verify and record payments; in return, they earn bitcoins, which are created as the result of this payment processing work as a reward. Bitcoins can also be exchanged for other currencies or used for buying products and services. Transactions are computationally impractical to reverse, and these non-reversible transactions protect sellers from fraud.

Besides, although Bitcoin has emerged to be used in the financial sector, its blockchain structure, which is a distributed ledger, and its P2P network attracted the academic community, as well. This attraction resulted in numerous studies on blockchain taking place in the literature (Monrat et al., 2019; Bhutta et al., 2021; Abbas & Sung-Bong, 2019; Dave et al., 2019). Smart metering is one of the areas that blockchain is utilized. Smart metering is utility metering becoming intelligent in the digital new world. Smart meters transmit utility consumptions at the end of every predefined measurement period during a billing period. Clients utilizing smart meters can observe their up-to-date consumptions, and have more frequent and transparent information about their utility consumptions.

With the help of smart metering, manual measurement reading of utility consumption process is automated. Smart metering also has advantages like efficient management, control, and operation of utility and allowing integration of utility trading platforms or renewable energy sources (Aklilu & Ding, 2022). Smart metering systems can utilize blockchain infrastructures with the motivations like securing consumption data (Bokhari et al., 2019), increasing transparency, trust, and democracy among all the entities, monitoring more efficiently, having a fault-tolerant network, preventing unauthorized modifications to data (Mollah et al., 2020).

On the other hand, differential privacy, which was proposed in 2006 (Dwork et al., 2006), is a privacy notion that is related to the distinguishability of the presence/absence of an element in a dataset via query functions. A mechanism is differentially private if this distinguishability is below some threshold. There are methods for providing differential privacy, and these methods can be used for improving privacy. Perturbing data with added noise is a way of providing differential privacy, and this method is used for sharing private data for analysis purposes instead of sharing real data. For instance, in order to ensure differential privacy, data from a health database is shared with researchers under certain rules, e.g., a certain number of queries are allowed, and the actual data is perturbed with the addition of noise. This approach provides *global* differential privacy since the addition of the noise is done after the data aggregation. Differential privacy can be achieved *locally*, as well. In this approach, noise is added before data is aggregated to a database. This local approach is utilized by Apple for collecting data from devices as given in *Differential Privacy Overview*, (https://www.apple.com/privacy/docs/Differential_Privacy_Overview.pdf) and by Google for collecting data from Chrome web browsers (Bittau et al., 2017). There is a trade-off between privacy and data utility. Adding more noise improves privacy, but it also decreases data utility. This trade-off is formally controlled using a parameter called epsilon (ε). As ε gets smaller, the amount of noise increases, resulting in improved privacy and decreased utility. There are many studies utilizing differential privacy approaches in different areas, some examples include messaging, health, scheduling for ridesharing, artificial intelligence, deep learning, and software defect prediction (Dazar et al., 2018; Dankar & Emam, 2012; Tong et al., 2017; Zhu & Yu, 2019; Chen et al., 2019; Abadi et al., 2016).

1.1 Motivation

To use Bitcoin, users are not required to provide real names. Instead, pseudonyms are used, so it is explicitly seen that some entities transact with each other, but the real identities stay hidden like in stock exchange operations. However, since all transactions are publicly available, activities of the users can be tracked and linked. Therefore, profile of the users can be extracted, and the user identities can be revealed by linking one of the transactions to off-network information, as clearly shown in the previous studies that analyze anonymity and privacy in Bitcoin. Therefore, users cannot stay completely anonymous, and user privacy is not provided since amount values, sender and receiver user addresses are explicitly visible in the blockchain. This shortcoming, which introduces the possibility of tracing, results in, for example, spending history of a user becoming available and accessible to all other people, or cash flow of merchants becoming exposed to their competitors. For instance, with the knowledge that someone shopped online for 0.000381 BTC from a well-known e-commerce site, Bitcoin addresses that made a 0.000381 BTC valued shopping can be found by querying the Bitcoin address of the site and the transaction amounts equal to 0.000381 from the blockchain. Consequently, room for research came up for anonymity and privacy improvement in Bitcoin, and many academic papers have been published (Kus-Khalilov & Levi, 2018; Conti et al., 2018; Amarasinghe et al., 2019; Venkatakrishnan et al., 2017; Zhu et al., 2020). In these studies, generally used methods for anonymity and privacy improvement include mixing protocols, ring signatures, zero-knowledge proof, homomorphic commitments, and off-chain storage systems. Some of these studies are implemented, for example, Monero (https://www.getmonero.org) using ring signatures, and Zcash (https://z.cash) using zero-knowledge proofs are two of the prominent privacy improving cryptocurrencies.

While researchers are exploring new ways to improve anonymity and privacy in blockchain-based cryptocurrencies, taking extra measures for improving anonymity and privacy complicates checking the integrity of the system. This complication is due to the use of public Bitcoin addresses and transaction amounts to check the integrity of the system. For instance, when the transaction amounts are hidden using a cryptographic approach, the total number of coins in the system cannot be counted, and if someone

breaks the system, he can issue coins without being detected. Similarly, when the links between transactions are broken using cryptography, the flow of bitcoins cannot be tracked. Considering these, we hypothesize that the Bitcoin blockchain may benefit from differential privacy, which will not affect the checkability of the integrity of the system. Hiding actual transaction amounts by adding noise can be a way of applying differential privacy. Our motivation for this approach also arises from the fact that perturbing actual data with noise makes anonymization and privacy breaches by direct queries impossible. For instance, in the previously mentioned scenario with 0.000381 BTC valued shopping from a well-known e-commerce site, if some noises are added to the transaction amounts while adding them to the blockchain, a value of 0.000381 would be updated as 0.000383 or 0.000377. Therefore, the detection of these shoppers would be prevented by direct queries. Moreover, there would be no guarantee that the closest value to 0.000381 corresponds to the related transaction. To the best of our knowledge, in the literature, there is no study examining Bitcoin's public blockchain in terms of differential privacy. However, public blockchain structure can benefit from differential privacy mechanisms for improved privacy, by hiding actual values, and preserving checkability of the integrity of the blockchain.

On the other hand, despite the benefits of smart metering, privacy concerns are raised when utility metering becomes smart. Illegal activities to a house can be planned by estimating the routine of the household living in that house, or a utility can use consumption values of its clients unethically, e.g., selling consumption data to businesses that do targeted advertisements (Hassan et al., 2019). When smart metering is used with blockchain, even though blockchain comes with its advantages, there are again privacy concerns since data kept in distributed ledgers which may remain public. Therefore, we focus our study on combining smart metering with blockchain and differential privacy mechanisms to improve privacy. There are many studies combining smart metering with blockchain (Mollah et al., 2020; Andoni et al., 2019; Guo et al., 2022), or smart metering with differential privacy (Farokhi, 2020; Hassan et al., 2020a; Marks et al., 2021). However, there are few studies combining smart metering with both blockchain and differential privacy. Although utilization of differential privacy and blockchain with smart metering is investigated in (Hassan et al., 2020b), many issues were remained abstract, e.g., noise addition is done for hiding actual consumption values, however reconciliation process for billing according to actual consumption values is not

considered. (Gai et al., 2019) improved privacy by adding dummy accounts to hide distribution and trends of utility consumptions instead of adding noise. They state that adding noise would not hide distribution and trends, however it is worth examining in terms of differential privacy theoretically to confirm these arguments.

1.2 Contributions

In this dissertation, first, we provide a theoretical examination of differential privacy in Bitcoin public blockchain. We examine the current Bitcoin blockchain structure using the differential privacy formulation. Then, we present feasibility of utilization of two differential privacy mechanisms to be applied to the blockchain of Bitcoin: (i) noise addition to the transaction amounts, and (ii) user graph perturbation. Moreover, we implement noise addition to the transaction amounts by using a public software library. We compare four differential privacy mechanisms using varying parameter values in order to determine the feasible ones.

As another contribution of this dissertation, utilization of differential privacy in smart utility metering is investigated. In addition to differential privacy, we leverage blockchain for achieving federation of smart homes and different utility providers, i.e., electricity, water and gas. As a result, we propose a blockchain-based differentially-private federated smart utility metering framework which hides actual consumptions with added noise while providing fair settlement among the clients and the utility providers.

We summarize the main contributions of this dissertation as follows:

- We provide a theoretical examination of Bitcoin public blockchain from differential privacy perspective,
- We investigate and apply noise addition and user graph perturbation methods for improving differential privacy in Bitcoin public blockchain,
- We propose a blockchain-based differentially-private federated smart utility framework by utilizing noise addition approach.

At the time of this writing, two articles have been published out of this dissertation. In our first article (Kus-Khalilov & Levi, 2018), we presented a comprehensive survey and

detailed investigation of anonymity and privacy in Bitcoin-like digital cash systems. We use writings from this article specifically in Section 2.1, 2.2, 2.3, and 3.1. Our second article (Kus & Levi, 2022) investigated and applied differential privacy in Bitcoin public blockchain. We use writings from this article specifically in Section 2.1, 3.2, and 4.

1.3 Organization

The organization of this dissertation is as follows: The related background information is given in Section 2, including briefings on anonymity and privacy, Bitcoin and blockchain, smart utility metering, and anonymity and privacy issues of Bitcoin, blockchain and smart utility metering. In Section 3, we review the literature on anonymity and privacy in Bitcoin-like digital cash systems and application of differential privacy to financial distributed ledger applications. Section 4 provides investigation and application of differential privacy in Bitcoin. Here, we theoretically examine Bitcoin from differential privacy perspective. Then, we present feasibility of two differential privacy improving mechanisms; (i) utilization of noise addition to Bitcoin transaction amounts, and (ii) user graph perturbation in Bitcoin. In this section, we also provide an empirical study on noise addition to Bitcoin transaction amounts. In Section 5, we propose a blockchain-based differentially-private federated smart utility metering framework. We provide requirements, design, information leakage and differential privacy analysis, and future research ideas for the framework. Finally, Section 6 concludes this dissertation.

2. BACKGROUND

The background information for this study is given in the following five subsections; the first two subsections give information about anonymity, privacy, Bitcoin and blockchain. Then, anonymity and privacy in Bitcoin and blockchain are explained in the third subsection. The last two subsections provide brief information about smart utility metering and privacy in smart utility metering.

2.1 Anonymity and Privacy

Anonymity and privacy are two concepts for which telling the difference may be difficult as Bradbury (Bradbury, 2014) mentioned, where privacy is hiding the context, and anonymity means hiding the owner of it. In daily life, generally, user privacy is sought more than anonymity, since personal data needs to be protected for proper usage. For instance, ownership information of a personal e-mail account can be known by everyone, but the content is restricted, protected and can be accessed by only the account owner using a password. Privacy is also essential in most systems and applications (Eckoff & Wagner, 2017; Xiao & Xiao, 2013; Ferrag et al., 2017). On the other side, anonymity is maybe the most important property that the criminals seek. The actions of criminals become usually public, but the actor aims his identity to remain unknown. With anonymity, holding someone accountable for an action becomes impossible (Davenport, 2002). However, there are some cases where anonymity is desired in daily life too. An example may be the applications, which are practiced in companies occasionally for the workplace evaluation. In these applications, personal opinions on a topic are gathered without identity information at an out-of-sight place. Then, the opinions are consolidated

and announced publicly. Another well-known example is voting in free elections (secret ballot).

For anonymity, the objective is being unidentifiable and untraceable (Kelly et al., 2012). Ensuring true anonymity is difficult. Many applications claiming to be anonymous occur to have flaws, which leaks identity information. Mixing services (Chaum, 1981), which are also called mixing networks (mixnets) or laundry services, are used for preventing tracing activities of messages through a network by including a sequence of intermediaries or a pool structure. However, they cause computation and communication overheads (Chaum et al.., 2016) or may be unreliable. Also, anonymization services, which use onion routing (Reed et al., 1998), are widely employed for hiding identity by addressing the issue of IP tracking. Even The Onion Router (TOR) (Dingledine et al., 2004), which is one of the most successful anonymity networks, is known to have flaws (Bradbury, 2014; Erdin et al., 2015). Besides, these kinds of mixing services may be blocked by some websites or applications, so they are not always utilizable. Hong et al. (2018) proposed a de-mixing algorithm for Bitcoin mixing services, as well.

One of the most dominant factors that prevent true anonymity is meta-data. In systems that consist of electronic transactions, meta-data of the transactions, e.g. log data, may lead to identities when handled with an analytical and holistic approach. For instance, IP addresses or timing of transactions are the data which can be utilized. One well-known example for this case is AOL releasing an “anonymized” search history for researchers, which then caused unexpected and undesirable results as researchers could find out the identity of individuals. One disclosed identity was Thelma Arnold divulged with her research history which shows her personal interests (Bradbury, 2014).

Anonymity and privacy usually come with a price. On the one hand, in general, systems which aim to provide anonymity and privacy require more resources in space, time or computational power, since extra work is done. One the other hand, the users need to pay more to become anonymous and private. For instance, the mobile applications that bring drivers and passengers together, i.e., Uber and similar applications following it, provide cheap riding service compared to taxicabs. However, users need to reveal their identities to use these applications, since the driver and the passenger rate each other about the ride and each ride is logged. This requirement causes the loss of anonymity and privacy which is provided in a regular and more expensive taxicab service, which may have negative consequences.

Dwork et al. (2006) introduced ε-indistinguishability as a new notion of privacy leakage in 2006. A mechanism is defined as ε-indistinguishable if for all databases D_1 and D_2 differing in a single row and for all responses to a query function, the probability of obtaining response r for the database D_1 is within a (1+ε) multiplicative factor of the probability of obtaining the same response, r, when the database is D_2. Dwork et al. stated that ε-indistinguishability is obtained by adding noise to real data according to the Laplace distribution as $\mathrm{P}[x] \propto e^{-\varepsilon |x|/S(f)}$ where $S(f)$ is the sensitivity of function f: $D^n \rightarrow \mathbb{R}^d$. $S(f)$ is the smallest number such that for all D_1, $D_2 \in D^n$ which differ in a single row, $\|f(D_1) - f(D_2)\|_1 \leq S(f)$. More noise means more privacy. However, as the amount of noise increases, data utility for analysis decreases, so there is a trade-off between privacy and utility. ε determines the amount of privacy loss, the smaller ε is the better privacy and ε is a parameter chosen by the policy. Dwork et al. also called ε as *leakage*. Differential privacy was defined by Dwork (Dwork, 2006) as a new measure in the same year and the formulation of differential privacy is given as follows. A function f is ε-differential private if Formula 2.1 holds for all datasets D_1 and D_2 which differ in at most a single row and for all subsets $S \subseteq \mathrm{Range}(f)$ where P denotes probability.

$$P[f(D_1) \in S] \leq exp(\varepsilon) \times P[f(D_2) \in S] \tag{2.1}$$

For noise calculation, although the Laplace distribution was the first mechanism proposed, the Gaussian (Dwork & Roth, 2014), the Geometric (Ghosh et al., 2009), and the Uniform (Geng & Viswanath, 2013) distributions can be also used for numeric data as an alternative, and the Exponential distribution can be used for non-numeric data (Dwork & Roth, 2014). Differential privacy can be applied to graphs, as well. Graph perturbation (Torra & Salas, 2019) is the noise graph addition to real graph structure and used for obtaining differentially private graphs (Sala et al., 2011; Jorgensen et al., 2016).

2.2 Bitcoin and Blockchain

Bitcoin is a distributed, P2P digital currency where no central authority exists. *Bitcoin* is the unit of the currency, and it is shortened as *BTC*. *Satoshi* is the smallest unit of the currency, and it is equal to a one hundred millionth of a single bitcoin (0.00000001 BTC). Bitcoins can be transferred from one address to another address. A transaction is a

transfer of bitcoins and/or satoshis. Transaction management and issuance of bitcoins are performed jointly by the peers in the network.

Bitcoin uses public key cryptography. The address of a user is the hash of the public key of the user. The user can spend his bitcoins by using his private key to sign in a transaction. As a matter of fact, a bitcoin is a chain of digital signatures. A user can pass a bitcoin to another user by digitally signing hash of the previous transaction by his private key and includes the public key of the new owner in the transaction. New owner verifies signatures to verify the chain of ownership.

2.2.1 Blockchain

The blockchain is the general ledger of Bitcoin; it is the public record of all transactions, which are shared between all users and used to verify transactions. Blockchain consists of blocks. A block contains and confirms a part of new waiting transactions. Confirmation means a transaction getting processed by the network and being added to the blockchain. Transactions at each block are hashed, paired and hashed again until a single hash is obtained, which is the Merkle root (Merkle, 1988). Merkle root is stored in block header. Each block also includes hash of previous block header, which results in a chain of blocks. The basic structure of blockchain is given in Figure 2.1.

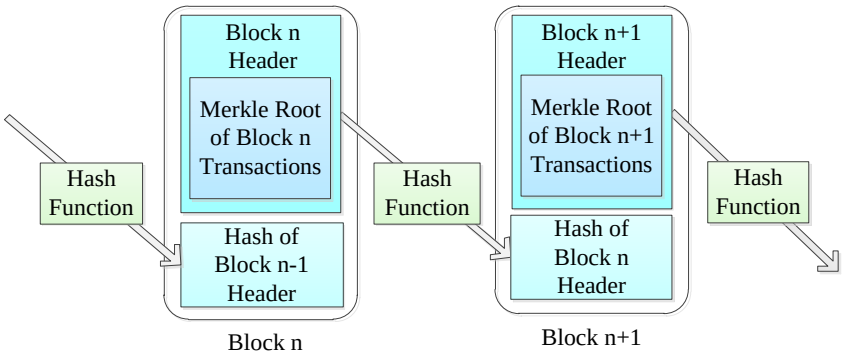


Figure 2.1: Simplified version of blockchain

2.2.2 Transactions

Each transaction has at least one input and one output that include address and amount information. In the input, a user can use bitcoins, which was received as an output in one or more transactions previously. As a result, flow of bitcoins between transactions also forms a chain structure. An output, which is not spent by an input, stays as Unspent Transaction Output (UTXO) until it is spent. The sum of all UTXOs assigned to a user determines the balance of the user. For example, if we say that one has 10 bitcoins, this means that he has 10 bitcoins waiting in one or more UTXOs assigned to him. An example illustration of this chain structure is shown in Figure 2.2. The difference of sum of outputs and sum of inputs in a transaction corresponds to the transaction fee. Transaction fees of all transactions in a block are earned by the miner, i.e., the user who generated that block.

The conditions, which allow the transfer of bitcoins that are held in an output of a transaction to an input of another transaction, are specified by a script, written in a simple non-Turing-complete scripting language. In Figure 2.3, an output segment of a transaction and the corresponding input segment are shown in more detail. The output of transaction n goes to the input of transaction $n + 1$. The one who can satisfy the conditions of pubkey script in the output segment of former transaction gets ownership of bitcoins in the specified amount. The data parameters, which satisfy the conditionals in the pubkey script, are provided in the signature script in the input of the latter transaction to spend these bitcoins. For example, if Alice sends bitcoins to Bob in transaction n, in the output of transaction n, Alice indicates Bob; i.e. in order to assign these bitcoins, she mentions Bob's public key in pubkey script segment. When Bob wants to spend these bitcoins in transaction $n + 1$, Bob has to represent himself in the input, therefore he uses his signature (private key) in the signature script. Elliptic Curve Digital Signature Algorithm (ECDSA) (Johnson et al., 2001) is used with the secp256k1 curve for digital signatures.

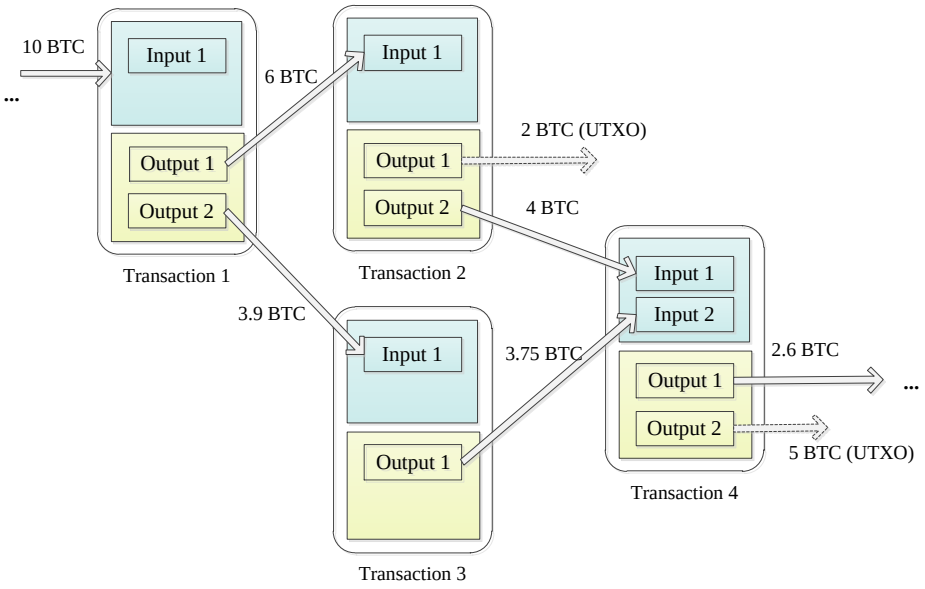


Figure 2.2: A sample flow of bitcoins from transactions to transactions

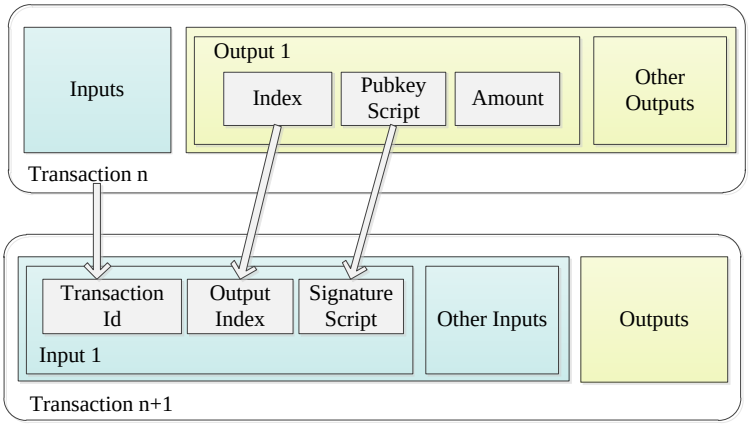


Figure 2.3: Output and input segments of two related transactions from Bitcoin Developer Guide (https://bitcoin.org/en/developer-guide)

2.2.3 Change Addresses

When a user wishes to spend an output of a transaction which is owned by him, he has to use all of it. This is an important difference between Bitcoin transactions and regular bank transactions. For example, in a regular bank account, if a user has 50 dollars in his account, he can use just 20 of it in a payment. However, in Bitcoin, if a user has 10 BTC as a UTXO assigned to him, he has to use all of this 10 BTC in a transaction. If the payment amount, which is indicated in one output of the transaction, is less than the

amount in a UTXO, the user should state a second output address belonging to him to get back the change, if he does not want to give the remaining amount as the transaction fee. As the change address, the user can use an old address or generate a new address to use. It is suggested to use a new address at each transaction to reduce traceability and improve anonymity.

2.2.4 Mining and Incentive

Bitcoin transactions are broadcast to the network. Since the transactions are public, nodes running Bitcoin software checks their validity. Then in the *mining* process, they form a new block that contains new transactions. This new block is added to the latest copy of the blockchain and broadcasted to other nodes. The process of adding a block is called *mining* since each block comes with a reward. In each block generation, new bitcoins are issued and assigned to the creator of the block. Block generation reward halves at every 210 thousand blocks. It was 50 BTC at the beginning, then decreased to 25 BTC, then decreased to 12.5, and it is 6.25 BTC since May 2020. Block generation reward is implemented by putting a new, special transaction as the first transaction in the block where the payee is the creator of the block. This first transaction in the block is called *coinbase transaction*. There is a race in this mining process to get the reward. The mining process also acts as an incentive for nodes to support the network and puts new bitcoins into circulation.

Creator of a block is also rewarded with the transaction fees of the block. A transaction has a transaction fee if the output value of a transaction is greater than its input value. Payers include fees in order to get their transactions processed quicker; miners may select to process transactions with fees. Maximum numbers of bitcoins are determined to be 21 million. Therefore, after that number is reached, the only incentive to mine will be the transaction fees. These incentives increase the possibility of the nodes behaving honestly.

The longest blockchain is recognized as the actual and the latest blockchain. All miner nodes work for generating another block to add to this latest copy. Transactions receive a confirmation when they are included in a block, and a transaction is confirmed again each time another block is added to the blockchain after the block of the transaction.

It should be noted that UTXO of a coinbase transaction cannot be used as an input, which means cannot be spent, for at least 100 blocks. This is for guaranteeing that a block reward is not spent until the permanence of the block in the blockchain becomes absolute. This rule is required due to the possibility of having of blockchain forks.

2.2.5 Proof-of-Work (PoW)

In a block header, a nonce value is included, which is used as the proof-of-work (PoW) for creating the block. PoW is a system to prevent Denial-of-Service (DoS) attacks and other system misuses. This system requires a user to show that he performed some work and spent some effort, i.e., processing time, to complete a task. However, the proof can be easily verified. A PoW is like a puzzle, takes some time and effort to solve, but it can be verified easily. Bitcoin uses a PoW algorithm named *Hashcash* (Back, 2016). In Hashcash, a hash value, which begins with a specific number of zeroes, is required. Since this is a specific requirement and cannot be obtained directly, it shows that the user spends some effort and time. Hashcash can be implemented by incrementing a nonce value until it provides the requirement of a specific number of zeroes at the beginning of its hash. An example of the Hashcash implementation is given in Figure 2.4.

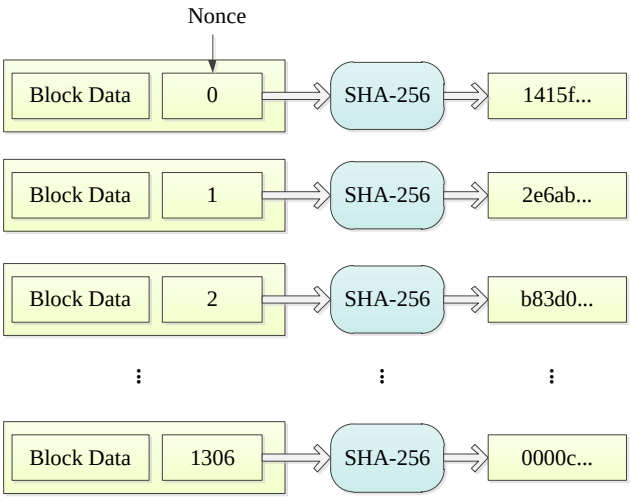


Figure 2.4: Hashcash example

In this example, the required number of zeroes is determined as four in hexadecimal. Nonce value is added to the end of the block data. This nonce value, starting from zero, is incremented until the hash begins with four zeroes. It takes 1,307 tries to obtain a valid hash value.

In this scheme, average work required is exponential in the number of required zero bits, but it can be verified by computing a single hash value. For instance, if the required number of zeroes is 20 and SHA-256 is used as the hash function, out of 2^{256} possible hash values, there are 2^{236} hash values that satisfy this criterion. The possibility of randomly selecting a number that will have 20 zeroes as the beginning of the hash is 1 in 2^{20}. Thus, one has to try $2^{20}/2 = 2^{19}$ values on the average to find such a hash value.

While miners compete for adding a block to the blockchain, they actually race for finding a PoW for a block of transactions. By always accepting the longest chain as the actual and the most recent blockchain, the greatest PoW computation is guaranteed. This means if the majority of nodes in the network are honest, then the chain formed by these honest nodes is the longest and beat any other alternative, competing chains. For example, to modify a past block, finding PoW of the block and the blocks after that block must be achieved again. Also, the work and the chain of the honest nodes must be outperformed, i.e., a longer chain must be obtained. The probability of an attacker to be successful decreases exponentially as the successive blocks are added.

The number of zero bits determining the PoW difficulty is named as *difficulty target*. Improvements in the hardware speed and changes in the number of running nodes in the network affect the generation rate of blocks. Therefore, there is a need for difficulty retargeting and the difficulty target is adjusted according to the block generation rate (average number of blocks per hour), i.e., if blocks get generated too fast, the difficulty increases by increasing the required number of zeroes. The average number of blocks per hour is determined to be 6, so creating a block should take 10 minutes on average. In order to achieve this, every 2,016 blocks, the network calculates the number of seconds to generate these 2,016 blocks by using timestamps at block headers. This value is expected to be 1,209,600 seconds, corresponding to two weeks. If it took less than two weeks to generate these 2,016 blocks, then the difficulty target is increased proportionally, and if it took more than two weeks, then the difficulty target is decreased proportionally.

2.2.6 Double-Spending

The most known problem for digital currencies is the *double-spending* problem, in which a malicious user tries to spend to two different payees with the same money. To prevent double-spending, it must be ensured that a digital coin is not spent twice by its owner. Before Bitcoin, this problem is solved by using a central authority which approves transactions. However, in Bitcoin, if someone tries to spend the same bitcoin twice, blockchain acts as the single source of verification, and the network does not accept to add the second transaction to the blockchain. This is achieved by nodes following *consensus rules*. These are the rules that nodes in the network follow to maintain consensus, i.e., to reach an agreement on having same blocks in their blockchain. These rules are also called *validation rules* since transactions and blocks are validated according to these rules, and a block violating the consensus rules is rejected. Examples of Bitcoin consensus rules are as follows; (i) signatures must be correct for the bitcoins being spent in transactions and (ii) the maximum number of bitcoins that can be created in a block is limited (6.25 BTC as of May 2020).

The PoW (PoW) scheme in the mining process is used by Bitcoin to reach consensus on the blockchain and presents a solution to the Byzantine Generals Problem (Lamport et al., 1982). Miners can include different transactions in their blocks while racing for adding a new block to the blockchain; therefore, it is possible that two miners come up with two different new blocks at the same time and broadcast to the rest of the network. This results in a forked blockchain and obligates the network to reach a consensus on which block to add to the blockchain. In this case, miners save both blockchains but select one of them and try to find a new block to add that chain. Meanwhile, if a miner receives a new block from the network for one of the chains, he discards the shorter chain and continues working for adding a new block to the longer chain. So, the blockchain provides the structure as a single history of the order in which transactions were processed and assures the integrity of the system without a central authority. This is achieved by recording information of all transactions which are impossible to forge but at the same time quickly verifiable. In order to add a block to the blockchain, a hash (SHA-256) of a block of items

(transactions) with their timestamp value is taken which proves the existence of data at that time, and the order between transactions is established. While taking the hash, the hash of all previous blocks is also included as a chain, so each block (and its timestamp) supports the integrity of the previous blocks. Thus, modification of a block requires modification of all the blocks coming afterward. This modification requirement requires a huge amount of processing power due to the PoW structure. To change a transaction which happened 60 minutes (6 blocks) ago, e.g., to remove information that spending some bitcoins for double-spending, one has to change the record for that transaction and solve a new PoW problem for that block (find a new nonce). Then he has to construct an alternative chain which goes forward, by solving a new PoW problem for each block and surpass the actual chain by forming a longer chain. This can be achieved if and only if the malicious user has more computing power than the sum of all other miners' powers and this is known as a *majority attack*; otherwise, he cannot surpass the actual chain. It is suggested to wait for 6 confirmations for a high-value bitcoin transfer, which means that 5 additional blocks should be added after the block that contains the transaction as stated in Bitcoin web site (https://bitcoin.org/en/you-need-to-know).

2.2.7 P2P Network

Peers connect to each other over an unencrypted TCP channel. When a peer first wishes to enter the network, some DNS servers, which are called DNS seeds, are queried. These DNS seeds are hardcoded in Bitcoin clients to find active peers. The response includes IP addresses of the peers that accept new incoming connections. To connect a peer, a *version* message is sent, including version number, block, and the current time of the sender peer. Receiver peer replies this message with its *version* message. Then both nodes send *verack* message to acknowledge that the connection has been established. After a peer enters the network, peer discovery is based on an address propagation mechanism, where peers can request IP addresses from each other with *getaddr* messages, and send their IP address lists to other peers with *addr* messages. Each address has a timestamp which shows its freshness. Peers can have a total of 125 connections, where 8 of them are outgoing connections, and 117 of them are incoming connections, except peers behind NAT or firewall; they can have only outgoing connections and cannot accept incoming

connections. Each peer stores a list of IP addresses of their connections. When a new block is generated by a miner, it is broadcasted by the miner to its peers. Receiving peers validate the block and then forward to their peers. When a peer forms a transaction, then it sends this message to its connected peers. A transaction is sent to a peer by first sending an *inv* message. If the receiver peer replies with a *getdata* response, then the transaction is sent using *tx* message. The transaction continues to be propagated between peers in the same manner if it is a valid transaction. A reputation-based protocol exists, where each peer keeps a penalty score for every connection and increases these penalty scores for the connections that send faulty messages. The peer bans a connection for 24 hours when its penalty score reaches a threshold.

2.2.8 Summary of the Process

As a summary, the process can be defined as follows:

- New transactions are broadcast to all nodes.
- Each miner node collects new transactions into a block.
- Each miner node works on finding a PoW for its block.
- When a miner finds a PoW, it broadcasts the blockchain with the added block to all nodes.
- Other miner nodes accept the addition only if the block and all transactions in the block are valid.
- Other miner nodes show their acceptance by working on creating the next block in the chain, using the hash of the accepted block as the previous hash.
- Miner nodes consider the longest chain and keep working on extending it.
- If two miner nodes broadcast different versions of the blockchain with a valid new block addition at the same time, miners work on the first one they received, but also keep the other chain for the possibility of it becoming longer.
- When a PoW is found for the next block, and one chain becomes longer, nodes that were working on the other chain discard that chain and switch to the longer one.

A brief technical background on Bitcoin is given herein. Unquestionably, the whole technical structure of Bitcoin is much deeper, more detailed information can be found given in Bitcoin Developer Guides (https://bitcoin.org/en/developer-guide).

2.3 Anonymity and Privacy in Bitcoin and Blockchain

In the traditional banking model, information about the parties in transactions are limitedly shared and secured by the trusted third parties, however the banks, which are trusted third parties, know everything about their customers. In Bitcoin, everything is transparent; all transactions are publicly announced. The only thing done for anonymity is to keep public keys anonymous, using pseudonyms for the addresses. Everyone can monitor that users transfer bitcoins to each other, but the real names are not provided, only the pseudonyms are used. It is similar to the stock exchange operations in that sense. Since pseudonyms are used in Bitcoin transactions, the general impression can be as Bitcoin provides anonymity. However, as mentioned explicitly in Bitcoin website (https://bitcoin.org/en/you-need-to-know), it is not anonymous, and it is remarked as "probably the most transparent payment network in the World" (https://bitcoin.org/en/protect-your-privacy). All transactions are kept public where payers and payees are specified with their pseudonyms, which means all transactions of a Bitcoin address (pseudonym) can be seen. Blockchain does not keep balances, just the transactions; however, balance of an address can be calculated after obtaining all transactions related to this address. Nevertheless, since no other information about the users is stored, real identity of a user is not known, unless there is a need for revealing it, e. g., in order to receive services or goods or law enforcement. Even so, the real identity is not revealed to everyone unless the service provider announces publicly. So, it can be said that Bitcoin users cannot stay completely anonymous, but can be considered as pseudonymous.

Except revealing identity to receive service or goods to merchant and/or payment processor, identity and address can be linked while trading bitcoins on exchange since exchanges may be subject to money laundering regulations. In this case, customers need to prove their identity to the exchange.

It can be said that provided level of privacy is determined by the behavior of a user while using Bitcoin. Some security countermeasures are suggested by Bitcoin (https://bitcoin.org/en/protect-your-privacy), e.g., generating and using a new key pair (a new address) for each transaction. Because when a new key pair is generated, it cannot be linked to previous transactions of the user and therefore the number of bitcoins the user has cannot be learned. Another recommendation is to use separate wallets for different purposes. Wallet refers to the programs or files that are used for managing transactions and creating and managing Bitcoin addresses, public-private key pairs as stated in Bitcoin Developer Guide (https://bitcoin.org/en/developer-guide). Transactions at different wallets cannot be linked and can stay isolated. Also, the first thing that has to be done for privacy is to be careful about not to disclose addresses. However, it should be noted that there might be cases where the address is disclosed, e.g. for receiving public donations, or for proving a payment is made in order to receive a good. Similarly, information about transactions, like amount, should not be disclosed since they may help to find addresses related to them. Hosted wallet services know the addresses of the users who use them, because data of wallets are stored in servers owned by the wallet services. Additional information to use these wallet services, such as e-mail address or phone number, can help these services to link this information to identity.

Bitcoin also warns its users to take some issues into consideration of which users may not be aware. For instance, it should be known that IP address of a Bitcoin address can be logged by connecting to active nodes in the network and listening for transaction relays. Similarly, the Internet service provider (ISP) can intercept transaction messages that a user sends and figure out the addresses owned by him. IP addresses do not directly reveal identity, but they can be used to find it. IP addresses can be hidden by using anonymization services like TOR. There are also mixing services, which mix transactions of users by receiving and sending back the same amount using independent addresses and make impossible to trace activities of users. However, these services require trust to the service since the users actually transfer bitcoins to the service. Moreover, these services can log requests of users. Another reason which limits the usage of mixing services is that these services are inefficient for hiding large transactions, although they are effective while hiding small amounts.

Bitcoin’s blockchain was designed to be public, and it does not provide privacy per se. We just briefly mention some important blockchain privacy studies here. Enigma

(Zysking, 2015) is a P2P network in which different parties can jointly store data and run computations, at the same time keeping the data private. The computational model of Enigma is based on Secure Multiparty Computations (SMC). Data is split between nodes and data queries are computed distributed. Blockchain serves as an unalterable log of events and manages identities and access control. A privacy respecting approach for blockchain-based sharing economy applications, which leverages a zero-knowledge scheme, was proposed (Xu et al., 2017). There are also recent studies on privacy in blockchains of smart contracts. Ethereum (Buterin, 2014; Wood, 2014) is the first smart contract blockchain platform. All transactions in Ethereum were public, as in Bitcoin. However, Ethereum platform enables setting up private and permissioned blockchains to improve privacy. For instance, Quorum (https://quorum.com) supports both public and private transactions. Details of private transactions are revealed only to those party of the transactions. Symmetric encryption and hash functions are used to keep data private. A distinct private state database is stored at each node additional to the common public state database. Private contract code in a private transaction can only be executed by the nodes that are party to that transaction. Privacy is obtained in Hyperledger Fabric (https://hyperledger-fabric.readthedocs.io), another smart contract platform, similar to Quorum; by using hash functions and symmetric encryption.

2.4 Smart Utility Metering

Utilities are the organizations supplying the community with electricity, gas, or water. Traditional utility services are centralized and work in a subscription basis; houses that would like to utilize utilities apply to the utility providers and become subscribers. Utility meters are devices that are used for measuring utility usages. These meters have parts that show current consumption records. Utility providers obtain these meter records regularly and bill the utility users. In traditional utility systems, obtaining meter records are done manually; personnels of the utility providers visit the subscribers, check their meters, and record the readings. Therefore, this operation requires labor and is prone to human errors, since readings are gathered and transferred manually. Moreover, although households may check the meters and read the numbers in the meters, it is difficult to convert current

consumption numbers to payment amounts at any time in order to anticipate and control their bills at that billing period.

Smart utility metering is conversion of traditional utility metering systems to intelligent systems. Smart utility metering is also called smart metering or smart grid. Smart metering systems are realized by advanced metering infrastructures (AMI). AMI consists of several enhanced technologies such as smart meters (SM), home area networks (HAN), wide area networks (WAN) or neighbored networks (Kabalci, 2016). In smart metering systems, the consumption and other related billing parameters are measured in predefined intervals. The measured data are transmitted to utility companies over wireline or wireless networks. Therefore, smart metering systems do not require labor work and there are no room for human error. Smart metering also enables household to view their current consumptions and billing amounts via smart devices.

2.5 Privacy in Smart Utility Metering

Smart utility metering raises privacy concerns since real-time consumptions of houses are transmitted to utility centers. Transmission of real-time consumption data leaks information about household, such as number of people at the house, their sleeping and eating routines, since usage patterns from smart meter data can be extracted using statistical methods (Molina-Markham et al., 2010). For instance, if the consumption values transmitted to the utility providers are zero in a period, the utility provider can conclude that there is no one at the house at that period. This information can be used for malicious activities such as theft. Frequent transmission of consumption values to utilities also allows utilities to utilize consumption data to raise the prices according to the consumption, if dynamic pricing is used (Hassan et al., 2019). Privacy concerns using blockchains on smart metering are addressed by European Commission Joint Research Centre Smart Electricity Systems and Interoperability (https://ses.jrc.ec.europa.eu/node/31976), as well. Privacy improvement mechanisms can be added to smart metering to improve privacy, such as securing measurements by encryption. However, differential privacy has several advantages over encryption in smart utility metering scenario. Encryption requires high computational capacity and cooperation between all smart

meters to exchange distributed keys. Moreover, failure of even a single smart meter may cause faults in the whole network (Barbosa et al., 2016). Compared to encryption and other cryptographical methods, differential privacy is simple to implement and light-weight. Moreover, the level of privacy can be tuned (Gough et al., 2021).

3. RELATED WORK

In this section, we give related work on anonymity and privacy in Bitcoin-like digital cash systems and application of differential privacy to financial distributed ledger applications in the following subsections.

3.1 Survey on Anonymity and Privacy in Bitcoin-like Digital Cash Systems

We present a brief survey on the studies, related to anonymity and privacy issues in Bitcoin in this subsection. There are studies that survey the literature related to anonymity and privacy of the Bitcoin blockchain and Bitcoin-like cryptocurrencies (Schaffner, 2017; ShenTu & Yu, 2015a; Herrera-Joancomartí, 2015; Bonneau et al., 2015; Narayanan et al., 2016; Tschorsch & Scheuermann, 2016; Maurer, 2016; Conti et al., 2018; Fabian et al., 2018; Genkin et al., 2018; Alsalami & Zhang, 2019; Feng et. Al, 2019; Averin et al., 2020; Zaghloul et al., 2020; Zhu et al., 2020; Bergman & Rajput, 2021; Peng et al., 2021; Ghesmati et al., 2022). We classify studies related to anonymity and privacy issues in Bitcoin according to methods and outcomes. For instance, Ethereum (Buterin, 2014; Wood, 2014), which is the second widely used digital currency utilizing smart contracts, is not covered in our analysis and taxonomies since it does not focus on improving anonymity and privacy. Transactions in Ethereum are public (Tikhomirov, 2017) as in Bitcoin. Nevertheless, proposals for improving anonymity and privacy in Bitcoin can also be evaluated and used for improving anonymity and privacy in Ethereum or similar cash systems. For instance, Zero Knowledge Succinct Non-interactive ARguments of Knowledge (ZK-SNARKs) (Ben-Sasson et al., 2013a), which are a special kind of Succinct Non-interactive ARgument of Knowledge (SNARK) (Ben-Sasson et al., 2013b), was integrated to Ethereum as of September 2017 (Wilcox, 2017).

We divide the studies related to anonymity and privacy in Bitcoin and similar digital cash systems in two main categories: (i) the studies that analyze the anonymity and privacy in Bitcoin, and (ii) the studies that propose improvements for anonymity and/or user privacy as an extension or alternative to Bitcoin. We detail these two categories in the following subsections.

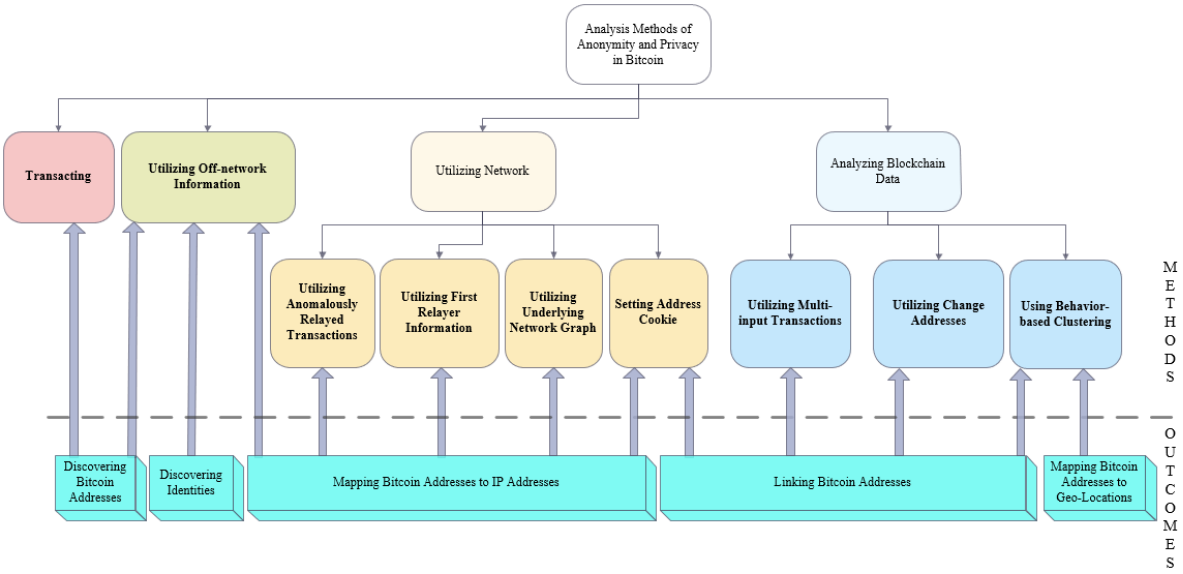


Figure 3.1: Taxonomy of methods of analyzing anonymity and privacy in Bitcoin and the outcomes

3.1.1 Taxonomy of Studies on Anonymity and Privacy Analysis

We classify methods of analyzing anonymity and privacy in Bitcoin that are described and used in the literature as given in Figure 3.1. Essentially, analyzing anonymity and privacy is done through spending effort to achieve deanonymization and extract information that would impair privacy of users. Therefore, methods in the literature serve these purposes. For a method, while some studies may use the method, some studies may only mention the method but do not use it; therefore, for each method, studies that mentioned or applied the method are given respectively. Resulting outcomes are given in the bottom part of the figure. Outcomes are actually potential aims to be achieved after analyses. There are five outcomes of analyzing anonymity and privacy in Bitcoin. Each outcome is described in detail in the following.

- Discovering Bitcoin Addresses: Possible Bitcoin address of a person or an entity

is discovered starting from an identity information, such as name and surname of a person, or name of a company.

- Discovering Identities: Possible identity information, such as name and surname of a person or company name is obtained starting from a Bitcoin address.
- Mapping Bitcoin Addresses to IP Addresses: Bitcoin addresses are mapped to possible IP addresses where the transactions are generated.
- Linking Bitcoin Addresses: Bitcoin users are suggested to use new Bitcoin address each time they receive a new payment in the Bitcoin website (https://bitcoin.org/en/protect-your-privacy). Therefore, a user can have more than one Bitcoin address. Addresses expected to belong to the same user are linked together in this outcome.
- Mapping Bitcoin Addresses to Geo-Locations: Information about the physical location of a user is obtained starting from Bitcoin address.

Transition can happen between these outcomes. For instance, a Bitcoin address belonging to a person can be discovered, and then this address can be linked to his other Bitcoin addresses. Similarly, a Bitcoin address can be mapped to the IP address, and then this IP address can be used to discover identity or geo-location of the user that owns the Bitcoin address.

We describe each method in more detail, and give the related studies for each method in the following.

- Transacting: By transacting with other users, e.g., purchasing of goods and services, Bitcoin address of the other end is learned. A buyer must know Bitcoin address of a seller in order to make a payment to the seller, so a seller must share his Bitcoin address if he wants to receive payments. Therefore, one can act as a buyer and learn Bitcoin addresses of parties that he would like to know, assuming that these parties are involved in sales activities. Transacting method means actively participating in the network and may also include using marked coins or operating a money laundry service as Reid & Harrigan (2012) stated. Meiklejohn et al. (2013) used transacting method and named it as *re-identification attack*. Transacting can also be used to understand the mode of operation of mixing services, i.e., anonymization services as done by Möser et al. (2013) and Wu et

al. (2021).

- Utilizing Off-network Information: Publicly available off-network data sources, which are obtained externally (out of Bitcoin network and blockchain), can be used to discover identities belonging to Bitcoin addresses, or vice versa. Donation websites that include information related to Bitcoin addresses to prevent service abuses can be given as an example of these data sources. Also, users can voluntarily disclose Bitcoin addresses in forums. In addition, large and highly active entities are publicly recognized on the website blockchain.com. Off-network information from this website can also be used to obtain IP address belonging to a Bitcoin address that initiates the transaction. Reid & Harrigan (2012), Ron & Shamir (2013), Ortega (2013), Meiklejohn et al. (2013), Fleder et al. (2015), Spagnuolo et al. (2014), Baumann et al. (2014), Biryukov et al. (2014), Lischke & Fabian (2016), and Jawahari et al. (2020) used this approach.

- Utilizing Network: By analyzing Bitcoin network traffic or using network infrastructure, information about transactions can be obtained. *Utilizing anomalously relayed transactions*, *utilizing entry nodes*, *utilizing first relayer information* and *setting address cookie* for user fingerprinting are the analysis methods which utilize the Bitcoin network.

    - Utilizing Anomalously Relayed Transactions: Sending of a message by a peer to other peers is also called relay. By analyzing Bitcoin network traffic and transaction message relays, abnormal relay patterns can be defined, such as a transaction being relayed by a single person or a transaction being rerelayed (relayed more than once) by at least one user. Transactions matching these patterns can be used to map Bitcoin addresses to IP addresses. Koshy et al. (2014) proposed this method, inspired by the idea of using P2P network information which was introduced by Kaminsky at his blog (https://dankaminsky.com/2011/08/05/bo2k11).

    - Utilizing First Relayer Information: When connected to every node in the Bitcoin network, it can be assumed that the first node to inform of a transaction is the source, i.e., the owner of it. P2P network and relays were introduced as another source of data for deanonymization firstly by Kaminsky, where he proposed utilizing first relayer information. This

method was also mentioned and accepted by Ortega (2013) Fanti & Viswanath (2017), Neudecker & Hartenstein (2017), and Biryukov & Tikhomirov (2019a; 2019b) used this method.

- Utilizing Underlying Network Graph: Bitcoin clients can be identified by utilizing underlying P2P network graph. Biryukov et al. (2014) introduced utilizing entry nodes method. Entry nodes are the nodes that a Bitcoin client connects to. Fanti and Viswanath (2017), and Feld et al. (2014) used this method.
- Setting Address Cookie: Different transactions and Bitcoin addresses of the same user and IP addresses can be linked by setting an address cookie on the user's computer. The user can be fingerprinted simply by checking this address cookie. This method does not require blockchain analysis and is based on Bitcoin's peer discovery mechanism. Since Bitcoin peers get addresses from other peers, a unique combination of fake addresses, which would behave as a fingerprint, can be sent to a peer to fingerprint him. The peer stores these addresses. The next time he connects, his address database can be queried, and the user is identified if the fingerprint addresses (in the address cookie) are present. This method was proposed by Biryukov & Pustogarov (2015) to correlate the same user across different sessions.

- Analyzing Blockchain Data: Since entire transaction history is publicly available in the blockchain, flow of bitcoins between Bitcoin addresses is traceable. Blockchain data can be gathered using APIs of Bitcoin clients. Bitcoin Core (https://bitcoin.org/en/bitcoin-core), which is the official Bitcoin client, and blockchain.info are the two well-known and widely used clients. Reid & Harrigan (2012) were first to analyze blockchain for anonymity and privacy of Bitcoin. They introduced two network structures, transaction and user networks, which are also used and utilized in subsequent studies extensively. The flow of bitcoins between transactions over time is shown in the transaction network and flow of bitcoins between users over time is depicted in the user network. Constructing transaction network from blockchain is straightforward; transactions are represented as nodes and flow of bitcoins are represented as directed edges with

amount and timestamp information. An output of a source node becomes input to a target node.

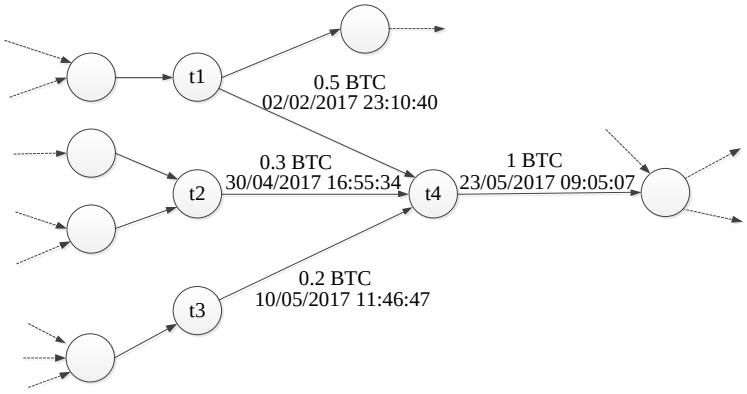


Figure 3.2: Example sub-network of transactions

Figure 3.2 shows an example sub-network of transactions. $\boldsymbol{t1}$ has one input and two outputs. It was performed on 2nd February 2017, and one of its outputs transferred 0.5 BTC. $\boldsymbol{t2}$ is a transaction with two inputs and one output. It was performed on 30th April 2017, and the output transferred 0.3 BTC. $\boldsymbol{t3}$ is a transaction with one input and one output. It was performed on 10th May 2017, and the output transferred 0.2 BTC. Finally, $\boldsymbol{t4}$ is a transaction with three inputs and one output. It was performed on the 23th May 2017. All inputs come from the outputs of $\boldsymbol{t1}$, $\boldsymbol{t2}$, and $\boldsymbol{t3}$, which are mentioned above. The output of $\boldsymbol{t4}$ is 1 BTC; equal to the sum of its inputs.

In the user network, users are represented as nodes whilst directed edges, which have also amount and timestamp information, represent flow of bitcoins between them. A source node represents a payer, whereas a target node represents a payee. A user node includes Bitcoin addresses (hash of the public key) of the corresponding user. User network cannot be derived from the blockchain directly; it needs extra work. At first, the graph can be constructed by representing each address with a node as shown in Figure 3.3. In this figure, each square represents a Bitcoin address, and directed edges are the transfer of bitcoins between addresses.

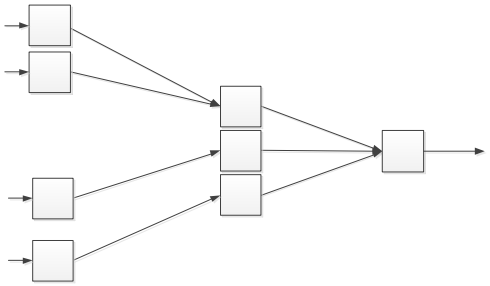


Figure 3.3: Formation of user network, representing each address with a node

However, a user may have multiple addresses, since it is suggested to generate a new private-public key pair for each transaction. Reid & Harrigan tried to cluster nodes, which belong to the same user, using this fact. It is not possible to obtain a perfect and true user network, since real owners of Bitcoin addresses are unknown; in this way Bitcoin provides anonymity to some extent. Nevertheless, different Bitcoin addresses that are expected to belong to the same user can be linked by (***i***) utilizing multi-input transactions, (***ii***) utilizing change addresses and (***iii***) behavior-based clustering. Figure 3.4 shows clustering of addresses in Figure 3.3 into users. Each circle represents a user and contains addresses owned by that user. Directed edges represent transfer of bitcoins between users.

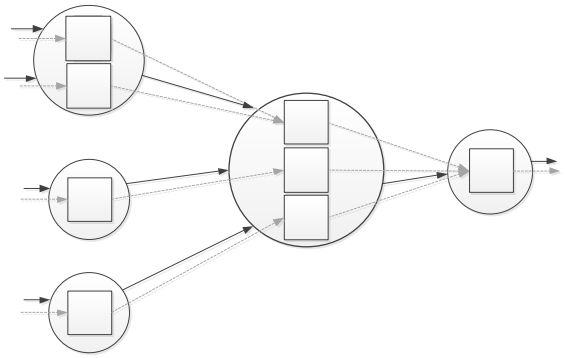


Figure 3.4: Formation of user network, clustering addresses to users

- Utilizing Multi-input Transactions: Different addresses can be linked to a single user utilizing multi-input transactions. A multi-input transaction occurs when a user performs a payment using more than one address by combining these addresses in a transaction. This may happen, for example, when the payment amount is greater than each of the balances in the user's addresses. This fact is also indicated by Nakamoto (2008); multi-input transactions reveal that their inputs are owned by the same user and if the

owner of an address that is used in one of these inputs is revealed, then it can be figured out that the other transactions, using other input addresses, belong to the same user. This linking can be simply made by analyzing transactions in the blockchain. Reid and Harrigan (2012) were first to use this heuristic. Most of the studies that analyze blockchain data adopted this heuristic. Ron & Shamir (2013), Androulaki et al. (2013), Ober et al. (2013), Ortega (2013), Meiklejohn et al. (2013), Fleder et al. (2015), Spagnuolo et al. (2014), Baumann et al. (2014), Lischke & Fabian (2016), Dupont & Squicciarini (2015), Zhao & Guan (2015), Nick (2015), and Jourdan et al. (2018) are the other researchers utilizing this heuristic.

- Utilizing Change Addresses: Change addresses are the addresses generated by Bitcoin to allow users to take their changes. If a transaction has two outputs and one of them is an old address, and the other is a new address, then it can be assumed that the new address is the change address and belongs to the user who owns the input address of the transaction, or similar heuristics can be used. Transactions in the blockchain can be analyzed to find out change addresses that are expected to belong to the users who are input to transactions, and these addresses can be linked. Androulaki et al. (2013) were first to use this heuristic. Ortega (2013), Meiklejohn et al. (2013), Möser et al. (2013), Spagnuolo et al. (2014), Zhao and Guan (2015), Nick (2015), and Neudecker & Hartenstein (2017) are some of the other researchers utilizing this heuristic. Zhao et al. (2022) improved the heuristic for change address for better identification when there are multiple outputs by considering transaction fees and amounts.

- Using Behavior-based Clustering: Clustering is assigning each object in a set of objects to a group (cluster) such that objects in the same cluster are more similar to each other than to those in other clusters. Behavior-based clustering is clustering by evaluating behaviors of objects. Behavior-based clustering techniques can be used to extract data about the users, like linking Bitcoin addresses that are expected to belong to the same user. Several attributes can be determined and data can be retrieved from the blockchain for the analysis. In addition, by analyzing spending habits of the users, possible information about the physical locations of the users

can be determined. By analyzing the times of day at which a user has made transactions, an informed guess can be made as to that user's time zone of residence. Reid & Harrigan (2012) mentioned that Bitcoin addresses that are used at similar times over an extended time period may be owned by the same user, therefore clustering can be done using this heuristic. Androulaki et al. (2013) were first to use behavior-based clustering. Ortega (2013), Ron & Shamir (2013), Dupont & Squicciarini (2015), Bistarelli et al. (2021), and Xueshuo et al. (2021) also used this heuristic.

These described methods can be used in combination to obtain further information.

3.1.2 Taxonomy of Studies with Anonymity and Privacy Improvements

We classify methods of improving anonymity and privacy in Bitcoin-like digital cash systems that are used in the literature as given in Figure 3.5. In this figure, we included a hierarchic numbering to serve as an index. For each method, studies that applied the method is given. Resulting outcomes are given in the bottom of the figure. The outcomes and the methods are explained in detail in the following subsections. The number of the studies that have proposals against network analysis is quite a few compared to the number of the studies that have proposals against network analysis, and a detailed taxonomy cannot be provided for the studies on against network analysis. Therefore, for the sake of readability and clarity, these studies are not included in Figure 3.5, although they are described.

Outcomes in the taxonomy of studies on anonymity and privacy analysis in Bitcoin, which were described in Section 3.1.1, and the methods in the taxonomy of studies with anonymity and privacy improvements, which are described in this section, are related to each other as given in Table 3.1. To be more specific, Table 3.1 includes the improvement methods that address outcomes of the analyses.

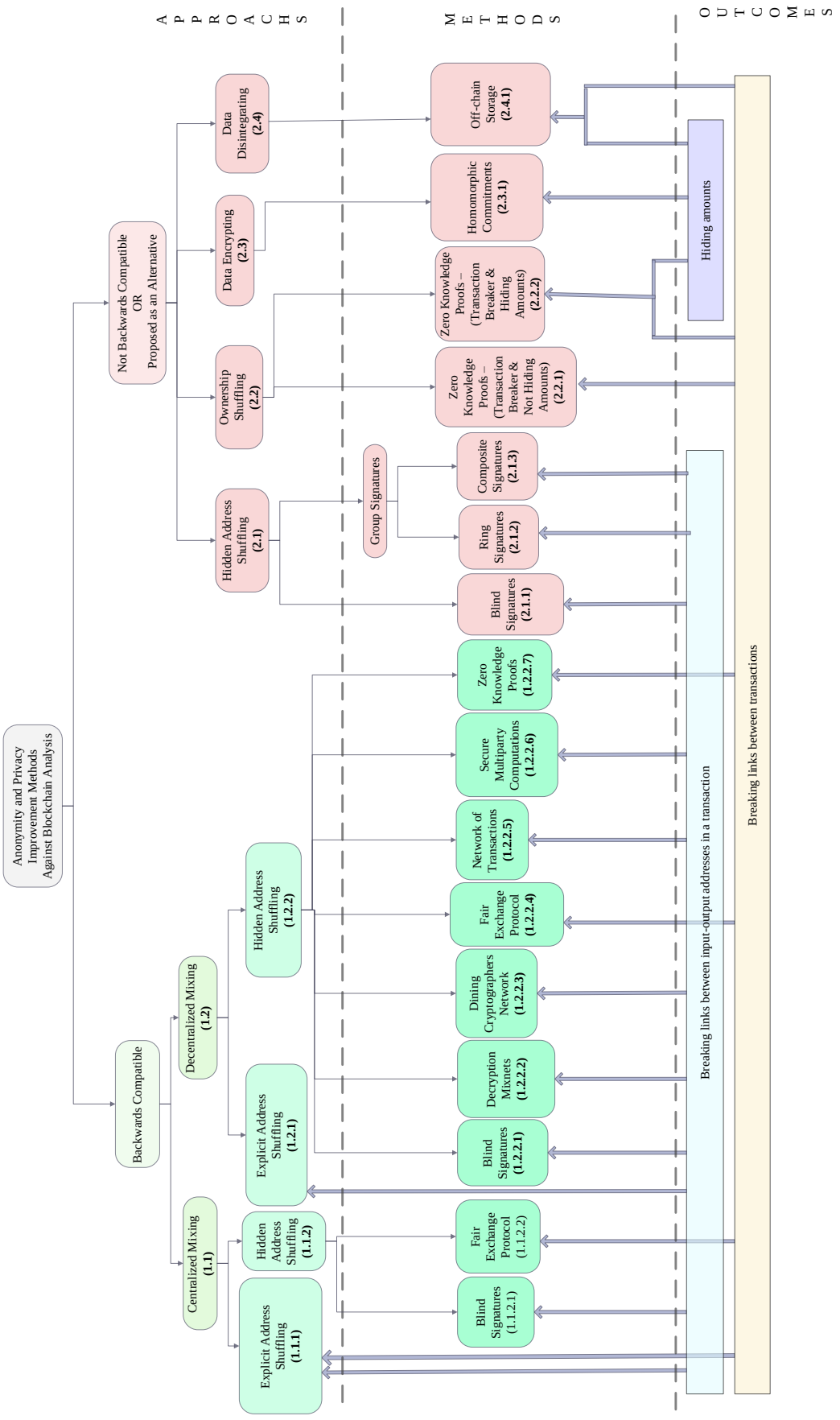


Figure 3.5: Taxonomy of anonymity and privacy improvements against blockchain analysis

Table 3.1: Relationship of outcomes of analyses and improvement methods

Outcome of Analyses	Improvement Methods that Address the Outcomes
Discovering Bitcoin Addresses	Cannot be addressed by the methods
Discovering Identities	Cannot be addressed by the methods
Mapping Bitcoin Addresses to IP Addresses	All methods against network analysis
Linking Bitcoin Addresses	All methods against blockchain analysis except Homomorphic Commitments
Mapping Bitcoin Addresses to Geo-locations	Cannot be addressed by the methods

Discovering Bitcoin addresses is done by transacting or utilizing off-network information. There is no measure that can be taken against transacting, if the receiver would like to receive bitcoins, then he has to provide his Bitcoin address to the sender. There is no improvement method against utilization of off-network information as well; the measure that can be taken is not sharing any information that will relate identity to Bitcoin addresses in the off-network.

Discovering identities is done by utilizing off-network information, therefore this outcome cannot be addressed by the improvement methods against blockchain analysis or network analysis. Any information that will relate Bitcoin addresses to identity information should not be shared in the off-network to prevent discovery of identities.

Mapping Bitcoin addresses to IP addresses is done by utilizing the network. Therefore, it can be addressed by the methods against network analysis.

Linking Bitcoin addresses is performed by analyzing blockchain data or setting address cookie by utilizing the network. This outcome can be addressed by all methods against blockchain analysis except homomorphic commitments and the methods against network analysis. Homomorphic commitments only hide amount information in transactions. Therefore, relationships of Bitcoin addresses between transactions remain explicit, and they cannot address this outcome.

Mapping Bitcoin addresses to geo-locations is done by analyzing blockchain data and behavior-based clustering. Times of day that a user makes transactions are used in this type of analysis, and this information cannot be hidden. Therefore, this outcome cannot be addressed by any improvement method.

We detail outcomes and methods in the following. There are four main outcomes of the anonymity and privacy improvement methods. These outcomes are (i) breaking links between input-output addresses in a transaction, (ii) breaking links between transactions, (iii) hiding amounts and (iv) hiding IP addresses. Details are given in the following.

- Breaking links between input-output addresses in a transaction: Links between inputs and outputs of a transaction are broken, and inputs and outputs cannot be linked. In other words, input-output address links are obfuscated. For an input address in a transaction, one cannot determine the addresses that are output in that transaction, or for an output address in a transaction, one cannot determine the addresses that are input to that transaction.

- Breaking links between transactions: Two transactions are linked if an output of one of them becomes input to the other. Breaking links between transactions is removing links by adding link obscuring mechanisms in the middle. For a given input of a transaction, the output of another transaction, which becomes the source to that input, cannot be detected if links are broken between two transactions. Similarly, for a given output of a transaction, the input of another transaction, which becomes the destination to that output cannot be detected.

- Hiding amount: For improving privacy, amount values in transactions are hidden. However, this outcome prevents checking the integrity of the system as a whole, for instance, one cannot count the total number of coins in the system since the amounts are hidden. As a result, if someone can break the system, he can issue coins without being detected.

- Hiding IP addresses: IP address of a Bitcoin user is hidden. This results in preventing linking of Bitcoin addresses to IP addresses. This outcome is not included in Fig. 10 since it is the outcome of the methods against network analysis. We did not include the methods against network analysis in the figure either, for the sake of readability and clarity.

Proposals that improve anonymity and privacy in Bitcoin and Bitcoin-like digital cash systems can be divided into two main categories. The first category is the group of proposals against deanonymization by network analysis, and the second category is the group of proposals against deanonymization by blockchain analysis.

The outcome of the methods used for preventing network analysis is hiding IP addresses. Methods against blockchain analysis do not prevent against network analysis, therefore they are suggested to be used in conjunction with the methods against network analysis. TOR (Dingledine et al., 2004) is the most popular tool used for achieving anonymity, i.e., hiding real IP address, while using the Internet. It is a distributed overlay network (Lua et al., 2005). Data is encrypted multiple times at the beginning according to the three TOR nodes that the user selects. While traveling in the network, data is routed through a path over that nodes, where decryption of one layer is done at each node, like peeling the layers of an onion, until it reaches its destination. TOR design is based on Chaum's mixnets (Chaum, 1981). It is common among Bitcoin users preferring to use Bitcoin over TOR in order to hide IP addresses. Anoncoin (https://bitcointalk.org/index.php?topic=1481693.0) is another coin that has built-in support of TOR.

The Invisible Internet Project (I2P) (www.geti2p.net) is another onion routing tool that creates a hidden network within the Internet. There are Bitcoin clients that are developed to allow running Bitcoin with I2P as stated in Bitcoin Forum (https://bitcointalk.org/index.php?topic=151181) and Bitcoin exchanges that can be used with I2P (https://bitcointalk.org/index.php?topic=6025.0). Besides supporting TOR, Anoncoin is designed to be a fully I2P darknet coin.

Transaction Remote Release (TRR) (ShenTu & Yu, 2015b) is a new anonymization technology designed for Bitcoin. TRR is inspired by TOR. Its design goal is to defeat attacks that exist while using Bitcoin over TOR. TOR encrypts all blockchain data. However, TRR only encrypts and transmits new transactions since it is designed specifically for Bitcoin. As a result, the performance and throughput of nodes are improved. The need for modifying Bitcoin protocol is the weakness of TRR.

Proposals against deanonymization by blockchain analysis can also be examined in two broad categories. The first category is the proposals that are backwards compatible, in which no modification is required to the Bitcoin protocol; thus, the proposed approach can be deployed immediately. The deployment of such a proposal does not affect the

soundness of the previous transactions and the blockchain that exist until the deployment. The second category includes the proposals that are not backwards compatible. These proposals are either developed for Bitcoin, but needs modification to the Bitcoin protocol to run, or proposed as an alternative to Bitcoin, i.e., to run independently. These two main categories for improvement proposals are divided into subcategories according to the approaches, protocols, and methods used as shown in Figure 3.5 and these are explained in the following.

Mixing is the main approach that is adopted by the proposals that are backwards compatible. Mixing can be achieved by obfuscating inputs and outputs of a transaction. Maxwell introduced this idea to the Bitcoin community with his CoinJoin proposal in 2013 in Bitcoin Forum (https://bitcointalk.org/index.php?topic=279249.0). In CoinJoin, which is a transaction formation style to improve privacy, users make joint payments by forming transactions together. Although most of the studies that analyze blockchain data assumed that inputs of a multi-input transaction belong to the same user, Maxwell stressed that it is not a requirement and the opposite is very possible. In CoinJoin, Bitcoin users individually and separately sign a transaction, where they agree on a set of input and a set of output addresses. Then they merge their signatures. As a result, obfuscation is achieved by shuffling the addresses. A transaction formed in this way cannot be distinguished from a transaction that is formed conventionally. Visualization of a sample CoinJoin transaction with three users is given in Figure 3.6. Each user provides an input to a transaction, and each user receives an output; however, which output is owned by which user is not known for an outsider, i.e., the ones that do not participate the transaction. The transaction acts as a black box. To increase anonymity, it is important to determine a uniform amount and provide inputs accordingly. This hardens for an outside party to distinguish input and output relations and the anonymity set size becomes the number of parties in the transaction.

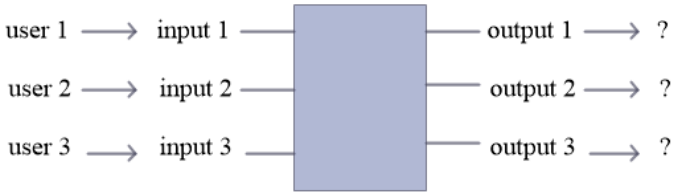


Figure 3.6: A sample CoinJoin transaction

CoinJoin can be implemented in both centralized and decentralized ways, as Maxwell described four alternatives. Although CoinJoin has remained in a forum post, and it was not turned into a paper, it has been widely accepted, and many proposals that are inspired by it came after.

Centralized Mixing (1.1): Mixing is performed by a central mixing server. Users that would like to mix their coins share their input and output address information and the mixing server break links between these addresses. A centralized mixing service can be implemented in two ways, according to the address shuffling, i.e., relation of input and output addresses, being visible to the mixing service or not.

In the explicit address shuffling (1.1.1), the relation of input and output addresses is explicit to the mixing server. The mixing server can link input and output addresses of the users. Therefore, users cannot stay anonymous against the mixing server, although other parties cannot trace the flow of coins after the mixing. The outcomes of this methods are breaking links between input-output addresses or breaking links between transactions, depending on the number of transactions done to perform mixing. If mixing is done within a single transaction, the outcome becomes breaking links between input-output addresses. Several transactions can also be used to perform mixing, i.e., mixing server can transfer user's assets from user's input address to an address that belongs to the server and then can send to output address of the user from another address that belongs to the server. If several transactions are used, then the outcome becomes breaking links between transactions. CoinJoin can be implemented in this way, such that centralized mixing by explicit address shuffling is performed, as stated before. Möser et al. (2013) provided examination of several centralized mixing services. Mixcoin (Bonneau et al., 2014), which was published as a refereed paper, is a centralized mixing approach where mixing service gets to know input and output addresses of the users.

In the hidden address shuffling (1.1.2), the relation of input and output addresses is hidden from the mixing server. The mixing server cannot link input and output addresses. Hidden address shuffling is done using either blind signatures or fair exchange protocols.

- *Blind Signatures (1.1.2.1):* The idea of blind signature came from Chaum (1983). In a blind signature scheme, the content of a message is blinded by the message owner using a blinding factor before it is signed. This is required when the signer should not know the content of the message. The signed message can be later verified against the

signer's public key. The owner can unblind the message, i.e., remove the blinding factor. CoinJoin can be implemented using blind signatures, such that centralized mixing by hidden address shuffling is performed, as stated before. Blindcoin (Valenta & Rowan, 2015), which was proposed as an improved version of Mixcoin, hides output address from the mixing server by utilizing a blind signature scheme (Fuchsbauer, 2009), as a result, the input and output address linking is not explicit to the mixer. The outcome becomes breaking links between input-output addresses in a transaction. Another blind mixing scheme based on an Elliptic Curve Cryptography (ECC) blind digital signature algorithm was proposed by ShenTu & Yu (2015b). Fei et al. (2020) proposed an anonymous Bitcoin mixing scheme based on semi-trusted supervisor using group blind signature.

- *Fair Exchange Protocol (1.1.2.2):* A fair exchange protocol ensures that either all participating parties get the exchanged item or all of them get nothing (Schunter, 2005). TumbleBit (Heilman et al., 2017) is a scheme that allows anonymous payments through a mixing server, where no trust to the server is required. TumbleBit was built on blindly signed contracts (Heilman et al., 2016), which was not backwards compatible with Bitcoin. The link between input and output Bitcoin addresses are broken by performing multiple transactions; therefore, the outcome becomes breaking links between transactions. Another fair exchange protocol that guarantees strong fairness while preserving the anonymity of the consumer and the merchant was provided by Jayasinghe et al. (Jayasinghe et al, 2014].

In the decentralized mixing (1.2), no third party, i.e. a central mixing server, is required. Mixing is performed collectively by the participating users. Decentralized mixing can be done by either explicit address shuffling or hidden address shuffling.

In the decentralized mixing with explicit address shuffling (1.2.1), relation of input and output addresses is learned by the participating users; there is not any mechanism to hide. CoinJoin can be implemented in this manner. Cloak coin (https://www.cloakcoin.com) is another example of decentralized mixing with explicit address shuffling. In Cloak coin, users cloak transactions of other users by providing inputs and outputs and earn a reward in return. However, a cloaked user learns input-output address pairs of the cloaking users. Since mixing is done in a single transaction, the outcome is breaking links between input-output addresses in a transaction.

In decentralized mixing with hidden address shuffling (1.2.2), the relation of input and output addresses is hidden from other participating users in the mixing. This can be done using various cryptographic protocols discussed below.

- *Blind Signatures (1.2.2.1):* As mentioned before, Maxwell's CoinJoin can be implemented in a decentralized way by using blind signatures resulting in breaking links between input-output addresses in a transaction. Another study that uses blind signatures focused on secure and joint Bitcoin trading (Wu et al., 2017) using partially blind fuzzy signatures, in case of a Bitcoin account is owned by multiple people.

- *Decryption Mixnets (1.2.2.2):* Decryption mixnets were introduced by Chaum (1981). In these structures, a set of inputs pass through a set of mix nodes, where each mix node shuffles the inputs and applies encryption and decryption. CoinParty (Ziegeldorf et al., 2015; Ziegeldorg et al., 2018) is a mixing protocol designed using combination of decryption mixnets with threshold signatures (Bleumer, 2005). CoinShuffle (Ruffing, 2014) was proposed as another decentralized mixing protocol which was inspired by CoinJoin and the accountable anonymous group messaging protocol Dissent (Corrigan-Gibbs & Ford, 2010). A general approach for pseudonym mixing, based on CoinShuffle, was proposed and a specific design for Bitcoin, which was called BitNym, was given in detail by Florian et al. (2015). Privacy-enhancing overlays (Meiklejohn & Orlandi, 2015) is another study that explained how mixing can be achieved by utilizing decryption mixnet approach. Coutu (2014) proposed a decentralized synchronous N-to-N mixing model in his master thesis. SecureCoin (Ibrahim, 2017) is another study which is fully compatible with Bitcoin.

- *Dining Cryptographers Network (1.2.2.3):* Dining Cryptographers Network (DC-net) is a method proposed by Chaum (1988). For a DC-net consisting of two users, the users share a key $\boldsymbol{k}$. When one of the users wishes to anonymously publish a message $\boldsymbol{m}$, where $|\boldsymbol{m}| = |\boldsymbol{k}|$, he publishes $\boldsymbol{M}_1 = \boldsymbol{m} \oplus \boldsymbol{k}$, where $\oplus$ denotes the *exclusive or* (XOR) operation (bitwise addition modulo 2), and the other user publishes $\boldsymbol{M}_2 = \boldsymbol{k}$. Then the message $\boldsymbol{m}$ can be computed as $\boldsymbol{M}_1 \oplus \boldsymbol{M}_2$ by an observer, however the observer cannot identify the sender. Golle & Juels (2004) detail the extension of this protocol to multiple users. DiceMix (Ruffing et al., 2017), which was built on the original DC-net protocol, was proposed as a general decentralized mixing protocol,

providing sender anonymity, by the authors of CoinShuffle. Moreover, built on CoinJoin and DiceMix, CoinShuffle++ was introduced in the same study.

- *Fair Exchange Protocol (1.2.2.4):* Two-party decentralized mixing via a fair exchange protocol can be performed using scripting functionalities of Bitcoin. Barber et al. proposed a Fair Exchange Protocol (Barber et al., 2012), which can be used as a two-party mixing protocol. A cut and choose protocol (Crépeau, 2005), and scripting features of Bitcoin were utilized in the approach which were explained in the study briefly. XIM (Bissias et al., 2014) is another two-party mixing protocol which also allows users to find partners to mix with anonymously. The fair exchange protocol of Barber et al. was utilized in this study; however, the authors state that SMCs of Andrychowicz et al. (Andrychowicz et al., 2014) can also be used. CoinSwap proposed in Bitcoin Forum (https://bitcointalk.org/index.php?topic=321228) is also a two-party mixing protocol proposed by Maxwell. Fair exchange was achieved by utilizing a special transaction type, called *hashlock* transactions. The study of Wijaya et al. (2016) can also be considered as a fair exchange protocol. This protocol requires 5 middlemen in addition to a payer and a payee, totaling 7 participants. Since several transactions are used in a fair exchange protocol, the outcome is breaking links between transactions.

- *Network of Transactions (1.2.2.5):* Coutu (2014) introduced the network of transactions approach, where a network of transactions consists of a number of small two-party switchboxes that are combined in a structured network with the purpose of performing permutation of the addresses. The output of a switchbox is only known the participants of the switchbox since they determine the input and output mapping. This approach is similar to decryption mixnets, however, encryption and decryption operations are not performed. As the outcome, links between input-output addresses in a transaction are broken. CoinLayering (Lu et al., 2022) was proposed as a coin mixing scheme for large scale Bitcoin transactions, based on a *User-Mix-Supervisor* system model.

- *Secure Multiparty Computation (1.2.2.6):* SMC allows a group of users to compute the value of a public function using their private data, while they keep their inputs private. SMC was introduced by Yao (1982). Using SMC for shuffling addresses was first proposed in bitcointalk forum by a member named hashcoin

(https://bitcointalk.org/index.php?topic=12751.msg315793#msg315793). In this proposal, address shuffling is done using a permutation function in SMC, and links between input-output addresses in a transaction are broken. Andrychowicz et al. (2014) proposed SMC on Bitcoin based on the coin-tossing protocol of Blum (1982). SecureCoin (Ibrahim, 2017) utilized secret sharing schemes and SMC in the first aggregation phase, in which each user deposits to a temporary aggregation address before the address shuffling. The use of a set of mixing peers was proposed in CoinParty. CoinParty employs a threshold variant of the ECDSA scheme realized using Damgard et al.'s (2019) protocol for general SMC protocol.

- *Zero Knowledge Proofs (1.2.2.7):* The concept of zero knowledge protocols was introduced by Goldwasser et al. (1989). A zero knowledge proof allows one to prove that a statement is true without giving any other information than the statement is true. Zero Knowledge Contingent Payments (ZKCP) (Campanelli et al., 2017) utilizes hashlock transactions and zero knowledge proofs. Another zero knowledge contingent payment protocol was provided by Banasik et al. (2016). In zero knowledge protocols, mixing is achieved in multiple transactions, therefore links between transactions are broken.

There exist several improvement proposals that require modification, some of them are designed to be used with Bitcoin, whereas some of them are inspired by Bitcoin and designed to be similar to Bitcoin but completely independent, proposed as an alternative Bitcoin-like digital cash system. These are classified as Not Backwards Compatible / Proposed as an Alternative.

Hidden Address Shuffling (2.1): In the hidden address shuffling (2.1), sender and/or receiver Bitcoin address(es) is/are shuffled with other Bitcoin addresses. Moreover, which output address corresponds to which input address remains hidden. The aim is to break traceability of bitcoin flows. Methods in this approach are using blind signatures, ring signatures and composite signatures.

- *Blind Signatures (2.1.1):* Ladd (2012) introduced a new method of forming transactions where blind signatures are used with cut and choose. Modification to the scripting functionalities of Bitcoin, like the addition of a new opcode and a new signature type is required. Blindly Signed Contracts (Heilman et al., 2016), which is the study done prior to TumbleBit, uses blind signatures and smart

contracts to implement a fair exchange protocol. There is a need for adding an opcode that supports elliptic curves with efficient bilinear pairings; thus, modification is required. Darkcoin, which was a privacy-centric cryptographic currency based on Bitcoin, uses a decentralized implementation of CoinJoin, which is called DarkSend. In DarkSend, transactions are merged together into a larger anonymous transaction, resulting breaking links between inputs and outputs addresses in the transaction. Darkcoin was later turned into Dash (https://www.dash.org). In Dash, a chaining approach is adopted to increase anonymity. (Yi & Lam, 2019) proposed a blind signature scheme that allows generating a blind signature compatible with the standard ECDSA to achieve bitcoin transaction anonymity.

- *Ring Signatures (2.1.2):* A ring signature is a special type of a group signature, where there is not any group manager. This signature type was introduced by Rivest et al. (2001). Any member of the group can sign using the ring signature, and the signing member cannot be identified by the ring signature. CryptoNote (Saberhagen, 2013) uses one-time ring signature which is a type of group signature and based on traceable ring signature of Fujisaki & Suzuki (2007). In CryptoNote, for sender privacy, the address of the sender is grouped with other addresses using ring signatures. As the result, links are broken between input and output addresses in a transaction. Bytecoin (https://bytecoin.org) is the first cryptocurrency that used CryptoNote as a base. DigitalNote (https://digitalnote.org), Aeon (https://www.aeon.cash), and Monero (https://www.getmonero.org) are other cryptocurrencies that were implemented using the CryptoNote framework. One of the cryptographic structures that Maxwell's Confidential Transactions (CT) explained in Elements website (https://elementsproject.org/features/confidential-transactions/investigation) use is Borromean ring signatures (Maxwell & Poelstra, 2015). To further improve the privacy provided by Monero, hiding amounts of transactions using a new type of ring signature; multi-layered linkable spontaneous anonymous group signature was proposed. This proposal leveraged Maxwell's approach of CT, which was combined with ring signatures, resulting Ring CT (Noether & Mackenzie, 2016) hiding sender and receiver, as well as hiding amount information. Usage of Franklin and Zhang's unique ring signature protocol (Franklin & Zhang, 2012)

was examined by Mercer (2016) to improve privacy in Bitcoin. Liu et al. (2018) utilized ring signatures, as well.

- *Composite Signatures (2.1.3)*: Composite signatures are extension of aggregate signatures (Boneh et al., 2003). A composite signature combines a number of individual signatures, where there is not any order among them. It allows adding more signatures at any time and it is computationally hard to obtain individual signatures from the composite signature. Saxena et al. (2014) used composite signatures to enhance anonymity in Bitcoin-like cryptocurrencies.

In the ownership shuffling approach (2.2), the ownerships of coins are shuffled. This is achieved by breaking coin and ownership connection, at the same time storing which user owns how many coins. Then, a user can prove that he owns a certain amount of coins and spend them. In this way, the ownerships of the coins are shuffled, and coins cannot be tied to users. Ownership shuffling can be achieved utilizing zero knowledge proofs.

- *Zero knowledge proofs (Transaction breaker & Not hiding amounts) (2.2.1):* Zerocoin (Miers et al., 2013), which was one of the first proposals for improving anonymity in Bitcoin, utilizes zero knowledge proofs. It breaks links between transactions without adding trusted parties. In Zerocoin, bitcoins can be converted to zerocoins and then spending any zerocoin can be achieved by showing the validity of a zerocoin by proving that it belongs to a public list of valid coins. Pinocchio Coin (Danezis et al., 2013) was proposed as a variant of Zerocoin and suggested using elliptic curves and bilinear pairings. Pinocchio (Parno et al., 2013), which is a pairing-based proof system, was utilized in Pinocchio Coin.

- *Zero knowledge proofs (Transaction breaker & Hiding amounts) (2.2.2):* EZC (Androulaki & Karame, 2014), an extension of Zerocoin, was proposed to hide transaction amounts and address balances which Zerocoin cannot achieve since Zerocoin requires zerocoins to be converted back to bitcoins in order to spend them. EZC achieves this by allowing construction of multi-valued zerocoins with values only known the parties in a transaction and spending zerocoins without converting them back to bitcoins. Zerocoin was turned into Zerocash protocol (Ben-Sasson et al., 2014), which is more efficient than Zerocoin. In Zerocash, ZK-SNARKs (Ben-Sasson et al., 2013a) are used to hide inputs, outputs and amount information of a transaction. Zcash (https://z.cash) is the full-fledged ledger based

digital currency which is the implementation of Zerocash protocol. In Z-Channel (Zhang et al., 2019), Zerocash is improved to support multisignature and time lock functionalities resulting in improved scalability and reduced confirmation time for Zerocash payments. Komodo (https://komodoplatform.com) is another protocol that uses zero knowledge proofs of Zcash for hiding sender, receiver and amount information. CoinWitness, which is detailed in Bicoin Forum (https://bitcointalk.org/index.php?topic=277389.0), is another proposal by Maxwell that used ZK-SNARKs to construct compact proofs. Hawk (Kosba et al., 2016) was proposed as a decentralized smart contract system utilizing zero knowledge proofs. CT are also built using zero knowledge proofs. In compact CT (Lukianov, 2015), a short Non-Interactive Zero-Knowledge Proof (NIZKP) is used. Zether (Bünz et al., 2020) was proposed as a fully-decentralized, confidential payment mechanism that is compatible with Ethereum and other smart contract platforms. Zether keeps the account balances encrypted and deposit, transfer and withdrawal of funds to/from accounts are done through cryptographic proofs. In Zether, Σ-Bullets, an improvement of the Bulletproofs (Bünz et al., 2018), are used.

In data encrypting approach (2.3), privacy is preserved by encrypting data. Homomorphic commitments (2.3.1) are used for encryption. Homomorphic commitments allow committing a value without revealing it to the other parties by utilizing homomorphic encryption technique. A homomorphic encryption scheme allows performing computations on ciphertext where the decryption of result gives a value that is equal to the result of operations performed on plain text (Yi et al., 2014). This technique was first proposed by Back in Bitcoin Forum (https://bitcointalk.org/index.php?topic=305791.0). CT utilized cryptographic technique of additively homomorphic commitments for Bitcoin, inspired by Back's proposal that is detailed in Bitcoin Forum (https://bitcointalk.org/index.php?topic=305791.0). Pedersen commitments are the basic structure that CT is based on. CT are used in Elements project by Blockstream (https://blockstream.com/elements). Elements project includes the usage of side chains, which are extensions to existing blockchains, in order to add new features like smart contracts and CT in order to improve privacy and functionality. Lukianov (2015) proposed a compact version of CT using elliptic point commitments (Boudot, 2000). As in Maxwell's original proposal, homomorphic commitments are used to hide transaction

amounts and ensure that the sum of the transaction inputs matches the sum of its outputs. Ring CT (Noether & Mackenzie, 2016) are the combination of Maxwell's approach of CT with ring signatures. Ring CT are included in Monero (https://www.getmonero.org), resulting combination of outputs of both methods, i.e., hiding amounts and input-output address links. Wang et al. (2020) proposed a framework to hide the amounts by employing the Paillier cryptosystem for encryption and decryption.

In data disintegrating approach (2.4), data is disintegrated and stored in blockchain partially. Data that is not stored in blockchain remains off-chain. For instance, the blockchain may store some transactions, and remaining transactions may stay between only sender and receiver. Another example is using blockchain for storing only hash of the transactions. Thus, we name the method used in this approach as off-chain storage. Data that will be kept off-chain is up to the design and the additional methods used.

- *Off-chain storage (2.4.1):* In this approach, all transaction data are not stored in blockchain, but some data are stored off-chain. Utilizing off-chain storage results in improved scalability, broken links between transactions and hidden amounts. Maxwell's Maxwell proposed CoinWitness in Bitcoin Forum (https://bitcointalk.org/index.php?topic=277389.0) using ZK-SNARKs (Ben-Sasson et al., 2013a) to construct proofs of correctness which shows a side chain payment is valid. In Hawk (Kosba et al., 2016), transactions are not stored with full financial data in the blockchain. Cash flows and transaction amounts are hidden in the private contracts; therefore, hidden from the public view. In another proposal, tonych, a user in Bitcoin Forum (https://bitcointalk.org/index.php?topic =1574508.0), showed that all transaction data can be hidden by just using hashing, and no other cryptographic structures. In this approach, only hash of inputs and outputs are stored in blockchain. Obyte (https://obyte.org) is the first coin that implemented this approach. Another off-chain scheme, which uses micropayment channel networks was provided by Heilman et al. (2016). In this scheme, some transactions are not recorded in blockchain and stay only between sender and receiver upon establishing a pairwise micropayment channel after forming an escrow transaction. Quantum Bitcoin (Jogenfors, 2016) is a proposal of a Bitcoin-like currency that runs on a quantum computer and based on no-cloning theorem (Wootters & Zurek, 1982) of quantum mechanics. In Quantum Bitcoin, local transactions that are only between sender and receiver are used and no-cloning

theorem acts as a copy-protection mechanism to prevent double-spending. Eberhardt & Tai (2018) proposed ZoKrates, a processing model which employs non-interactive zero knowledge proofs to off-chain computations. ZeroCross was proposed (Li et al., 2022) as a sidechain-based privacy-preserving cross-chain solution for Monero. Erdin et al. (2021) proposed an off-chain solution where retailers create a private payment channel network among them to serve their business needs.

3.2 Application of Differential Privacy to Financial Distributed Ledger Applications

Bitcoin is the first financial distributed ledger application. Generally used methods for anonymity and privacy improvement in Bitcoin-like systems rely on mostly cryptographic protocols. However, the Bitcoin blockchain may benefit from differential privacy, which is not only light-weight but also easy to understand and implement compared to cryptographic protocols. Hiding actual transaction amounts by adding noise can be a way of applying differential privacy. Perturbing actual data with noise makes anonymization and privacy breaches by direct queries to the blockchain impossible.

To the best of our knowledge, there is no study on the examination of Bitcoin in terms of differential privacy in the literature. There are studies combining differential privacy and blockchain mostly in general areas. Privacy-preserving solutions for general blockchain structure were studied in (Bernabe, 2019), and differential privacy was mentioned as a potential solution very briefly. Differential privacy was used in (Duan et al., 2019) while aggregating crowd data via blockchain by a service provider before sharing it with a data consumer. Differentially private machine learning models via blockchain were studied in (Chen et al., 2018; Hynes et al., 2018). Differential privacy was used in (Yang et al., 2018) to obfuscate the results of statistical queries in a differentially private blockchain-based data-sharing model.

The utilization of differential privacy in financial blockchain-based systems for improving anonymity and privacy has recently begun to be considered. Digital currency and international money transfers are considered areas as future applications of differential privacy in blockchain (Hassan et al., 2020c). Correspondingly, inspired by

Monero, an approach for a cryptocurrency utilizing differential privacy was introduced in Zcash Foundations Github page (https://github.com/ZcashFoundation/GrantProposals-2018Q2/issues/36) as a proposal to Zcash Foundation, and granted; however, there is no follow-up study that details and verifies the approach as of this writing. The addition of noise to transaction amounts in the Ethereum blockchain and analysis according to the Eigen centrality measure was done in (Kumar, 2020). The implementation was done in R using relevant network packages, and January 2019 blockchain transaction data (1551 transactions) obtained from the Etherscan website (https://etherscan.io) was used in the study. A graph structure was formed using these transactions, and the most central nodes were detected before and after adding Gaussian noise to transaction amounts respectively. It was shown that the central nodes changed when the noises were added. The motivation for using centrality comes from the idea that more central nodes are at higher risk of being attacked, therefore, preserving privacy for these nodes is important. In this model, the noise addition is done by dedicated and distributed servers before publishing the transactions online. The actual transaction amounts can be accessed through these servers by authenticated users. The Gaussian parameters were determined trial and error, ε was determined as 0.9, and the delta (δ) was determined as 0.4. This study did not examine other differential privacy mechanisms, nor gave the results for different Gaussian parameter values.

In (Hassan et al., 2020b), four variants of differential privacy mechanisms (Laplace, Gaussian, Uniform, and Geometric) were tested in decentralized blockchain-based smart metering. In this system, smart meters act as blockchain nodes sending their real-time data plus noises generated via differential privacy mechanisms to grid utility databases. The grid energy data (Muratori, 2018) was modified accordingly to carry out an experiment for 24-hour usage. The evaluation was carried out on 144 data values ranging between 200 and 1900. For the implementation, Python libraries NumPy v1.14 and pandas v1.0.3 libraries were used. The Laplace, the Gaussian, and the Geometric mechanisms were compared using different ε values (ε = 0.01, 0.05, 0.1, 0.3, 0.7, and 1), and the same values are used for δ in the Uniform mechanism. The evaluation was done according to Mean Absolute Error (MAE). MAE is calculated by summing absolute differences between the noisy values and the original readings, and taking the mean. Graphs, showing the original and protected readings, were generated at the stated ε and δ values for the mechanisms. The results showed that the mechanisms provide high privacy

by adding a large amount of noise when ε or δ is low (ε, δ = 0.01), and the privacy reduces gradually as ε or δ increases. Among these four mechanisms, the Geometric and the Laplace are found to be performing better at lower ε values by adding a sharp amount of noise, resulting in higher MAEs. Specifically, the Geometric mechanism is found to be more suitable for protecting high peak values (e.g., high usage), and the Laplace mechanism is found to be more suitable for protecting low peak values (e.g., low usage) at ε = 0.01. It was stated that an adequate amount of noise is added when ε, δ = 0.01 and 0.05, to protect privacy, and ε, δ = 0.01 were declared as the most suitable privacy parameters. The MAE values for ε, δ > 0.05 were not provided in the study.

Blockchain structure is utilized for managing smart metering data, as well. (Mollah et al., 2019; Andoni et al., 2019; Farokhi, 2020; Hassan et al, 2020a; Marks et al., 2021; Aklilu & Ding, 2021) surveyed the related literature. We researched the literature on smart metering with privacy and blockchain keywords. Specifically, we focused on differential privacy than the general privacy notion. We categorize the related work into four categories as presented in Table 3.2.

In C1, extensive amount of literature is presented on smart metering using encryption, so that only the utility service provider knows the actual consumption values of subscribed smart homes. Some example studies are (Saputo & Akkaya, 2012; Li et al., 2010; Ruj & Nayak, 2013) utilizing homomorphic encryption. One can find many studies in the literature in this category.

C2 includes studies that use blockchain infrastructure in smart metering scenario. Distributed ledger structures can be examined in this category. As the sample studies; (Bürer et al., 2019) provided use cases, (Dorri et al., 2019; Eisele et al., 2020; Aitzhan & Svetinovic, 2018; Knirsch et al., 2018; Eberhardt et al., 2020) studied on utilization of blockchain in energy trading. (Abdo & Zeadally, 2020) included mutiple utilities in their blockchain-based utility market solution. (Firoozjaei et al., 2020) proposed a hybrid blockchain solution for energy transactions with subnetworks. (Wang et al., 2019) combined blockchain with homomorphic encryption to protect metering data during the aggregation process.

Table 3.2: Categorization of the studies on smart metering, privacy, and blockchain

Category Code	Definition
C1	Provides a solution on smart metering with improved privacy (other than differential privacy)
C2	Provides a solution on smart metering using blockchain
C3	Provides a solution on smart metering with differential privacy
C4	Provides a solution on smart metering using blockchain with differential privacy

C3 includes studies that use differential privacy in smart metering scenario without blockchain utilization. (Zhao et al., 2014) studied differential privacy in the Battery-based Load Hiding (BLH) problem. (Backes & Maiser, 2013; Liu et al., 2021; Hossain et al., 2021; Kserawi et al., 2022) studied achieving differential privacy with rechargeable batteries. (Acs & Castelluccia, 2012; Bao & Lu, 2015; Barbosa et al., 2016; Eibl & Engel, 2017; Ni et al., 2017; Hassan et al., 2019; Zheng et al., 2021; Gough et al., 2022) are some of the other studies in this category.

Studies in C4 utilizes both blockchain and differential privacy. The number of studies in this category is lower compared to the other categories. (Gai et al., 2019) brought together blockchain and differential privacy with energy trading. (Hassan et al., 2020d) proposed differentially private auction for blockchain based microgrids energy trading. (Fotiou et al., 2021) proposed a privacy-preserving statistics marketplace using local differential privacy and blockchain. They utilized randomized responses (Erlingsson et al., 2014). (Hassan et al., 2020b) and (Guan et al., 2018) are the other prominent studies in this category.

4. INVESTIGATION AND APPLICATION OF DIFFERENTIAL PRIVACY IN BITCOIN

One can infer that Bitcoin does not provide differential privacy by a pragmatic approach since the presence of a Bitcoin address is explicit in the public Bitcoin blockchain. Although real names are not paired with Bitcoin addresses, addresses can be related to user identities using off-network information. Another argument supporting Bitcoin is not differentially private is the explicitness of transaction amounts and whether a transaction occurred between two specific addresses in the public blockchain. It is worth examining Bitcoin in terms of differential privacy theoretically to confirm these arguments.

The formulation of differential privacy, given as Formula 2.1, has to be checked to examine Bitcoin in terms of differential privacy theoretically, and finding a counterexample to 2.1 suffices to detect a violation of differential privacy. In the case of Bitcoin, a set of transactions in the blockchain can be considered as a dataset. In the following subsections, we check the formula for four functions querying; (i) transactions between two specific addresses, (ii) transactions above a specific amount, (iii) transactions for a specific transaction amount, (iv) transactions with a specific amount between two specific addresses, as given in Figure 4.1. These functions are chosen in the analysis since they can be used for exploiting information from the public blockchain for detecting addresses and deanonymizing users. We present our theoretical examination in Section 4.1.

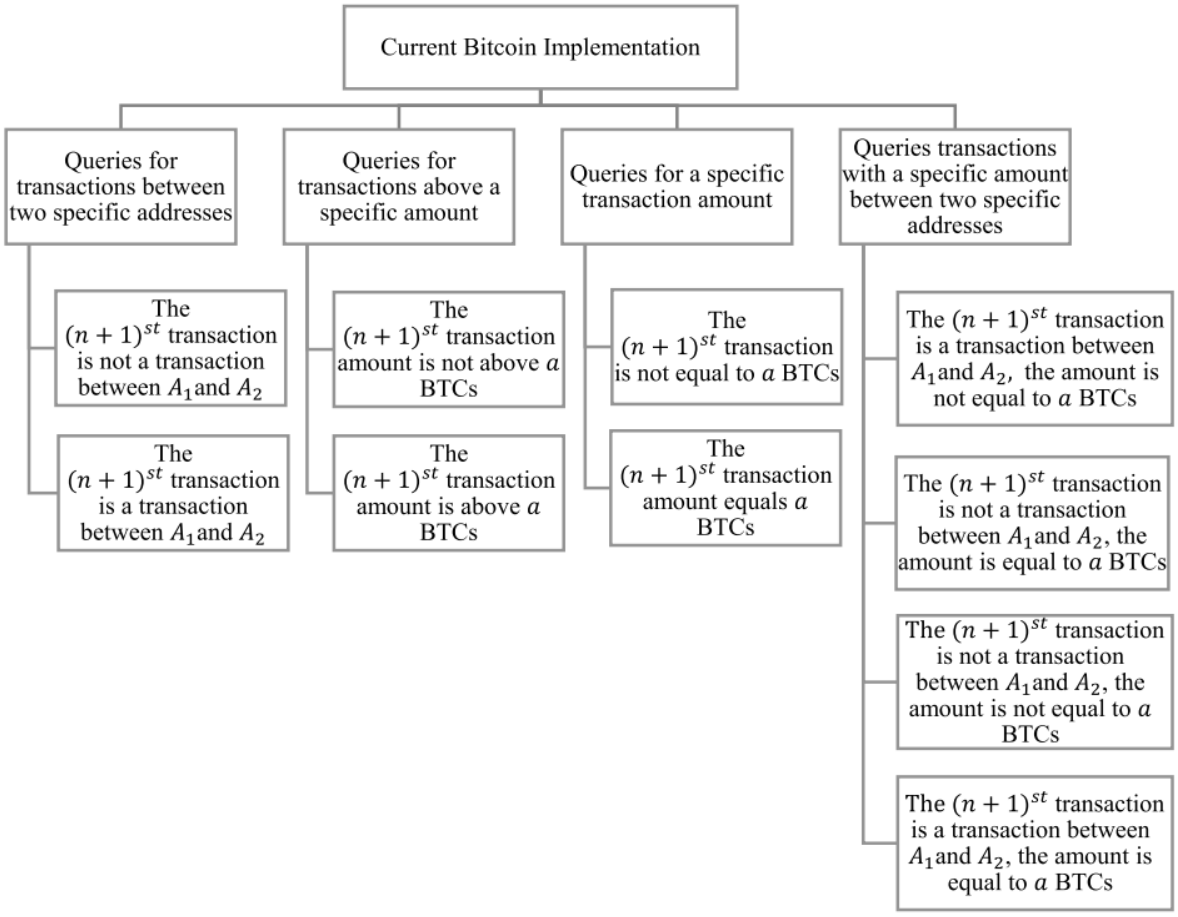


Figure 4.1: Examined cases for the investigation of current Bitcoin implementation from the differential privacy perspective

After examining the current Bitcoin implementation, we investigate the effects of applying differential privacy mechanisms as shown in Figure 4.2. We present our solutions in Section 4.2 and 4.3.

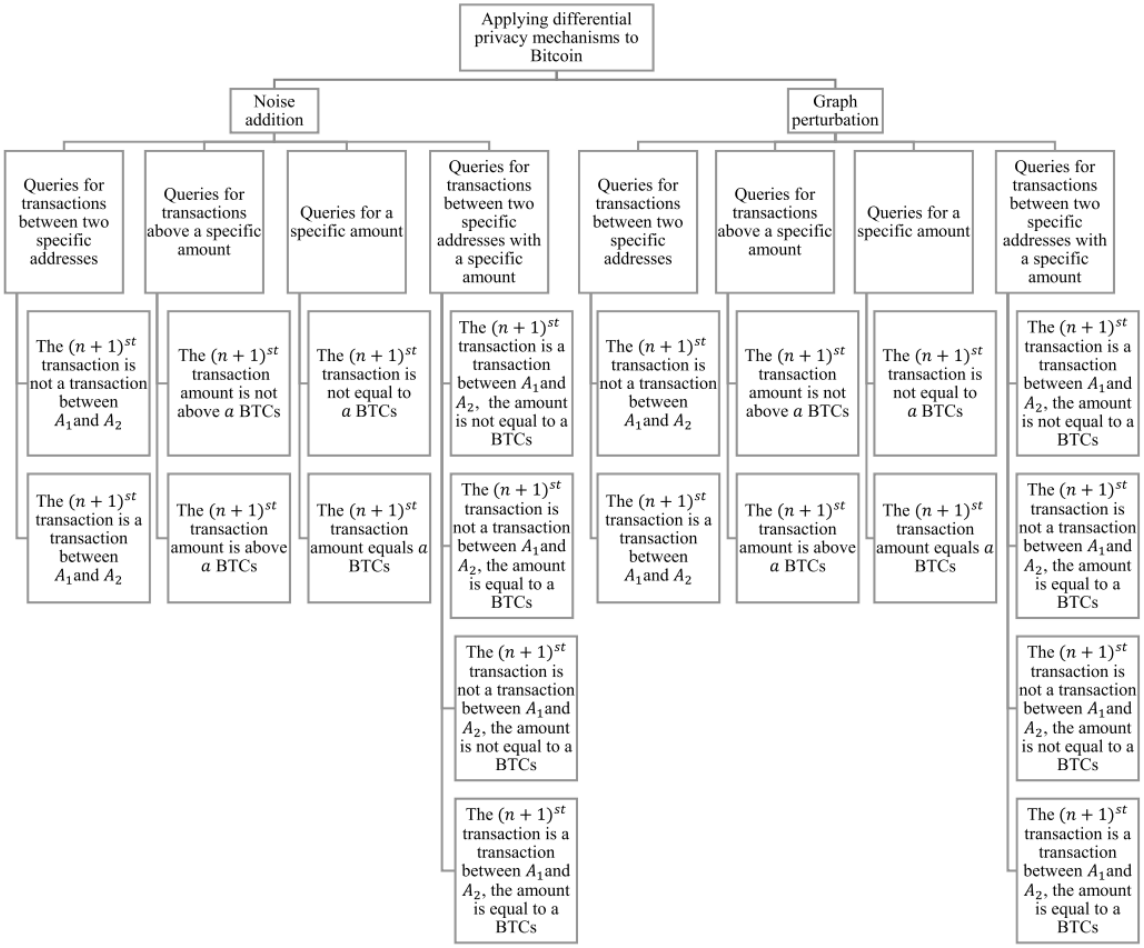


Figure 4.2: Examined cases for the investigation of Bitcoin from the differential privacy perspective with the application of differential privacy mechanisms

After examining theoretically, we demonstrate a practical utilization of a differential privacy approach in Bitcoin in an empirical way in Section 4.4. We add noise to the Bitcoin transaction amounts by applying the Laplace, the Gaussian, the Geometric, and the Uniform mechanisms for the noise generation at different ε values, and evaluate the results. In Section 4.5, we summarize our research and observations for investigation and application of differential privacy in Bitcoin.

4.1 Theoretical Examination of Bitcoin from Differential Privacy Perspective

We provide theoretical examination of Bitcoin from differential privacy perspective for four query functions in the following subsections.

4.1.1 Queries for Transactions Between Two Specific Addresses

Assume that one wishes to learn whether a transaction occurred between two specific Bitcoin addresses. Let A_1 and A_2 denote the addresses and F be a function that gives the average transaction amount between A_1 and A_2. Let D_1 consists of $n+1$ transactions and D_2 consists of n transactions which are exactly the same as the first n transactions of D_1, which makes D_1 and D_2 differ in a single row. The range of F is between 0 and 21×10^6 BTCs (the maximum number of bitcoins that will ever exist) theoretically. The sensitivity of this function is 21×10^6 divided by the number of transactions in the blockchain. To cover all possible datasets, two cases must be considered; (i) the $(n+1)^{st}$ transaction is not a transaction between A_1 and A_2, (ii) the $(n+1)^{st}$ transaction is a transaction between A_1 and A_2. The two cases for D_1 and D_2 can be visualized as in Figure 4.3. The $(n+1)^{st}$ transaction states, relations between $F(D_1)$ and $F(D_2)$, differential privacy provision or violation statuses in these cases are given in Table 4.1, where a_{x+1} denotes the $(n+1)^{st}$ transaction amount.

In the first case, $F(D_1)$ equals $F(D_2)$, and the differential privacy formula given in 2.1 is true for all subsets and ε values. For the second case, $F(D_1)$ equals $F(D_2)$ plus some value that comes from the $(n+1)^{st}$ transaction. The minimum amount that can be transferred in a Bitcoin transaction is 0.00000546. Let S be $[F(D_2) + (0.00000546/(n+1)), 21\ x\ 10^6]$. The Formula 2.1 turns into Formula 4.1 with these values.

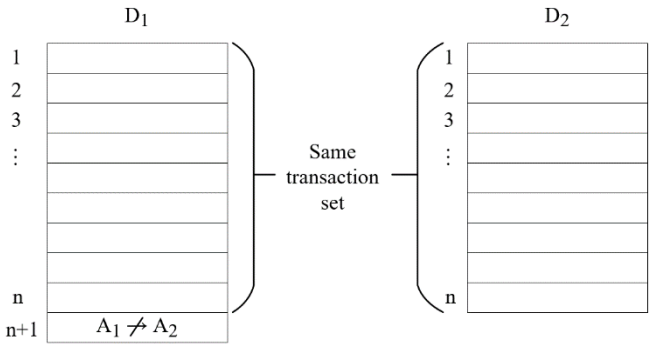
$$P\left[F(D_1) \in [F(D_2) + (\frac{0.00000546}{n+1}), 21 \times 10^6]\right] \leq$$

$$\exp(\varepsilon) \times P\left[F(D_2) \in [F(D_2) + (\frac{0.00000546}{n+1}), 21 \times 10^6]\right] \quad (4.1)$$

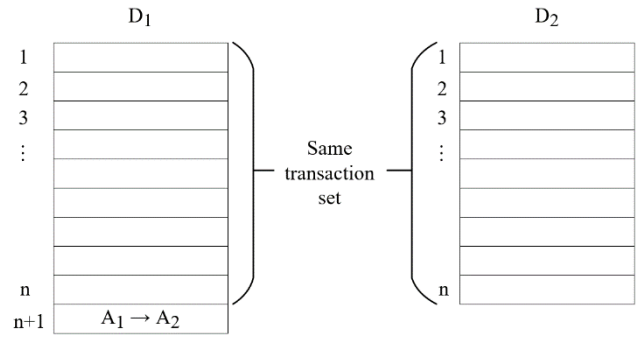
In this formula, $P\left[F(D_1) \in [F(D_2) + (0.00000546/(n+1)),\ 21 \times 10^6]\right]$ equals 1, $P\left[F(D_2) \in [F(D_2) + (0.00000546/(n+1)), 21 \times 10^6]\right]$ equals 0, and Formula 4.1 turns into 4.2.

$$1 \leq \exp(\varepsilon) \times 0 \quad (4.2)$$

Since Formula 4.2 is false for all ε values, this is a violation of differential privacy. This means that there is no differential privacy for a transaction between two Bitcoin addresses in $1/2$ of the cases considered.



(a)



(b)

Figure 4.3: Two transaction datasets that differ in a single transaction; (a) The $(n+1)^{st}$ transaction is not a transaction between A_1 and A_2; (b) The $(n+1)^{st}$ transaction is a transaction between A_1 and A_2

Table 4.1: Cases considered in differential privacy evaluation of the queries for transactions between two specific addresses

Case	$(n+1)^{st}$ Transaction	$F(D_1)$ and $F(D_2)$	Differential Privacy
i	$A_1 \nrightarrow A_2$	$F(D_1) = F(D_2)$	✓
ii	$A_1 \rightarrow A_2$	$\mathrm{F(D_1)} = \mathrm{F(D_2)} + \mathrm{a_{x+1}}/(n+1)$	X

4.1.2 Queries for Transactions Above a Specific Amount

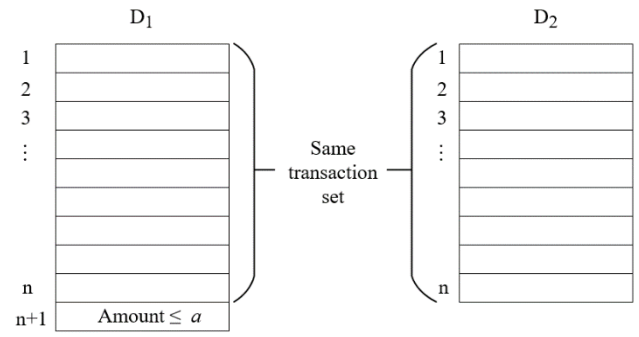
As a second examination, assume that one wishes to learn whether a transaction with an amount above a BTCs occurred. Let F be a function that gives the number of transactions having an amount above a BTCs in the blockchain. The sensitivity of this function is 1, since adding a single row to any dataset will change the output by at most 1. Let D_1 consists of $n+1$ transactions and D_2 consists of n transactions that are exactly the same as the first n transactions of D_1. To cover all possible datasets, two cases must be considered; (i) the $(n+1)^{st}$ transaction amount is not above a BTCs, (ii) the $(n+1)^{st}$ transaction amount is above a BTCs. The two cases for D_1 and D_2 can be visualized as in Fig. 4.4. The $(n+1)^{st}$ transaction states, relations between $F(D_1)$ equals $F(D_2)$; differential privacy provision or violation statuses in these cases are given in Table 4.2.

In the first case, $F(D_1)$ equals $F(D_2)$, and the differential privacy formula given in 2.1 is true for all subsets and ε values. For the second case, $F(D_1)$ equals $F(D_2)+1$. Consider the case when $F(D_2)$ equals 0, i.e., there is no transaction with an amount above a. In this case, $F(D_1)$ equals 1. The range of F is [0, $n+1$] for D_1 and $[0, n]$ for D_2. Let S be [1, n]. F is ε-differential private if the following holds.

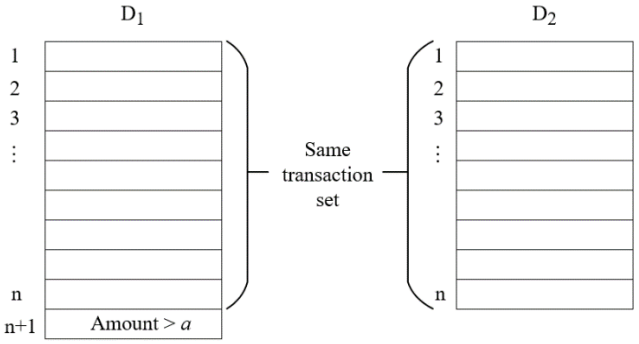
$$P\left[F(D_1) \in [1,n]\right] \leq \exp(\varepsilon) \times P\left[F(D_2) \in [1,n]\right] \tag{4.3}$$

Since $F(D_1)$ equals 1, $P\left[F(D_1) \in [1,n]\right]$ equals 1, and since $F(D_2)$ equals 0, $P\left[F(D_2) \in [1,n]\right]$ equals 0, Formula 4.3 turns into 4.4, which is false for all ε values, showing a violation of differential privacy. This means that differential privacy is violated for transactions having an amount above a specific value in $1/2$ of the cases considered.

$$1 \leq \exp(\varepsilon) \times 0 \tag{4.4}$$



(a)



(b)

Figure 4.4: Two transaction datasets that differ in a single transaction; (a) The $(n+1)^{st}$ transaction amount is not above a BTCs; (b) The $(n+1)^{st}$ transaction amount is above a BTCs

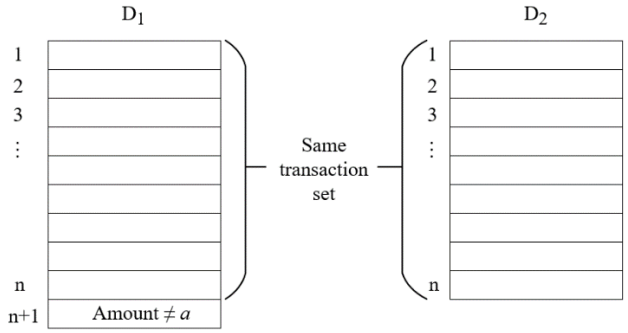
Table 4.2: Cases considered in differential privacy evaluation of the queries for transactions above a specific amount

Case	$(n+1)^{st}$ Transaction	$F(D_1)$ and $F(D_2)$	Differential Privacy
i	Amount $\leq a$	$F(D_1) = F(D_2)$	✓
ii	Amount > a	$F(D_1) = F(D_2) + 1$	X

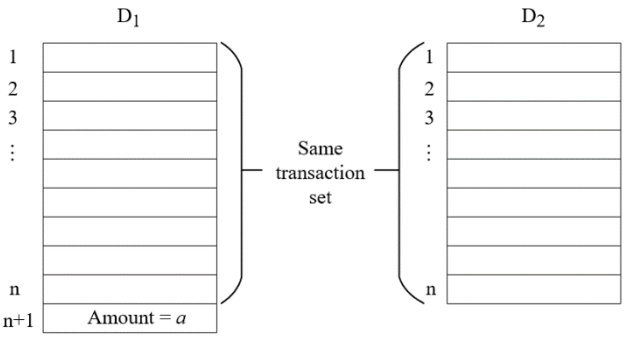
4.1.3 Queries for a Specific Amount

A question that comes to mind might be “What happens if blockchain was sought for transactions with a specific amount?”. Pragmatically, it can be said that transactions transferring a specific amount and related Bitcoin addresses can be detected easily from the public blockchain structure. However, a theoretical examination is required to confirm

these arguments. Therefore, as the last examination, we evaluate this query theoretically in terms of differential privacy. Assume that one wishes to learn whether there is a transaction with an amount equal to a BTCs. Let F be a function that gives the number of transactions that have an amount equal to a BTCs in the blockchain. The sensitivity of this function is 1, as well. Let D_1 consists of $n+1$ transactions and D_2 consists of n transactions that are exactly the same as the first n transactions of D_1. Again, to cover all possible datasets, two cases must be considered; (i) the $(n+1)^{st}$ transaction amount is not equal to a BTCs, (ii) the $(n+1)^{st}$ transaction amount equals a BTCs. The two cases for D_1 and D_2 can be visualized as in Figure 4.5. The $(n+1)^{st}$ transaction states, relations between $F(D_1)$ and $F(D_2)$, differential privacy provision or violation statuses in these cases are given in Table 4.3.



(a)



(b)

Figure 4.5: Two transaction datasets that differ in a single transaction; (a) The $(n+1)^{st}$ transaction amount is not equal to a BTCs; (b) The $(n+1)^{st}$ transaction amount equals a BTCs

In the first case, $F(D_1)$ equals $F(D_2)$, and the differential privacy formula given in 2.1 is true for all subsets and ε values. In the second case, $F(D_1)$ equals $F(D_2)+1$. Consider the case when $F(D_2)$ equals 0, i.e., no transaction amount is equal to a BTCs. In this case, $F(D_1)$ equals 1. The range of F is $[0, n+1]$ for D_1 and $[0, n]$ for D_2. For S is [1,

n], there is a violation of differential privacy as shown in the previous query. Again, this means that there is no differential privacy for a specific transaction amount $^1/_2$ of the cases considered.

Table 4.3: Cases considered in differential privacy evaluation of the queries for a specific amount

Case	$(n+1)^{st}$ Transaction	$F(D_1)$ and $F(D_2)$	Differential Privacy
i	Amount $\neq a$	$F(D_1) = F(D_2)$	✓
ii	Amount = a	$F(D_1) = F(D_2) + 1$	X

4.1.4 Queries for Transactions with a Specific Amount Between Two Addresses

Assume that one wishes to learn whether a transaction with an amount a occurred between two specific Bitcoin addresses. Let A_1 and A_2 denote the addresses and F be a function that gives the number of transactions between A_1 and A_2 that have an amount equal to a BTCs. This query function is basically the combination of the query functions examined in Section 4.1.1 and Section 4.1.3. Let D_1 consists of $n+1$ transactions and D_2 consists of n transactions which are exactly the same as the first n transactions of D_1, which makes D_1 and D_2 differ in a single row. To cover all possible datasets, four cases must be considered; (i) the $(n+1)^{st}$ transaction is a transaction between $A_1$and A_2, the amount is not equal to a BTCs, (ii) the $(n+1)^{st}$ transaction is not a transaction between $A_1$and A_2, the amount is equal to a BTCs, (iii) the $(n+1)^{st}$ transaction is not a transaction between $A_1$and A_2, the amount is not equal to a BTCs, (iv) the $(n+1)^{st}$ transaction is a transaction between $A_1$and A_2, the amount is equal to a BTCs. The $(n+1)^{st}$ transaction states, relations between D_1 and D_2, differential privacy provision or violation statuses in these cases are given in Table 4.4.

Table 4.4: Cases considered in differential privacy evaluation of the queries for transactions with a specific amount between two specific addresses

Case	$(n+1)^{st}$ Transaction	D_1 and D_2	Differential Privacy
i	$A_1 \rightarrow A_2$ Amount $\neq a$	$F(D_1) = F(D_2)$	✓
ii	$A_1 \nrightarrow A_2$ Amount = a	$F(D_1) = F(D_2)$	✓
iii	$A_1 \nrightarrow A_2$ Amount $\neq a$	$F(D_1) = F(D_2)$	✓
iv	$A_1 \rightarrow A_2$ Amount = a	$F(D_1) = F(D_2) + 1$	X

In the first three cases, $F(D_1)$ equals $F(D_2)$ since the $(n+1)^{st}$ transaction is not a transaction between A_1 and A_2 that have an amount equal to a BTCs. As a result, the differential privacy formula given in 2.1 is true for all subsets and ε values for these cases. In the fourth case, $F(D_1)$ equals $F(D_2) + 1$. Consider the case when $F(D_2)$ equals 0, i.e., this means that there is no transaction between A_1 and A_2 with an amount a. In this case, $F(D_1)$ equals 1. The range of F is $[0, n+1]$ for D_1 and $[0, n]$ for D_2. For S is $[1, n]$, there is a violation of differential privacy according to the differential privacy formulation. This means that there is no differential privacy for transactions between two specific Bitcoin addresses with a specific amount in $1/4$ of the cases considered.

4.2 Feasibility of the Utilization of Noise Addition to Bitcoin Transaction Amounts

One way of utilizing differential privacy for improving privacy in Bitcoin may be the addition of Laplace noise to the transaction amounts while including transactions in the blockchain, as a local differential privacy application. This change clearly requires a modification of the Bitcoin transaction verification mechanism, as well. However, this study focuses on the examination of applying differential privacy mechanisms and results in terms of satisfying differential privacy; we leave the actual implementation of such a verification mechanism, and examination of the utility of noise added transaction amounts as a future study. In the following subsections, we examine the effect of noise addition on differential privacy for the four query functions, which were examined in Section 4.1.

4.2.1 Effect of Noise Addition on Queries for Transactions Between Two Specific Addresses

Consider the function in Section 4.1.1 provided as an example, where the existence of a transaction between two specific Bitcoin addresses, A_1 and A_2, is sought, and F is a function that gives the average transaction amount between A_1 and A_2. D_1 consists of $n+1$ transactions and D_2 consists of n transactions which are exactly the same as the first n transactions of D_1. Assume that the blockchain stores transactions with noise values generated according to the Laplace mechanism added to the transaction amounts. Moreover, assume that noise values are added accordingly so that the minimum and the maximum Bitcoin transaction amounts do not change, stay as 0.00000546 and 21×10^6 BTCs respectively.

The noise values that will be added can be calculated using the noise distribution function and the sensitivity of the query function. Again, the range of F is between 0 and 21×10^6 BTCs since even in the nonexistence of at least one transaction between A_1 and A_2, the average transaction amount is still 0. To cover all possible datasets, again, two cases must be considered; (i) the $(n+1)^{st}$ transaction is not a transaction between A_1 and A_2, (ii) the $(n+1)^{st}$ transaction is a transaction between A_1 and A_2. The two cases for D_1 and D_2 after the noise addition can be visualized as in Figure 4.6. These cases and the corresponding $(n+1)^{st}$ transaction states, relations between $F(D_1)$ and $F(D_2)$, differential privacy provision or violation statuses after the noise addition are given in Table 4.5, where $\mathrm{a}_{\mathrm{x}+1}$ denotes the $(n+1)^{st}$ transaction amount.

In the first case, $F(D_1)$ equals $F(D_2)$ after the noise addition, and the differential privacy formula given in 2.1 is true for all subsets and ε values. For the second case, again, $F(D_1)$ equals $F(D_2)$ plus some value that comes from the noise added $(n+1)^{st}$ transaction. The minimum amount that can be transferred in a Bitcoin transaction is still 0.00000546. For S is $[F(D_2) + (0.00000546/(n+1)), 21 \times 10^6]$, a violation of differential privacy can be shown as in Section 4.1.1 in $1/2$ of the cases considered for this query. Thinking pragmatically, it can be inferred that adding noise to transaction amounts does not hide the existence of a transaction between two specific addresses at any level, as well.

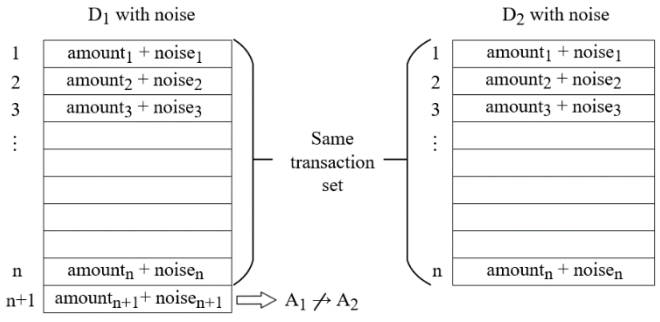


Figure 4.6: Two transaction datasets that differ in a single transaction after the noise addition; (a) The $(n+1)^{st}$ transaction is not a transaction between A_1 and A_2; (b) The $(n+1)^{st}$ transaction is a transaction between A_1 and A_2

Table 4.5: Cases considered in differential privacy evaluation of the queries for transactions between two specific addresses with noise addition

Case	$(n+1)^{st}$ Transaction	$F(D_1)$ and $F(D_2)$ After Noise	Differential Privacy After Noise
i	$A_1 \nrightarrow A_2$	$F(D_1) = F(D_2)$	✓
ii	$A_1 \rightarrow A_2$	$\mathrm{F(D_1) = F(D_2) + a_{x+1}/(n+1)}$	X

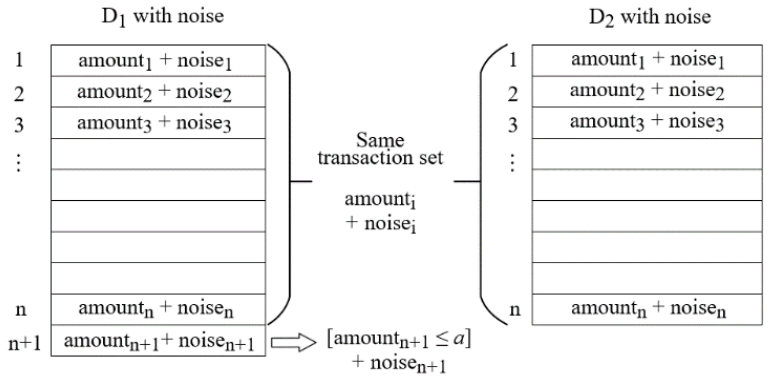
4.2.2 Effect of Noise Addition on Queries for Transactions Above a Specific Amount

Consider the function in Section 4.1.2, where one wishes to learn whether a transaction with an amount above a BTCs occurred and F is a function that gives the number of transactions greater than a BTCs in the blockchain. Again, let D_1 consists of $n+1$ transactions, and D_2 consists of n transactions which are exactly the same as the first n transactions of D_1. To cover all possible datasets, two cases must be considered again; (i) the $(n+1)^{st}$ transaction amount is not above a BTCs, (ii) the $(n+1)^{st}$ transaction amount is above a BTCs. The two cases for D_1 and D_2 after the noise addition can be

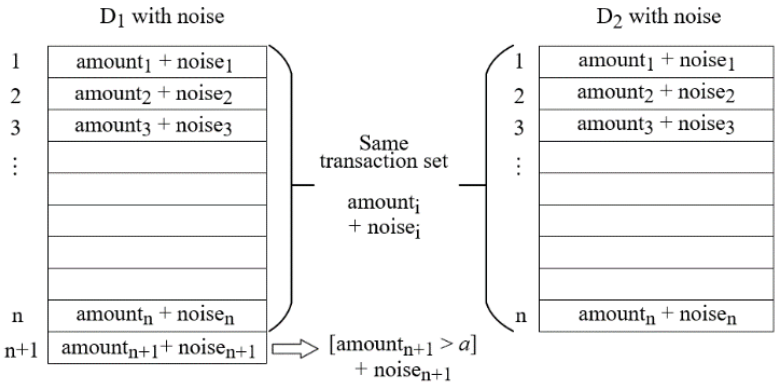
visualized as in Figure 4.7. These cases and the corresponding $(n+1)^{st}$ transaction states, relations between $F(D_1)$ and $F(D_2)$, differential privacy provision or violation statuses after the noise addition are given in Table 4.6.

Table 4.6: Cases considered in differential privacy evaluation of the queries for transactions above a specific amount with noise addition

Case	$(n+1)^{st}$ Transaction Before Noise	$(n+1)^{st}$ Transaction After Noise	$F(D_1)$ and $F(D_2)$ After Noise	Differential Privacy After Noise
i	Amount $\leq a$	Amount $\leq a$	$F(D_1) = F(D_2)$	✓
		Amount $> a$	$F(D_1) = F(D_2) + 1$	X
ii	Amount $> a$	Amount $\leq a$	$F(D_1) = F(D_2)$	✓
		Amount $> a$	$F(D_1) = F(D_2) + 1$	X



(a)



(b)

Figure 4.7: Two transaction datasets that differ in a single transaction after the noise addition; (a) The $(n+1)^{st}$ transaction amount is not above a BTCs; (b) The $(n+1)^{st}$ transaction amount is above a BTCs

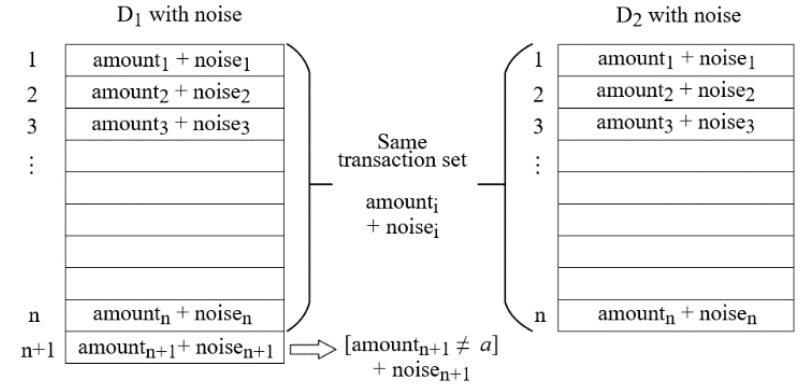
In the first case, there are two possible outcomes. $F(D_1)$ may be equal to $F(D_2)$ after the noise addition if the amount remains not above a. In this situation, the differential privacy formula given in 2.1 is true for all subsets and ε values. If the amount gets greater than a, $F(D_1)$ gets equal to $F(D_2)+1$. For the second case, there are two possible outcomes, as well. $F(D_1)$ may be equal to $F(D_2)$ if a negative noise is added to the $(n+1)^{st}$ transaction, which results in a transaction amount below a BTCs and true for the differential privacy formula given in 2.1 for all subsets and ε values. Alternatively, $F(D_1)$ may be equal to $F(D_2)+1$, if a positive noise is added to the $(n+1)^{st}$ transaction, which results in a violation of differential privacy as shown in Section 4.1.2. The differential privacy is violated for this query in $^2/_4$ of the cases considered.

4.2.3 Effect of Noise Addition on Queries for Transactions for a Specific Amount

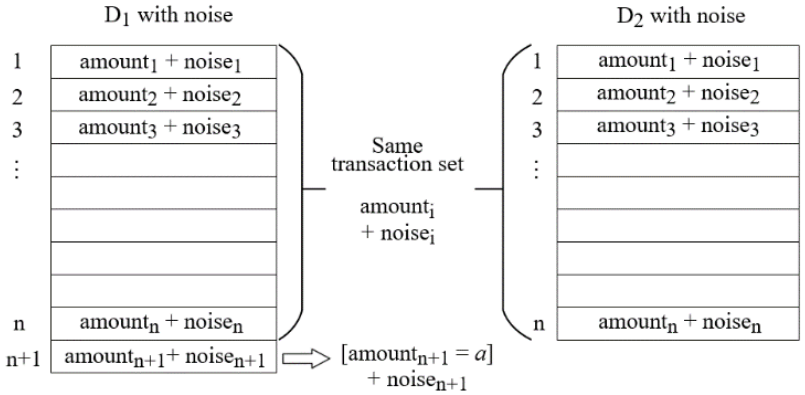
Consider the function in Section 4.1.3, where one wishes to learn whether a transaction with an amount equal to a BTCs occurred and F is a function that gives the number of transactions with the amount a in the blockchain. Let D_1 and D_2 be two neighbor datasets that consist of exactly the same n transactions and D_1 has an additional $(n+1)^{st}$ transaction. For this query function, two cases must be considered to cover all possible datasets; (i) the $(n+1)^{st}$ transaction amount is not equal to a BTCs, (ii) the $(n+1)^{st}$ transaction amount is equal to a BTCs. The two cases for D_1 and D_2 after the noise addition can be visualized as in Figure 4.8. These cases and the corresponding $(n+1)^{st}$ transaction states, relations between $F(D_1)$ and $F(D_2)$, differential privacy provision or violation statuses after the noise addition are given in Table 4.7.

Table 4.7: Cases considered in differential privacy evaluation of the queries for transactions with a specific amount with noise addition

Case	$(n+1)^{st}$ Transaction Before Noise	$(n+1)^{st}$ Transaction After Noise	$F(D_1)$ and $F(D_2)$ After Noise	Differential Privacy After Noise
i	Amount ≠ a	Amount ≠ a	$F(D_1) = F(D_2)$	✓
		Amount = a	$F(D_1) = F(D_2)+1$	X
ii	Amount = a	Amount ≠ a	$F(D_1) = F(D_2)$	✓



(a)



(b)

Figure 4.8: Two transaction datasets that differ in a single transaction after the noise addition; (a) The $(n+1)^{st}$ transaction amount is not equal to a BTCs; (b) The $(n+1)^{st}$ transaction amount is equal to a BTCs

In the first case, two outcomes can occur after the noise addition; $(i.i)(n+1)^{st}$ transaction amount gets a value different from a BTCs, $(i.ii)$ $(n+1)^{st}$ transaction amount gets equal to a BTCs. In case $(i.i)$, the numbers of transactions having an amount equal to a BTCs are equal for D_1 and D_2, and $F(D_1)$ is equal to $F(D_2)$, therefore, differential privacy is provided. In case $(i.ii)$, $F(D_1)$ equals $F(D_2)+1$. Consider the case when $F(D_2)$ equals 0, i.e., no transaction amount is equal to a BTCs after the noise addition. In this case, $F(D_1)$ is equal to 1. The range of F is $[0, n+1]$ for D_1 and $[0, n]$ for D_2. For S is $[1, n]$, the violation of differential privacy can be shown as in Section 4.1.3. In the second case, when Laplace noise values are added to the amounts in these datasets, $(n+1)^{st}$ transaction of D_1 has no longer an amount equal to a. Remaining n transactions are the same for D_1 and D_2, and when the noise values are added to the

amounts, these n transactions again be the same. As a result, $F(D_1)$ equals $F(D_2)$, and the differential privacy formula given in 2.1 is true for all subsets and ε values. For case (i), ½ of the cases violates differential privacy, and for case (ii), there is no differential privacy violation. For this query, the weighted average of the differential privacy violation becomes $\frac{1}{2}\times\frac{1}{2}+\frac{1}{2}\times 0 = {}^{1}/_{4}$.

4.2.4 Effect of Noise Addition on Queries for Transactions with a Specific Amount Between Two Specific Addresses

Consider the function in Section 4.1.4, where one wishes to learn whether a transaction with an amount equal to a BTCs occurred between two specific Bitcoin addresses. Let A_1 and A_2 denote the addresses and F be a function that gives the number of transactions between A_1 and A_2 that have an amount equal to a BTCs. Let D_1 and D_2 be two neighbor datasets that consist of exactly the same n transactions and D_1 has an additional $(n+1)^{st}$ transaction. For this query function, four cases must be considered to cover all possible datasets; (i) the $(n+1)^{st}$ transaction is a transaction between $A_1$and A_2, the amount is not equal to a BTCs, (ii) the $(n+1)^{st}$ transaction is not a transaction between $A_1$and A_2, the amount is equal to a BTCs, (iii) the $(n+1)^{st}$ transaction is not a transaction between $A_1$and A_2, the amount is not equal to a BTCs, (iv) the $(n+1)^{st}$ transaction is a transaction between $A_1$and A_2, the amount is equal to a BTCs. These cases and the corresponding $(n+1)^{st}$ transaction states, relations between $F(D_1)$ and $F(D_2)$, differential privacy provision or violation statuses after the noise addition are given in Table 4.8.

The range of F is $[0, n+1]$ for D_1 and $[0, n]$ for D_2. When $F(D_1)$ equals $F(D_2)+1$, the violation of differential privacy can be shown by considering the case when $F(D_2)$ equals 0, and $F(D_1)$ is equal to 1 for S is $[1, n]$. When $F(D_1)$ equals $F(D_2)$, the differential privacy formula given in 2.1 is true for all subsets and ε values. For case (i), ½ of the cases violates differential privacy, and for cases $(ii-iv)$, there is no differential privacy violation. For this query, the weighted average of the differential privacy violation becomes $\frac{1}{4}\times\frac{1}{2}+\frac{1}{4}\times 0+\frac{1}{4}\times 0+\frac{1}{4}\times 0 = {}^{1}/_{8}$.

Table 4.8: Cases considered in differential privacy evaluation of the queries for transactions with a specific amount between two specific addresses with noise addition

Case	$(n+1)^{st}$ Transaction Before Noise	$(n+1)^{st}$ Transaction After Noise	$F(D_1)$ and $F(D_2)$ After Noise	Differential Privacy After Noise
i	$A_1 \rightarrow A_2$ Amount $\neq a$	$A_1 \rightarrow A_2$ Amount $\neq a$	$F(D_1) = F(D_2)$	✓
		$A_1 \rightarrow A_2$ Amount = a	$F(D_1) = F(D_2) + 1$	X
ii	$A_1 \nrightarrow A_2$ Amount = a	$A_1 \nrightarrow A_2$ Amount $\neq a$	$F(D_1) = F(D_2)$	✓
iii	$A_1 \nrightarrow A_2$ Amount $\neq a$	$A_1 \nrightarrow A_2$ Amount $\neq a$	$F(D_1) = F(D_2)$	✓
		$A_1 \nrightarrow A_2$ Amount = a	$F(D_1) = F(D_2)$	✓
iv	$A_1 \rightarrow A_2$ Amount = a	$A_1 \rightarrow A_2$ Amount $\neq a$	$F(D_1) = F(D_2)$	✓

4.3 Feasibility of the Utilization of User Graph Perturbation in Bitcoin

Another potential way of provisioning differential privacy in Bitcoin is the perturbation of the user graph. In the user graph, also named the user network, the flow of bitcoins between users over time is depicted as a directed graph. Nodes represent users, namely Bitcoin addresses, and directed edges represent the flow of bitcoins between users. An example of the user graph is given in Figure 4.9.

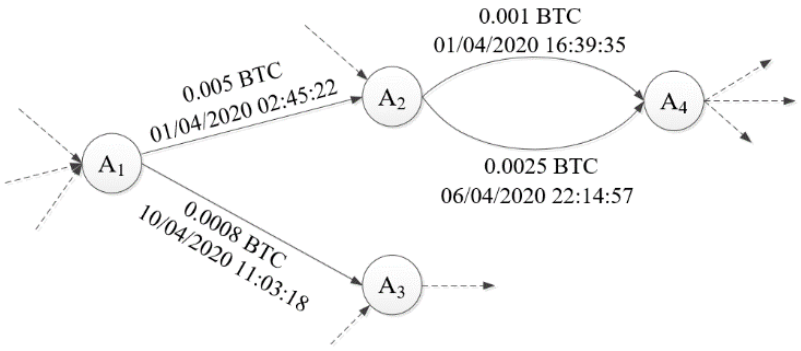


Figure 4.9: A sample Bitcoin user graph

Graph perturbation can be applied as adding dummy edges, i.e., dummy transactions, between users or deleting some existing edges, i.e., actual transactions. This change also requires a change of the Bitcoin transaction verification mechanism. Again, our focus in

this paper is on the examination of applying differential privacy mechanisms and the corresponding results; we leave the design of such a verification mechanism, and examination of the utility of perturbed transaction graph as future work. In the following subsections, we examine the effect of graph perturbation on differential privacy for the four query functions, which were examined in Section 4.1 and Section 4.2.

4.3.1 Effect of Graph Perturbation on Queries for Transactions Between Two Specific Addresses

Consider the query function that was given in Section 4.1.1, i.e., one wishes to learn whether a transaction occurred between two specific Bitcoin addresses, A_1 and A_2. Let D_1 and D_2 be two neighbor datasets that consist of exactly the same n transactions and D_1 has an additional $(n+1)^{st}$ transaction. Example graphs for D_1 and D_2 are given in Fig. 11.

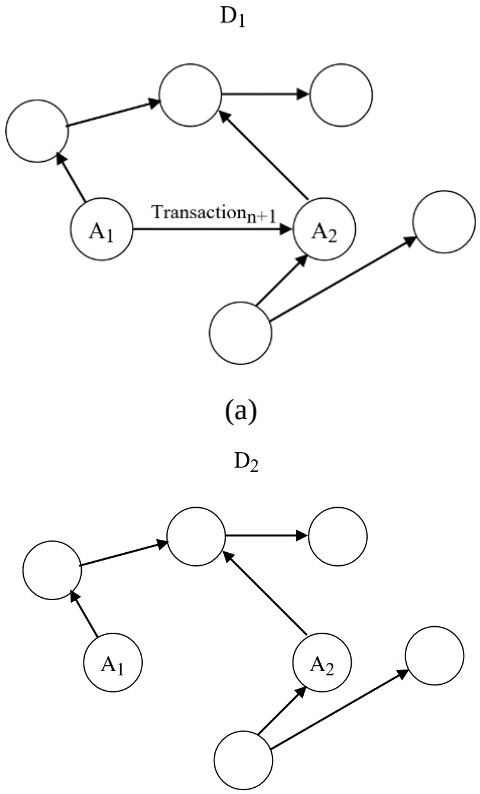


Figure 4.10: (a) D_1 consists of $n+1$ transactions that n of them are exactly the same with the n transactions of D_2 and an $(n+1)^{st}$ transaction which is between A_1 and A_2; (b) D_2 is a dataset that has exactly the same n transactions of D_1

Let F be a function that gives the average transaction amount between A_1 and A_2. For this query function, two cases must be considered to cover all possible datasets; (i) $(n+1)^{st}$ transaction is between A_1 and A_2, (ii) $(n+1)^{st}$ transaction is not between A_1 and A_2. In the first case, when graph perturbation is applied to these datasets, the following two cases can occur:

- In D_1, the graph perturbation deletes the $(n+1)^{st}$ transaction. Between A_1 and A_2, no or some dummy transactions may be added. In any case, $F(D_1)$ equals $F(D_2)$, and the differential privacy formula given in 2.1 is true for all subsets and ε values.
- In D_1, the graph perturbation does not delete the $(n+1)^{st}$ transaction. Between A_1 and A_2, no or some dummy transactions may be added. In any case, $F(D_1)$ equals $F(D_2)+1$. When $F(D_2)$ equals 0, for S is $[1, n]$, there is a violation of differential privacy.

In the second case, when graph perturbation is applied to D_1 and D_2, since $(n+1)^{st}$ transaction is not between A_1 and A_2, in the end, $F(D_1)$ equals $F(D_2)$. As a result, the differential privacy formula given in 2.1 is true for all subsets and ε values.

For case (i), ½ of the cases violates differential privacy, and for case (ii), there is no differential privacy violation. For this query, the weighted average of the differential privacy violation becomes $\frac{1}{2}\times\frac{1}{2}+\frac{1}{2}\times 0 = {}^1\!/_4$.

4.3.2 Effect of Graph Perturbation on Queries for Transactions for a Specific Amount

Consider the query function given in Section 4.1.3, i.e., one wishes to learn whether a transaction with an amount equal to a BTCs occurred. Function F gives the number of transactions with an amount a in the blockchain. Let D_1 and D_2 be two neighbor datasets as described earlier. For this query function, two cases that must be considered to cover all possible datasets are as follows; (i) $(n+1)^{st}$ transaction amount is a BTCs, (ii) $(n+1)^{st}$ transaction amount is not a BTCs. In the first case, when the graph perturbation is applied to these datasets, the following two cases can occur:

- In D_1, the graph perturbation deletes the $(n+1)^{st}$ transaction. No or some dummy transactions with an amount equal to a BTCs may be added. In any case, $F(D_1)$ equals $F(D_2)$, and the differential privacy formula given in 2.1 is true for all subsets and ε values.

- In D_1, the graph perturbation does not delete the $(n+1)^{st}$ transaction. No or some dummy transactions with an amount equal to a BTCs may be added. In any case, $F(D_1)$ equals $F(D_2)+1$. When $F(D_2)$ equals 0, for S is $[1, n]$, there is a violation of differential privacy.

In the second case, when graph perturbation is applied to D_1 and D_2, since $(n+1)^{st}$ transaction is not equal to a BTCs, in the end, $F(D_1)$ equals $F(D_2)$. As a result, the differential privacy formula given in 2.1 is true for all subsets and ε values.
For case (i), ½ of the cases violates differential privacy, and for case (ii), there is no differential privacy violation. For this query, the weighted average of the differential privacy violation becomes $\frac{1}{2}\times\frac{1}{2}+\frac{1}{2}\times 0 = {}^{1}/_{4}$, as well.

4.3.3 Effect of Graph Perturbation on Queries for Transactions with a Specific Amount Between Two Specific Addresses

Consider the query function given in Section 4.1.4, i.e., one wishes to learn whether a transaction with an amount equal to a BTCs occurred between two specific Bitcoin addresses. Let A_1 and A_2 denote the addresses and F be a function that gives the number of transactions between A_1 and A_2 that has an amount equal to a BTCs. Let D_1 and D_2 be two neighbor datasets as described earlier. For this query function, four cases that must be considered to cover all possible datasets are as follows; (i) the $(n+1)^{st}$ transaction is a transaction between $A_1$and A_2, the amount is not equal to a BTCs, (ii) the $(n+1)^{st}$ transaction is not a transaction between $A_1$and A_2, the amount is equal to a BTCs, (iii) the $(n+1)^{st}$ transaction is not a transaction between $A_1$and A_2, the amount is not equal to a BTCs, (iv) the$(n+1)^{st}$ transaction is a transaction between $A_1$and A_2, the amount is equal to a BTCs.

In the first three cases, since $(n+1)^{st}$ transaction is not a transaction between A_1 and A_2 with an amount equal to a BTCs, in any case, $F(D_1)$ equals $F(D_2)$ after the graph perturbation. As a result, the differential privacy formula given in 2.1 is true for all subsets and ε values.

In the fourth case, the following two cases can occur:

- In D_1, the graph perturbation deletes the $(n+1)^{st}$ transaction. No or some dummy transactions with an amount equal to a BTCs may be added. In any case, $F(D_1)$ equals $F(D_2)$, and the differential privacy formula given in 2.1 is true for all subsets and ε values.
- In D_1, the graph perturbation does not delete the $(n+1)^{st}$ transaction. No or some dummy transactions with an amount equal to a BTCs may be added. In any case, $F(D_1)$ equals $F(D_2)+1$. When $F(D_2)$ equals 0, for S is $[1, n]$, there is a violation of differential privacy.

For cases $(i-iii)$, there is no differential privacy violation. For case (iv), ½ of the cases violates differential privacy. As a result, the weighted average of the differential privacy violation for this query becomes $\frac{1}{4}\times 0+\frac{1}{4}\times 0+\frac{1}{4}\times 0+\frac{1}{4}\times\frac{1}{2}=1/8$.

4.3.4 Effect of Graph Perturbation on Queries for Transactions Above a Specific Amount

Consider the query function that was given in Section 4.1.2, i.e., one wishes to learn whether a transaction with an amount above a BTCs occurred. F is a function that gives the number of transactions greater than a BTCs in the blockchain. D_1 and D_2 are two neighbor datasets as described in the previous query function. Again, two cases must be considered to cover all possible datasets; (i) $(n+1)^{st}$ transaction is above a BTCs, (ii) $(n+1)^{st}$ transaction is not above a BTCs. In the first case, when graph perturbation is applied to these datasets, the following two cases can occur:

- In D_1, the graph perturbation deletes the $(n+1)^{st}$ transaction. No or some dummy transactions above a BTCs may be added. In any case, $F(D_1)$ equals

$F(D_2)$, and the differential privacy formula given in 2.1 is true for all subsets and ε values.

- In D_1, the graph perturbation does not delete the $(n+1)^{st}$ transaction. No or some dummy transactions above a BTCs may be added. In any case, $F(D_1)$ equals $F(D_2)+1$. When $F(D_2)$ equals 0, for S is $[1, n]$, there is a violation of differential privacy.

In the second case, when graph perturbation is applied to D_1 and D_2, since $(n+1)^{st}$ transaction is not above a BTCs, in the end, $F(D_1)$ equals $F(D_2)$. As a result, the differential privacy formula given in 2.1 is true for all subsets and ε values.

Again, for case (i), ½ of the cases violates differential privacy, and for case (ii), there is no differential privacy violation. For this query, the weighted average of the differential privacy violation becomes $\frac{1}{2}\times\frac{1}{2}+\frac{1}{2}\times 0 = {}^{1}/_{4}$.

4.4 An Empirical Study on Noise Addition to Transaction Amounts

We demonstrate a practical utilization of a noise addition to Bitcoin transaction amounts in an empirical way in this section. We add noise by applying the Laplace, the Gaussian, the Geometric, and the Uniform mechanisms for the noise generation at different ε values, and evaluate the results.

There are several differential privacy libraries to use. SmartNoise (https://smartnoise.org; https://github.com/opendp/smartnoise-core), which is a joint study of Microsoft and Harvard School of Engineering and Applied Sciences, Google's differential privacy library (https://github.com/google/differential-privacy), and Diffprivlib (https://diffprivlib.readthedocs.io; Holohan et al., 2019), the IBM Differential Privacy Library, are the prominent alternatives. The comparison of these libraries according to the variety of differential privacy mechanisms they provide is given in Table 4.9. We used SmartNoise v0.2.2 and this library offers the Laplace, the Gaussian, and the Geometric mechanisms. We used Google library v0.0.1 and it provides the Laplace and the Gaussian mechanisms. We used IBM library DiffPrivlib v0.4 and this library is the one affording the greatest number of mechanisms for numerical values. The library provides the

Laplace, the Gaussian, the Geometric, and the Uniform mechanisms for noise generation in order to achieve a differentially private model. Moreover, according to our evaluation, the documentation of DiffPrivlib is more comprehensible and the usage of the mechanisms is more straightforward, compared to the alternatives. As a result, we selected DiffPrivlib with Python support for our experiments.

Table 4.9: The comparison of the differential privacy libraries

Mechanism	SmartNoise v0.2.2	Google v0.0.1	IBM DiffPrivlib v0.4
Laplace	+	+	+
Gaussian	+	+	+
Geometric	+	−	+
Uniform	−	−	+

The referenced publication and the parameter details of the mechanisms provided by Diffprivlib are summarized in Table 4.10. Regarding the mechanism parameters, ε can have 1 as the maximum value for the Gaussian mechanism, whereas ε can have higher values than 1 for the Laplace, and the Geometric mechanisms. The Uniform mechanism only uses δ instead of ε, and δ can have a maximum of 0.5. The mechanisms also have a parameter for the sensitivity, which is not stated in the table. We use 1 for the sensitivity parameter for all runs since three out of four query functions that we analyzed in Sections 4.1 and 4.2 have sensitivity equal to 1. There are some points to be considered while adding noise to the Bitcoin transaction amounts. The minimum amount of bitcoin that can be sent in a transaction is 546 satoshis, which is equivalent to 0.00000546 BTC. Besides, we assume that the maximum amount of bitcoin that can be sent in a transaction at a certain time is equal to the total amount of bitcoins mined until that time. As of April 2021, we take this maximum value as 18,670,000 from blockchain.com website (https://www.blockchain.com/explorer/charts/total-bitcoins). Therefore, the minimum value that a noise added amount can get is 0.00000546 BTC, and the maximum value that a noise added amount can get is 18,670,000 BTC, and the noise values must be added accordingly. Diffprivlib offers folded versions of the Laplace and the Geometric mechanisms. In the folded versions, values outside a predefined range are folded back toward the domain around the closest point within the domain. Since the noisy values must be between 0.00000546 BTC and 18,670,000 BTC in our problem, rather than using the *Laplace* and the *Geometric* classes, we used the *LaplaceFolded* and *GeometricFolded*

classes. We set the lower and the upper bounds as 0.00000546 and 18,670,000 respectively in these methods. Although Laplace and LaplaceFolded can be used with real numbers, Geometric and GeometricFolded require an integer input. Therefore, while using GeometricFolded, if an amount is not an integer, we multiplied it with 10^8 to make it an integer value, then applied the *randomise* method to obtain the noisy value and then divided the output by 10^8. Since a folded version for the Gaussian mechanism is not provided in the library, the noise addition trial is done until the noisy value falls within the lower and the upper bounds. Another point to consider is that a noise-added value can have a decimal fraction of up to 8 digits since satoshi is the smallest unit of the currency, which is equal to one hundred millionth of a single bitcoin (0.00000001 BTC). Accordingly, outputs of the randomization methods are rounded to 8 decimal places. We utilized the Python NumPy libraries in our implementation.

Table 4.10: The details of the mechanisms provided by Diffprivlib

Mechanism	Reference in the documentation	Parameters	Input Type
Laplace	(Dwork et al., 2006)	ε: float. Must be in [0, ∞]. δ: float. Must be in [0, 1], default: 0.0.	integer
Gaussian	(Dwork & Roth, 2014)	ε: float. Must be in (0, 1]. δ: float. Must be in (0, 1].	integer
Geometric	(Ghosh et al., 2009)	ε: float. Must be in (0, ∞].	float
Uniform	(Geng et al., 2019)	δ: float. Must be in (0, 0.5].	integer

We used a published dataset including Bitcoin network transactional metadata (Shafiq, 2019). We carried out our experiments by adding noises to *in_btc* fields in this dataset, which are the input amounts of the transactions. We used randomly selected transaction data from 01.01.2014 and 02.01.2014.

In our experiments, first, we analyzed the effect of the dataset size on the behavior of the mechanisms. To this end, while applying the mechanisms, we changed the dataset size to 100, 1,000, and 10,000 respectively. For the evaluation, we used mean absolute error (MAE) values, calculated by summing the absolute differences between the noisy amount values and the actual values, and taking the mean. For the ε parameter of the Laplace, the Gaussian, and the Geometric mechanisms, we used 0.01, 0.05, 0.1, 0.5, and 1. For the δ parameter of the Uniform mechanism, we used 0.01, 0.05, 0.1, and 0.5 since this parameter can have a maximum of 0.5. Although ε can have a value greater than 1 in the

Laplace and the Geometric mechanisms, our tests showed that the amount of noise generated is insignificant when this value is greater than 1. As a result, we did not include the results for the greater ε values. We used 1 for δ in the Gaussian mechanism in all runs. The results are given in Figures 4.11-4.13. In the figures, there are no bars for the Uniform mechanism when ε is 1 since it cannot be greater than 0.5.

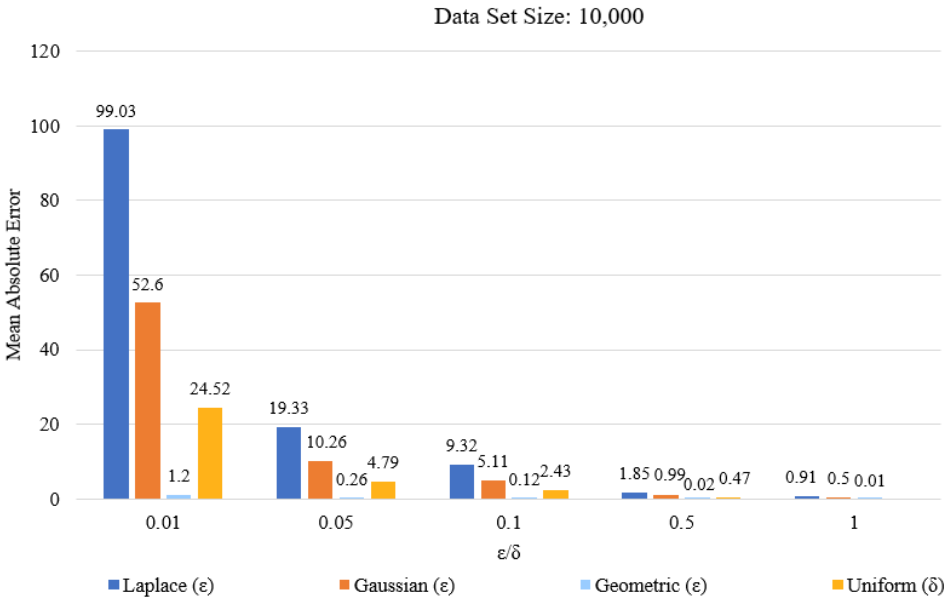


Figure 4.11: Mean absolute errors for varying ε, δ values when the dataset size is 10,000

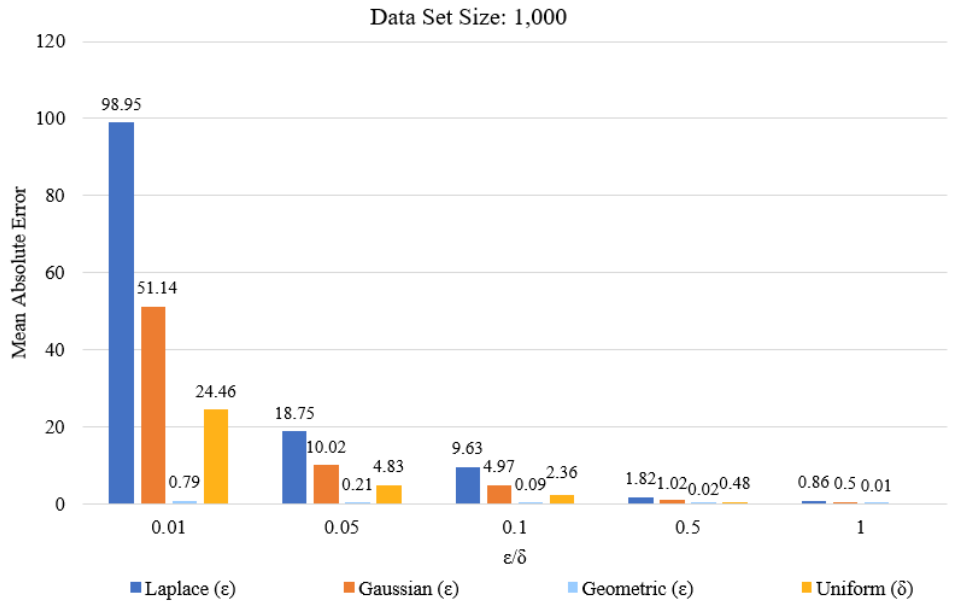


Figure 4.12: Mean absolute errors for varying ε, δ values when the dataset size is 1,000

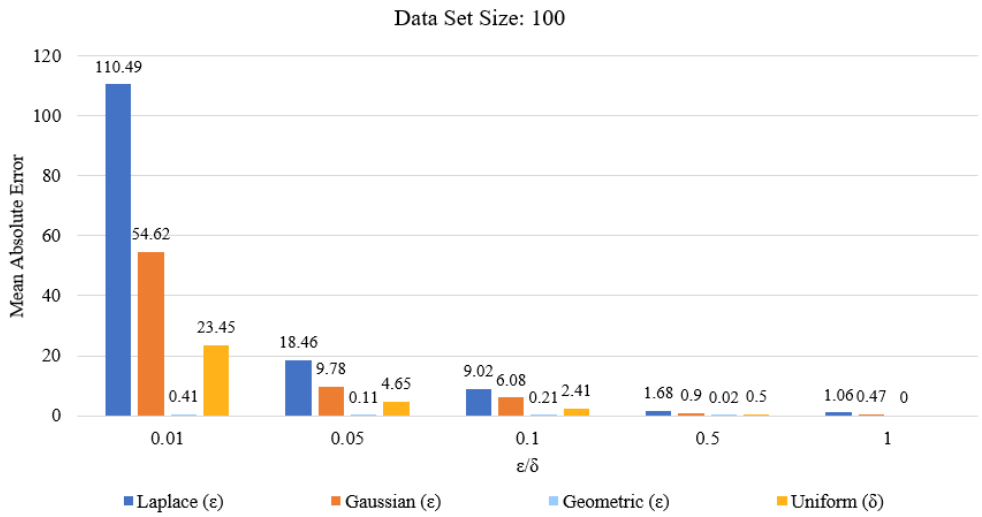


Figure 4.13: Mean absolute errors for varying ε, δ values when the dataset size is 100

Figures 4.11-4.13 show that changing the dataset size does not make a significant difference in the MAE values. Apart from the dataset size, the figures show that the MAEs decrease as the ε (or δ) value increases. This outcome is expected since privacy reduces as ε (or δ) increases, and the amount of noise reduces consequently. Moreover, changing the dataset size does not make a difference in the order of the mechanisms. The Laplace mechanism results in the highest MAEs for all dataset sizes and all ε values. The Gaussian is the second by adding approximately the half amount of noise compared to the Laplace. The Uniform is the third in the MAE ranking by adding approximately a quarter amount of noise compared to the Laplace mechanism. The Geometric mechanism results in the lowest MAEs, which are significantly lower compared to the other mechanisms. While comparing the mechanisms for the same ε value, it can be said that a higher MAE is better since a higher MAE means that the total amount of noise is higher, resulting in higher privacy protection, as in (Hassan et al., 2020b). Accordingly, the Laplace mechanism is the best for hiding transaction amounts by adding a larger amount of noise. The Gaussian comes next, and the Uniform follows the Gaussian. It is expected that the noisy and the actual amounts are close when the Geometric mechanism is used due to the low noise amounts.

We also visualize 100 actual transaction amounts belonging to 01.01.2014 from the dataset and the corresponding noisy values according to the mechanisms for ε equal to 0.01, 0.05, 0.1, 0.5, and 1 and δ equal to 0.01, 0.05, 0.1, and 0.5 in Figures 4.14-4.18. The average of the actual amounts is 1.409928611, the maximum is 14.96900006, and the

minimum is 0.001. From the figures, it is observed that the noisy values deviate a lot from the actual amounts in the Laplace, the Gaussian, and the Uniform mechanisms when ε or δ is smaller than 0.5. The fluctuation of the Laplace mechanism is significant when compared to the other mechanisms. The noisy values in the Geometric mechanism seem to be very close to the actual amounts for all ε values. In these figures, it can be seen that mostly positive amounts of noise are added, i.e., the actual amounts are lower than the noisy amounts mostly. This situation is due to that Bitcoin transaction amounts do not allow so much negative amount of noise since there is a minimum threshold of 0.00000546 BTC, which is the minimum transaction amount. Therefore, the mechanisms continue to generate noise until the noisy amount falls between the minimum and the maximum limits. Especially for the lower ε or δ values, i.e. greater noise amounts, the final noisy value tends to be a greater value than the actual value since the maximum limit, which is assumed as 18,670,000 in this study, is quite large.

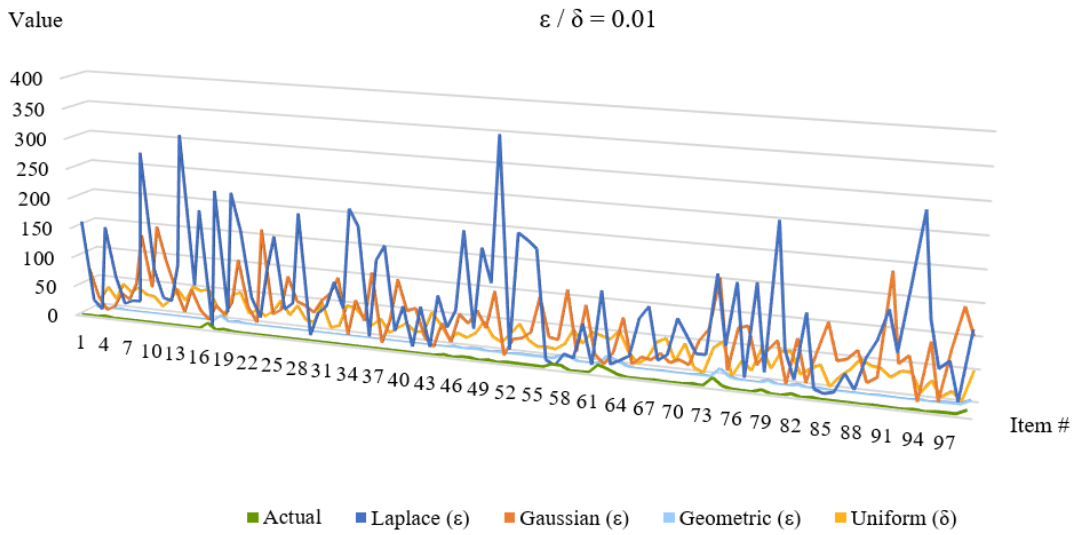


Figure 4.14: The actual transaction amounts along with the noisy amounts when ε or δ is 0.01

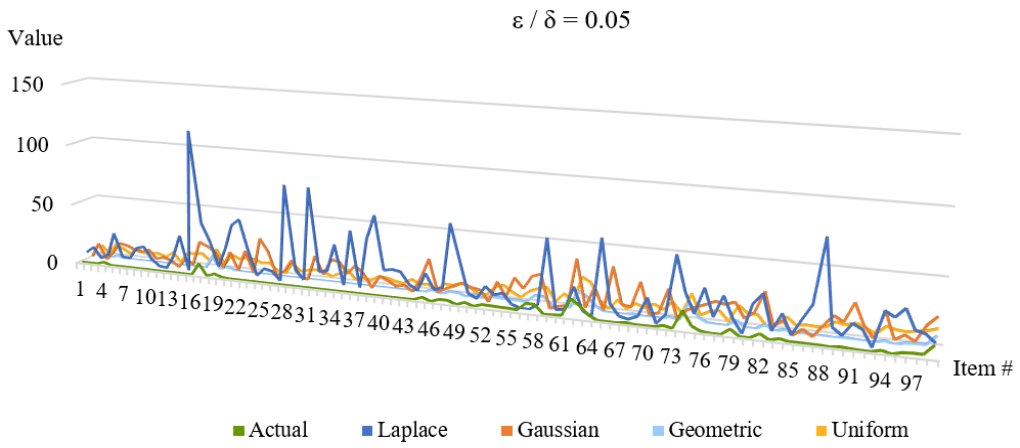


Figure 4.15: The actual transaction along with the noisy amounts when ε or δ is 0.05

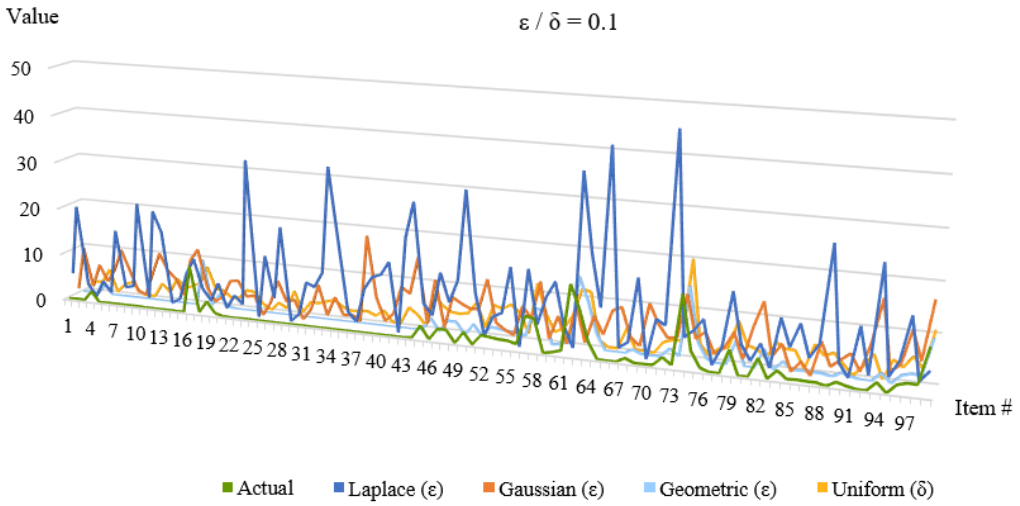


Figure 4.16: The actual transaction along with the noisy amounts when ε or δ is 0.1

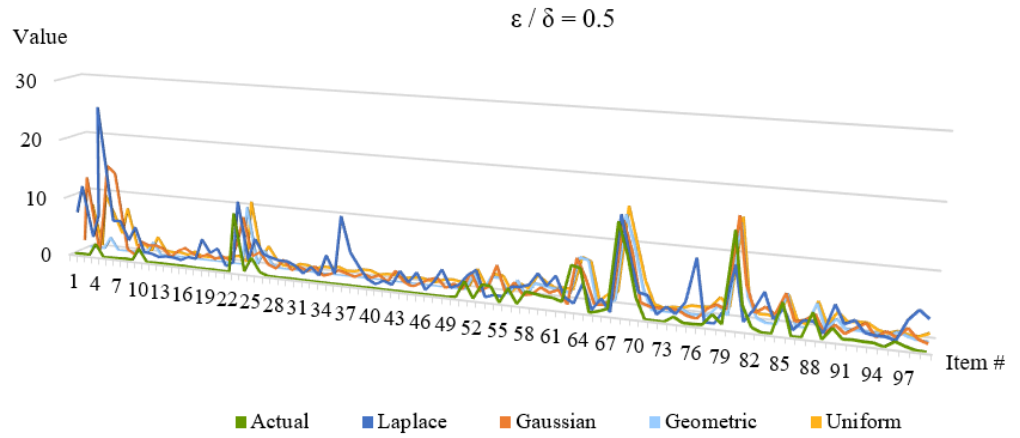


Figure 4.17: The actual transaction along with the noisy amounts when ε or δ is 0.5

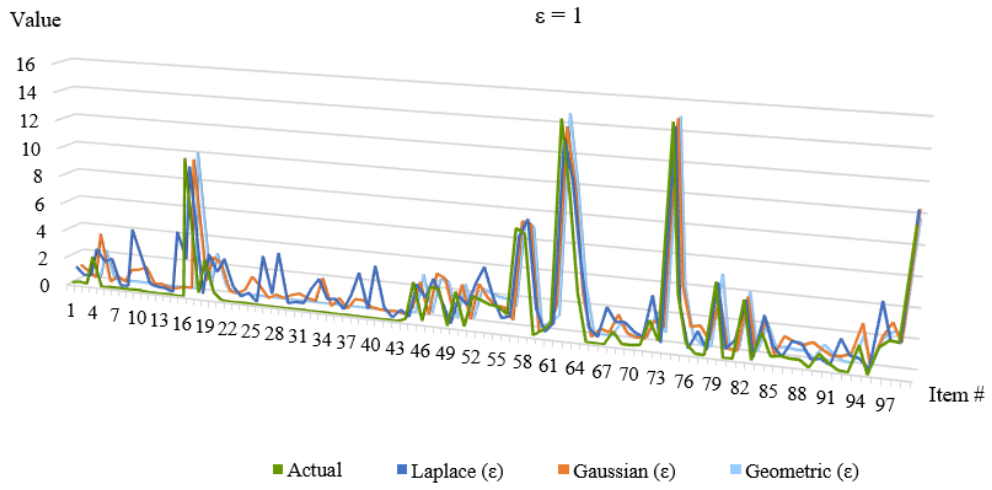


Figure 4.18: The actual transaction along with the noisy amounts when ε or δ is 1

One of our aims while considering differential privacy for improving anonymity and privacy in Bitcoin has been preventing privacy breaches via direct queries. In the previously mentioned scenario with 0.000381 BTC valued shopping from a well-known e-commerce site, the transactions with the noisy amounts near 0.000381 in the blockchain may be considered as the candidates while attempting to detect the corresponding transaction. Similarly, an observer may think of using the rank information of the transaction with 0.000381 amount value when all transactions in the dataset are sorted by amounts. The transaction with the same rank or the transactions having ranks close in the noise added dataset may be considered as the candidate transactions corresponding to the transaction sought. In order to examine the differential privacy mechanisms from this aspect, we examined the change in the ranks of specific transactions before and after adding noise. The amount of change shows the performance of the mechanism at hiding the actual rank, and a higher change in the rank makes it difficult for an observer to detect a transaction related to a specific transaction amount.

In our dataset with 100 amount values, we first checked the ranks of the noisy values corresponding to 14.96900006, which is the maximum of the actual amounts, for varying mechanisms and ε (or δ) values. The average ranks and standard deviations for 25 runs of noise addition are given in Table 4.11. When the average ranks rounded to the closest integer, we observed that the mechanisms are unable to hide the rank when ε is 1 for all mechanisms using ε. The rank of the noisy value does not change for all ε values in the Geometric mechanism. The Laplace mechanism hides the actual rank in $^{4}/_{5}$ of the cases,

the Gaussian mechanism hides the actual rank in $^3/_5$ of the cases, and the Uniform mechanism hides the actual rank in $^3/_4$ of the cases. The average ranks and the standard deviation values decrease as ε (or δ) increases.

Table 4.11: The average ranks and the standard deviations of the noisy values corresponding to 14.96900006 which is the 1st in the actual amounts in descending order

	ε or δ									
	0.01		0.05		0.1		0.5		1	
Mechanism	Avg Rank	Std Dev	Avg Rank	Std Dev	Avg Rank	Std Dev	Avg Rank	Std Dev	Avg Rank	Std Dev
Laplace (ε)	46.72	30.7	43.44	30.96	28.08	23.4	2.4	1.9	1.36	0.48
Gaussian (ε)	54.72	30.35	26.36	25.54	7.4	12	1.48	0.5	1.36	0.48
Geometric (ε)	1	0	1	0	1	0	1	0	1	0
Uniform (δ)	36.36	26.6	4.88	5.54	1.8	1.02	1.48	0.5	N/A	N/A

Then, we checked the ranks of the noisy values corresponding to 0.001, which is the minimum of the actual amounts, for varying mechanisms and ε values. The average ranks and standard deviations for 25 runs of noise addition are given in Table 4.12. Unlike the previous example, the Laplace and the Gaussian mechanisms hide the actual rank even when ε is 1. Again, the ranks of the noisy values do not change for all ε values in the Geometric mechanism. It can be seen that the Laplace and the Gaussian mechanisms hide the actual rank in all five ε values, and the Uniform mechanism hides the actual rank in all four δ values. No correlation can be observed in the average ranks or the standard deviations; the confusion arising from the noise addition is empirically shown.

Table 4.12: The average ranks and the standard deviations of the noisy values corresponding to 0.001, which is the 1st in the actual amounts in ascending order

	ε or δ									
	0.01		0.05		0.1		0.5		1	
Mechanism	Avg Rank	Std Dev	Avg Rank	Std Dev	Avg Rank	Std Dev	Avg Rank	Std Dev	Avg Rank	Std Dev
Laplace (ε)	41.76	26.27	44.44	26.89	55.8	30.23	50.28	24.72	40.44	21.24
Gaussian (ε)	50.64	30	46.52	29.24	39.28	27.02	37.96	20.57	30.24	19.49
Geometric (ε)	1	0	1	0	1	0	1	0	1	0
Uniform (δ)	47.88	29.7	53.23	26.78	37.4	28.76	37.88	27.06	N/A	N/A

Finally, we checked the ranks of the noisy values corresponding to the randomly selected 0.41510257 value, which is the 59th in the actual amounts in ascending order, for varying

mechanisms and ε (or δ) values. The average ranks and standard deviations for 25 runs of noise addition are given in Table 4.13. The rank of the noisy value stays the same for all ε values in the Geometric mechanism. The Laplace and the Gaussian mechanisms are successful at hiding the actual rank in all ε values, and the Uniform mechanism successfully hides the actual rank in all δ values. As in the previous example, no correlation can be observed in the average ranks or the standard deviations, and the confusion arising from the noise addition is empirically shown.

Table 4.13: The average ranks and the standard deviations of the noisy values corresponding to 0.41510257, which is the 59th in the actual amounts in ascending order

	ε or δ									
	0.01		0.05		0.1		0.5		1	
Mechanism	Avg Rank	Std Dev	Avg Rank	Std Dev	Avg Rank	Std Dev	Avg Rank	Std Dev	Avg Rank	Std Dev
Laplace (ε)	56.24	23.51	59.92	25.59	47.72	33.33	44.32	27.98	31.88	16.18
Gaussian (ε)	50.24	26.56	40.04	27.81	44.92	27.98	35.84	19.4	34.88	21.8
Geometric (ε)	59	0	59	0	59	0	59	0	59	0
Uniform (δ)	51.84	25.32	50.56	27.87	48.16	20.58	48	22.46	N/A	N/A

In order to generalize this approach to the whole dataset, we define a new metric called *mean ranking offset.* The mean ranking offset (MRO) over a dataset is calculated by taking the average of the absolute differences between the ranks of the actual values in the dataset in ascending order and the ranks of the noisy values in ascending order. As the MRO increases, the distances between the ranks of the noisy values and the actual values increase. Therefore, MRO is an indicator of how successful a mechanism is at hiding the actual ranks. We calculated the MRO values over our dataset with 100 transaction amounts for all mechanisms and ε, δ values that we evaluated in the previous analyses. The results are given in Table 4.14 and visualized in Figure 4.19. The largest MRO values are provided by the Laplace mechanism, for all ε (or δ) values considered. It is observed that the mean rank offset values for the Geometric mechanism are very close to 0 and the ineffectiveness of the mechanism compared to the other mechanisms can be clearly seen. For ε, δ = 0.01, the Uniform mechanism follows the Laplace, and the Gaussian mechanism comes after the Uniform. For ε, δ = 0.05, 0.1, and 0.5, the Uniform and the Gaussian change their order, the Gaussian follows the Laplace and the Uniform comes after the Gaussian. δ cannot be 1, therefore MRO is not calculated for the Uniform mechanism in this value. It is observed that as ε or δ increases, MRO values tend to decrease.

Table 4.14: Mean ranking offsets for varying mechanisms and ε/δ values

	ε or δ				
Mechanism	0.01	0.05	0.1	0.5	1
Laplace (ε)	35.29	35.07	31.8	27.6	22.06
Gaussian (ε)	31.43	27.76	29.11	23.78	14.38
Geometric (ε)	0.1	0.1	0.06	0.02	0.1
Uniform (δ)	33.07	25.78	25.01	17.84	N/A

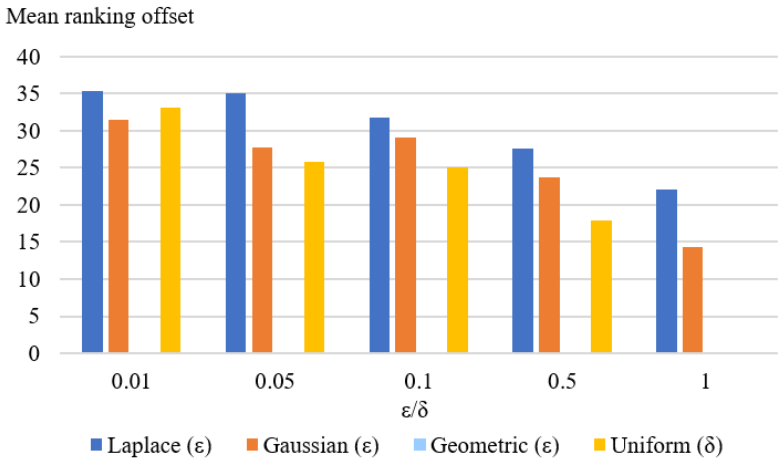


Figure 4.19: Mean ranking offsets for varying mechanisms and ε/δ values

4.5 Summary and Discussion

In this section, we summarize our research and observations for investigation and application of differential privacy in Bitcoin. In this study, firstly, the current implementation of Bitcoin is examined for four query functions in terms of differential privacy using the differential privacy formulation. Then, the feasibility of utilizing the noise addition and the graph perturbation mechanisms in Bitcoin is examined for these functions, as well. All possible cases for neighbor datasets are evaluated and the violations are detected. The fractions of the cases violating differential privacy are given in Table 4.15. The discussed functions query the average transaction amount between two specific addresses, the number of transactions having an amount above a BTCs, and the number of transactions having an amount equal to a BTCs, respectively. The selection of these functions was done by considering what an observer would like to learn and get insight from the public blockchain. Interactions between users and the amount values are some meaningful information to use with off-network information.

Table 4.15: The fraction of the cases violating differential privacy

Query	Function	Current implementation	Noise addition	Graph perturbation
1	Average transaction amount between A_1 and A_2	$1/2$	$1/2$	$1/4$
2	Number of transactions having an amount above a BTCs	$1/2$	$2/4$	$1/4$
3	Number of transactions having an amount equal to a BTCs	$1/2$	$1/4$	$1/4$
4	Number of transactions between A_1 and A_2 that have an amount equal to a BTCs	$1/4$	$1/8$	$1/8$

The current implementation of Bitcoin violates differential privacy in $1/2$ of the cases considered for the first three queries and $1/4$ of the cases considered for the fourth query. The application of noise addition does not change the fraction of the cases violating differential privacy for the first and the second functions, which query the average transaction amount between two specific addresses, and the number of transactions having an amount above a BTCs, respectively. However, the noise addition decreases the fraction of the cases violating differential privacy to $1/4$ for the third function, which queries the number of transactions having an amount equal to a BTCs. The noise addition decreases the fraction of the cases violating differential privacy to $1/8$ for the fourth function, which queries the number of transactions between two specific addresses with an amount equal to a BTCs.

The graph perturbation decreases the fraction of the cases violating differential privacy to $1/4$ for the first three functions. The fraction of the cases violating differential privacy is decreased to $1/8$ for the fourth function, similar to the noise addition. It can be concluded that both mechanisms can be used to improve anonymity and privacy, whereas the graph perturbation seems to be a better option for the first and the second functions. In these experiments, we covered all possible cases regardless of the amount and exact method of noise addition and perturbation. However, the amount of noise can be calculated using

$S(f)$, the sensitivity of a function. For the commonly used Laplace noise mechanism, adding noise with scale $S(f)/\varepsilon$ preserves ε-differential privacy.

Moreover, we demonstrated the utilization of the noise addition to transaction amounts by using the IBM differential privacy library. In our experiments, we examined the Laplace, the Gaussian, the Geometric, and the Uniform mechanisms for generating noise to add to the transaction amount values in a dataset for varying ε and δ values (ε = 0.01, 0.05, 0.1, 0.5, 1, and δ = 0.01, 0.05, 0.1, 0.5). The evaluations are done using MAE values. The results show that the MAEs decrease as ε (or δ) increases, as expected. We observed that the effect of changing the dataset size, to 100, 1000, and 10,000, does not make a significant difference in the MAE values. The dataset size change also does not make a difference in the order of the mechanisms. We hypothesize that the higher MAE is better since a higher MAE results in higher privacy protection. The Laplace mechanism results in the highest MAEs for all dataset sizes and all ε values. The Gaussian follows the Laplace, and the Uniform results in the third-highest MAEs. The Geometric mechanism is not found effective due to very low MAEs. The behaviors of the mechanisms, in terms of variation, are also noticed when the noisy values generated by the mechanisms for varying ε and δ values are visualized along with the actual amounts for 100 transactions. We also carried out experiments to analyze the effect of the noise addition on detecting a transaction with a specific amount. We introduced the *mean ranking offset* (MRO) metric, which gives the average rank change over a dataset after the noise addition when the transactions are sorted by amounts. In our evaluation for a dataset with 100 transactions, the Laplace mechanism provided the largest MRO values for all ε or δ values considered. The Gaussian showed a better performance compared to the Uniform in most of the cases and followed the Laplace. The Geometric is ineffective according to the MRO metric, as well. It is observed that the MRO values tend to decrease as ε or δ increases. Moreover, for the maximum and the minimum values in the dataset, we evaluated the mechanisms according to the fraction of the ε or δ values hiding the actual rank. The results are presented in Table 4.16. It can be seen that the rank of the actual minimum value is successfully hidden for all mechanisms except the Geometric. For hiding the rank of the actual maximum value, there is no mechanism that hides the actual rank for all ε or δ values. However, Laplace performs the best. The Gaussian follows the Laplace, and the Uniform comes after the Gaussian. The Geometric is unsuccessful at hiding both ranks for all mechanisms and ε, δ values.

Table 4.16: The fraction of the ε or δ values hiding the actual rank of the maximum and the minimum values in the dataset

Mechanism	Maximum	Minimum
Laplace	$4/5$	$5/5$
Gaussian	$3/5$	$5/5$
Geometric	$0/5$	$0/5$
Uniform	$3/4$	$4/4$

As the overall result of our experiments, within the mechanisms and the parameters we examined, the Laplace mechanism can be opted for successfully hiding the transaction amounts and ranks with ε equal or less than 0.5. However, in the previously mentioned related study (Hassan et al., 2020b), the most suitable values for ε and δ are determined as 0.01 for generating an adequate amount of noise. This may be due to the range of the values. The values in the mentioned study range between 200 and 1900, whereas the values used in this study are between 0.001 and 14.96900006 which exemplify the real Bitcoin transaction amounts. Another difference is that the Geometric mechanism is found to be successful for adequate noise generation in the mentioned study, whereas our experiments show the opposite by finding this mechanism ineffective.

While attaching the perturbation mechanism to the blockchain, it should be considered that the perturbation should not require a central party since the blockchain is managed collectively by the peers. A reasonable way of perturbation may be triggering and executing the perturbation algorithm automatically while publishing transactions, resulting in perturbed transaction data being added to the blockchain via dedicated and distributed servers as in (Kumar, 2020).

Another important point to consider is that the focus of this study was on the examination of applying differential privacy mechanisms and results in terms of satisfying differential privacy. In order to use these differential privacy mechanisms, the verification mechanism must be modified accordingly, and perturbed amounts or transaction graph must be examined in terms of utility. There may be concerns on the effect of the perturbation on the usability of data since hash values used in verification would change, however, these concerns can be addressed with the methods that come from the notion of modifiable

blockchains (Politou et al., 2019; Lee at al, 2019) emerged from the erasing requirements imposed by the GDPR's “right to be forgotten” provision.

Further research topics include the modification of the verification mechanism accordingly, and examining the effect of the perturbation on the degradation of utility. Moreover, applying these differential privacy mechanisms to other blockchain-based cryptocurrencies may be investigated, as well.

5. BLOCKCHAIN-BASED DIFFERENTIALLY-PRIVATE FEDERATED SMART UTILITY METERING

In this section, utilization of differential privacy in smart utility metering is investigated. In addition to differential privacy, we leverage blockchain for achieving federation of smart homes and different utility providers, i.e., electricity, water and gas. As a result, we propose a blockchain-based differentially-private federated smart utility metering framework. We detail key requirements and framework design in the first two subsections. Then, we present information leakage and differential privacy analysis in the next subsection. Finally, we state future research ideas in the last subsection.

5.1 Key Requirements

Key requirements of the proposed framework are determined as follows.

1. Multi-utility: The framework is federal. Multiple utilities take part in a federated utility infrastructure.
2. Smart metering: Smart metering, with authorized utility services and non-repudiated measurements, is applied.
3. Differentially-private: Noise values are added to the smart metering measurement values to achieve differential privacy.
4. Blockchain-based: Noisy consumption measurement values and aggregate noise values are kept in blockchain ledgers.
5. Financial settlement: It is guaranteed that the utility providers bill subscribers as much as they consumed.

5.2 Framework Design

In the proposed blockchain-based differentially-private federated smart utility metering framework, smart homes are connected via a decentralized blockchain network. All these smart meters are also connected to the utility service providers; i.e., smart meters for electricity are connected to the electricity utility provider, smart meters for water are connected to the water utility provider, and smart meters for gas are connected to the gas utility provider.

Smart meters regularly transmit their consumption measurement values to the utility service providers with a predefined measurement period. Utility service providers collect these values from all smart homes and put these values with some other details in a list awaiting validation. All smart homes can reach this list and the smart home selected as the validator put these values in a block. After the verification process, the block is added to the blockchain ledger and disseminated. Instead of three ledgers for three utilities, we use a single ledger as in (Williams, 2018). Utility type can be added as a field in the ledger records.

We hide the actual consumption measurement values by adding differentially private noise before transmitting them to the utility service providers. Each house has a tamper-proof data storage to keep the noise values. Our aim is to provide that the total amount that must be paid to all utilities without adding any noise becomes equal to the total amount that must be paid when noise is added. One method of providing this is to add noise to these three utilities in a related way. Aggregate noise values are transmitted to the utility providers in a predefined billing period. Aggregate noise values are stored in a separate reconciliation blockchain since they are used for reconciliation among the utility service providers. At the end of each billing period, there is a reconciliation process among the utility providers. This infrastructure is visualized in Figure 5.1 for three smart homes.

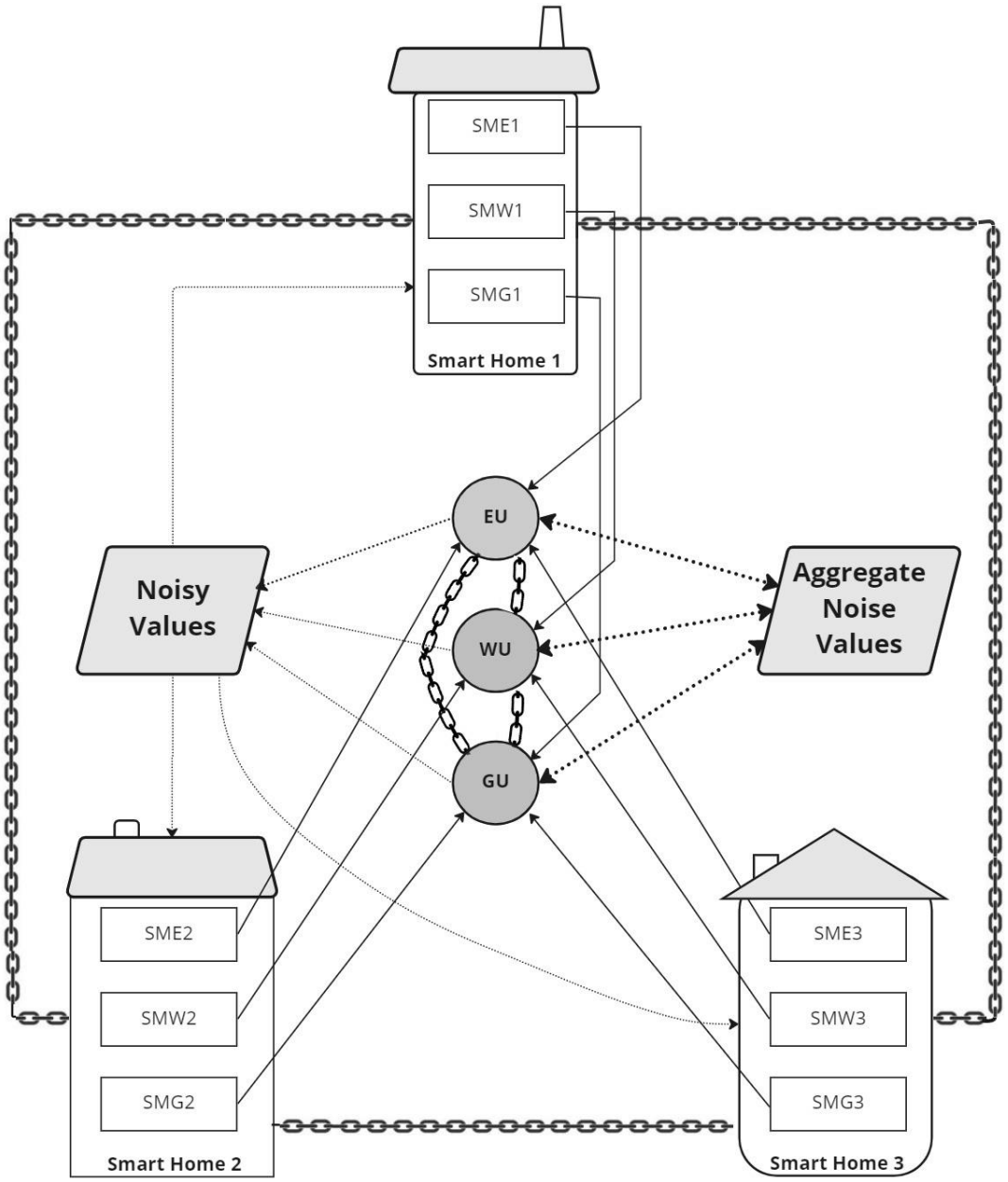


Figure 5.1: Proposed framework design

5.2.1 Entities

The entities in the framework are utility service providers, clients (smart homes), and an authority. Their roles are given in the following.

Utility service provider: This entity is an organization supplying utility and verified by the authority. It is also a blockchain node. The framework proposed in this study has three utility service providers; electricity, water, and gas.

- It receives noisy consumption measurement values at the end of each measurement period. It puts these values to the noisy values pool.
- It receives aggregate noise values from smart homes at the end of each billing period. It puts these aggregate noise values to the aggregate noise pool.
- It takes part in the reconciliation process between the utility service providers at the end of each billing period.
- It participates in block construction process for aggregate noise blockchain. If it is selected as the block creator, it creates a new block from the aggregate noise pool and adds to the blockchain. The created block is disseminated to all utility service providers.

Client (Smart home): This entity is the subscriber to the utility services, verified by the authority. It is also a blockchain node.

- It transmits the noise added consumption values to the service providers with a predefined measurement period. nce_{ij}, ncw_{ij}, ncg_{ij} are the noisy consumption measurement values, for electricity, water, and gas respectively, reported for the jth measurement period of a billing period for the ith smart home. ce_{ij}, cw_{ij}, cg_{ij} are the actual consumption measurement values and ne_{ij}, nw_{ij}, ng_{ij} are the noise values that are added for electricity, water, and gas respectively, for the jth measurement period of a billing period for the ith smart home. Then, nce_{ij}, ncw_{ij}, ncg_{ij} are calculated using Equations 5.1, 5.2, and 5.3.

$$nce_{ij} = ce_{ij} + ne_{ij} \quad (5.1)$$

$$ncw_{ij} = cw_{ij} + nw_{ij} \quad (5.2)$$

$$ncg_{ij} = cg_{ij} + ng_{ij} \quad (5.3)$$

- It has a tamper-proof data storage that stores aggregate noise values for that billing period. It adds the noise values for that measurement period to the current aggregate noise values. If $ane_{i(j-1)}$, $anw_{i(j-1)}$, and $ang_{i(j-1)}$ are the latest aggregate noise values, i.e., for the $(j-1)$th measurement period of a billing period for the ith smart home, then, the aggregate noise values at the end of the jth measurement period are calculated using Equations 5.4, 5.5, and 5.6.

$$ane_{ij} = ane_{i(j-1)} + ne_{ij} \quad (5.4)$$

$$anw_{ij} = anw_{i(j-1)} + nw_{ij} \quad (5.5)$$

$$ang_{ij} = ang_{i(j-1)} + ng_{ij} \quad (5.6)$$

- It participates in block construction process for noisy consumption blockchain. If it is selected as the block creator, it creates a new block from the noisy consumption pool and adds to the blockchain. The created block is disseminated to all smart homes.
- It transmits aggregate noise values to utility service providers at the end of each billing period.

Authority: This entity authorizes utility service providers and registers new smart homes after ensuring their legitimacy. It allows joining to both blockchain ledgers. It can reach to both blockchain ledgers and check the integrity of the framework.

5.2.2 Data Structures

The framework has two blockchain ledgers. One of them is between the smart homes, and the other one is between the utility service providers. These blockchain networks are private, smart homes and utilities can be added by an authority after they prove their legitimacy. In these networks, every blockchain node can participate in the consensus.

Instead of PoW, which depends on computational power and consumes resources, Proof-of-Stake (PoS) (Nguyen et al., 2019) consensus mechanism can be used.

In summary, the proposed framework has the following data structures:

- A blockchain ledger keeping noise added consumption measurement values of the smart homes:
    - It has a corresponding pool that keeps noisy consumption measurement values that await validation.
    - Each record consists of smart home identifier, utility type, consumption value, billing period, and measurement period.
- A blockchain ledger used for the reconciliation between utility service providers keeping aggregate noise values:
    - It has a corresponding pool that keeps aggregate noise values that await validation.
    - Each record consists of smart home identifier, utility type, aggregate noise value, and billing period.
- Tamper-proof data storages at smart homes that keep noise values.

5.2.3 Noise Addition

For hiding actual consumption measurement values, we apply a similar approach to non-random data obfuscation (Guan et al., 2018). In a measurement period, for a smart home, if the electricity consumption value is ce, the water consumption value is cw, and the gas consumption value is cg, and the electricity rate is r_e, the water rate is r_w, and the gas rate is r_g, then the total amount that will be paid to the utilities becomes $r_e \times ce + r_w \times cw + r_g \times cg$. If the noise values added for that measurement period are ne, nw, and ng, then the noisy consumption values that are transmitted to the utilities become $ce + ne$, $cw + nw$, and $cg + ng$. In this case, total amount that will be paid becomes $r_e \times (ce + ne) + r_e \times (cw + nw) + r_e \times (cg + ng)$. We provide that the total amount that must be paid to all utilities without adding any noise becomes equal to the total amount that must be paid when noise is added, by adding noise to two of the utility measurements randomly, and

adding noise to the remaining utility measurement in a way that will compensate the other two noise values. Therefore, Equation 5.7 must be hold.

$$r_e \times ce + r_w \times cw + r_g \times cg = r_e \times (ce + ne) + r_e \times (cw + nw) + r_e \times (cg + ng) \quad (5.7)$$

Equation 5.7 can be simplified to Equation 5.8.

$$r_e \times ne + r_w \times nw + r_g \times ng = 0 \quad (5.8)$$

For example, if the utilities that will receive noise values randomly are selected as the electricity and the water, then the noise that will be added to the gas consumption value, ng, becomes equal to $-\frac{(ne \times r_e + nw \times r_w)}{r_g}$.

Noise values can be either positive or negative. Besides, noisy consumption values can be negative.

5.2.4 Reconciliation and Billing

At the end of each billing period, two reconciliation processes must be executed between clients (smart homes) and utility service providers, and between utility service providers. For reconciliation and billing between clients and utility service providers, Algorithm 1 is run at the end of each billing period. SH is the list of smart homes. E denotes the electricity utility provider, W denotes the water utility provider, and G denotes the gas utility provider. Each smart home transmits aggregate noise values which are kept in the tamper-proof data storage to the corresponding utility service providers. Actual aggregate consumption values for each smart home is calculated by subtracting aggregate noise values from aggregate consumption values kept in utility service providers, which are noise added values. Then, bill amounts for each smart home are calculated by multiplying actual aggregate amounts by the rates of the utility services. be, bw, bg keeps bill amounts of smart homes for electricity, water, and gas, respectively.

Algorithm 1 Algorithm for Reconciliation Between Smart Homes and Utility Service Providers and Billing

Input: $\boldsymbol{SH}$, $\boldsymbol{ane}$, $\boldsymbol{anw}$, $\boldsymbol{ang}$, $\boldsymbol{ace}$, $\boldsymbol{acw}$, $\boldsymbol{acg}$, $\boldsymbol{r_e}$, $\boldsymbol{r_w}$, $\boldsymbol{r_g}$
Output: $\boldsymbol{be}$, $\boldsymbol{bw}$, $\boldsymbol{bg}$

for ($each\ s\ in\ SH$) **do**
    transmit($ane[s]$, E)
    transmit($anw[s]$, W)
    transmit($ang[s]$, S)
    ace[s] ← $ace[s] - ane[s]$
    acw[s] ← $acw[s] - anw$[s]
    acg[s] ← $acg[s] - ang[s]$
    be[s] ← $ace \times r_e$
    bw[s] ← $acw \times r_w$
    bg[s] ← $acg \times r_g$
end for
return (be, bw, bg);

For the reconciliation between utility service providers, Algorithm 2 is run at the end of each billing period.

Algorithm 2 Algorithm for Reconciliation Between Utility Service Providers

Input: $\boldsymbol{U}$

for ($each\ u\ in\ U$) **do**
    if $u.value < 0$
        N.add(u)
    else if $u.value > 0$
        P.add(u)
    end if else
end for
for ($each\ p\ in\ P$) **do**
    for ($each\ n\ in\ N$) **do**
        if $p.value \leq |n.value|$
            $p.value \leftarrow 0$
            $n.value \leftarrow -(|n.value| - p.value)$
            transfer(p, n)
            break;
        else if
            $n.value \leftarrow 0$
            $p.value \leftarrow p.value - |n.value|$
            transfer(p, n)
        end if else
    end for
end for

transfer is the function that makes payment. U denotes the utility list. Each utility has *value* which is the sum of aggregate noises for all smart homes multiplied by the utility rate. At the end of the first *for each* loop, *P* consists of utilities that have positive value for the sum of aggregate noises multiplied by the utility rate. Similarly, at the end of the first *for each* loop, *N* consists of utilities that have negative value for the sum of aggregate noises multiplied by the utility rate. In the second *for each* loop, reconciliation is done by transferring corresponding amounts from utilities that in *P* to utilities that are in *N*. At the end of the algorithm, each *value* in lists *P* and *N* becomes equal to 0.

For instance, if the sum of aggregate noise values for electricity multiplied by the electricity utility rate is 200, and the sum of aggregate noise values for water multiplied by the water utility rate is 100, then the sum of aggregate noise values for gas multiplied by the gas utility rate becomes -300. In this case, the amounts in the electricity and the water utilities must be transferred to the gas utility for the reconciliation, and Algorithm 2 provides this.

5.3 Information Leakage and Differential Privacy Analysis

In this subsection, we analyze the framework in terms of information leakage and differential privacy. Without adding noise and the blockchain ledgers, who consumes which utility and how much information are known by the utility service providers. Without adding noise, but with utilizing the blockchain ledger, this information is public to all clients (smart homes). With this data leak, presence or absence of the household, number of people at the house, daily routines of the household can be inferred. Simply, if the measurement values are zero for a period, i.e., there is not any consumption at a smart home, then there is no human being at the house in that period. The framework prevents this data leak by adding noise to the actual consumption measurements. Consider that there is no one at a house during a measurement period, and the actual smart meter measurements are equal to zero. Since noise values are added to the measurement values before they are transmitted to the utility service providers, no one can infer that there is no one at the house.

In order to examine information leakage, transmitted and immobile data of the framework can be considered. The framework has the following as the transmitted data:

- From smart homes to the utility service providers:
    - Noise added measurement values at the end of each measurement period
    - Aggregate noise values at the end of each billing period
- Between smart homes:
    - Validated block for the smart home consumptions ledger
- Between utility service providers:
    - Validated block for the reconciliation between utility service providers ledger

The framework has the following as the immobile data:

- Aggregate noise values kept in tamper-proof data storage at each smart home. At the end of the billing period, these values are reset after the reconciliation process.
- Noisy consumption measurement values that await validation in the smart home consumption pool
- Aggregate noise values that await validation in the pool
- Noise added measurement values in the smart home consumptions ledger
- Aggregate noise values kept in the ledger among the utility service providers

The values transmitted include the noise added consumption measurement and aggregated noise values. Therefore, actual consumption measurement values can not be obtained using this data. However, one issue to consider is the amount of noise that will be added. If the amount of the noise is low, then the noise added consumption values become close to the actual amounts and this can reveal information about the household. Therefore, the amount of the noise should not be too low. Thus, the proposed framework does not aim having low MAE values. Noise values can be also negative. While having high MAE, even if the noisy measurement values are low, absence of the household is not leaked.

In order to visualize this, assume that there are five measurement periods in a billing period, for the sake of simplicity. For a sample billing period, human presence, actual consumption values, noisy consumption values, noise values, and aggregate noise values are given in Table 5.1 for a simple smart home. Even though there is no human at the smart home in the measurement periods 4 and 5, assume that there is electricity

consumption due to the refrigerator. For the sake of simplicity, we assume that rates for the utilities are the same, i.e., the total of the noise values must be equal to 0 in a measurement period for a smart home.

Table 5.1: Sample measurement periods and the noise addition

Measurement Period Number	Human Presence	Actual Consumption Values (ce, cw, cg)	Noisy Consumption Values (nce, ncw, ncg)	Noise Values (ne, nw, ng)	Aggregate Noise Values (ane, anw, ang)
1	✓	6, 10, 4	3, 12, 5	-3, 2, 1	-3, 2, 1
2	✓	3, 4, 4	8, 7, -4	5, 3, -8	2, 5, -7
3	✓	6, 0, 0	4, -3, 5	-2, -3, 5	0, 2, -2
4	X	2, 0, 0	-1, -4, 7	-3, -4, 7	-3, -2, 5
5	X	2, 0, 0	4, -5, 3	2, -5, 3	-1, -7, 8
TOTAL		19, 14, 8	18, 7, 16		

For the considered billing period, total actual consumption values are {19, 14, 8} for electricity, water, and gas, respectively. Total of the noise added consumption values are {18, 7, 16}. The difference between these series are the aggregate noise values which are {-1, -7, 8}. These aggregate noise values are transmitted only at the end of the billing period. Although there is no human at the smart home in the measurement periods 4 and 5, the noisy consumption values transmitted at the end of the measurement periods do not allow inferring human absence at the house.

In order to examine the framework in terms of differential privacy theoretically, the formulation of differential privacy, given as Formula 2.1, has to be checked, and finding a counterexample to Formula 2.1 suffices to detect a violation of differential privacy. Assume that one wishes to find out whether there is anyone at a house or not. Let F be a function that gives the number of periods that have measurement values equal to the minimum measurement values. Let D_1 consists of measurement values for $n+1$ periods and D_2 consists of measurement values of n periods which are exactly the same as the first n periods of D_1, which makes D_1 and D_2 differ in a single row. The range of F is $[0, n+1]$ for D_1 and $[0, n]$ for D_2. The sensitivity of this function is 1, since adding a single row to any dataset will change the output by at most 1. To cover all possible datasets, the following cases must be considered for the $(n+1)^{st}$ period: (i) There is no

one at the house, there is no consumption, therefore the measurement values are equal to the minimum values, (ii) There is at least one person at the house. However, presence of someone at the house does not mean that there is any utility consumption. Therefore, two subcases occur: $(ii.i)$ There is no consumption, therefore the measurement values are equal to the minimum values, $(ii.ii)$ There is utility consumption, therefore the measurement values are greater than the minimum values. These cases and the corresponding $(n+1)^{st}$ period measurement values, relations between $F(D_1)$ and $F(D_2)$, differential privacy provision or violation statuses are given in Table 5.2.

Table 5.2: Differential privacy analysis of the traditional smart metering scenario

Case	Human Presence	$(n+1)^{st}$ Period Measurement Values	$F(D_1)$ and $F(D_2)$	Differential Privacy
i	X	= Minimum values	$F(D_1) \neq F(D_2)$ $F(D_1) = F(D_2) + 1$	X
$ii.i$	✓	= Minimum values	$F(D_1) \neq F(D_2)$ $F(D_1) = F(D_2) + 1$	X
$ii.ii$	✓	> Minimum values	$F(D_1) = F(D_2)$	✓

Without noise addition, i.e,, when the actual measurement values are transmitted, differential privacy is not provided in the human absence. In the presence of a human, differential privacy is provided in $1/2$ of the cases considered.

When noise is added to the measurement values to improve differential privacy, measurement values become different than the actual minimum measurement values. Moreover, even there are no consumption for two measurement periods, resulting actual minimum consumption measurement values, the noise added values differ for these two periods. Alternative cases of human absence or presence, the corresponding $(n+1)^{st}$ period measurement values, relations between $F(D_1)$ and $F(D_2)$, differential privacy provision or violation statuses after noise addition are given in Table 5.3.

Table 5.3: Differential privacy analysis of the proposed differentially-private smart metering framework

Case	Human Presence	$(n+1)^{st}$ Period Relation of Measurement Values with Minimum Values Before Noise	$(n+1)^{st}$ Period Relation of Noisy Measurement Values After Noise	$F(D_1)$ and $F(D_2)$ After Noise	Differential Privacy After Noise
i	X	= Minimum values	≠ Minimum values	$F(D_1) = F(D_2)$	✓
$ii.i$	✓	= Minimum values	≠ Minimum values	$F(D_1) = F(D_2)$	✓
$ii.ii.i$	✓	> Minimum values	≠ Minimum values	$F(D_1) = F(D_2)$	✓
$ii.ii.ii$	✓	> Minimum values	= Minimum values	$F(D_1) \neq F(D_2)$ $F(D_1) = F(D_2) + 1$	X

Differential privacy is provided for all the cases considered. This shows that adding noise to actual consumption values hides human absence or presence. To sum up, we hypothesize that by adding an adequate amount of noise and executing corresponding reconciliation algorithms, a blockchain-based differentially-private federated smart utility framework can be achieved.

5.4 Future Research Ideas

Determination of the correct amount of noise, and choice of the most efficient differential privacy parameter ε can be investigated as a future work. Besides, in addition to obfuscation of utility consumption values, methods of improving anonymity of smart home identities can be studied, as well. Moreover, smart homes may have different privacy requirements, therefore, the proposed framework may be extended to provide different privacy levels to answer these different privacy requirements.

Diversifying the noise addition algorithm may be another future research idea. One approach may be grouping smart homes and adding noise values to smart homes that are in the same group correlatively. In this case, the reconciliation and the billing algorithms must be modified accordingly, as well. If the reconciliation and the billing are not done

consecutively, i.e., the billing is done over noisy consumption values, and the reconciliation is done after a period of time, then the financial aspects must be considered. The tradeoff between privacy and the financial burden that the system outcomes, i.e., paying exactly as consumed versus paying lower or higher to be reconciled later, which brings up the interest issue, must be analyzed.

The proposed framework can be tested using a machine learning model that predicts human absence or presence, size of household, or categorizes household as young or elder. The machine learning model can be run before adding noise and after adding noise for a dataset, and the results can be compared to test differential privacy improvement.

6. CONCLUSION

In this dissertation, we investigate utilization of differential privacy in financial distributed ledgers. We start our examinations with Bitcoin, the cryptocurrency which brought about the emergence of blockchain, which is a distributed ledger infrastructure. In the related work, we present a survey, which analyzes state of the art anonymity and privacy studies in Bitcoin-like digital cash systems. We classified studies into two main categories; the studies that analyze anonymity and privacy and the studies that propose anonymity and privacy improvements. The first category focuses on revealing information by utilizing blockchain and network analysis, and deanonymization techniques. We examined the studies that take place in this category and provided a taxonomy. Examination of these studies clearly shows that Bitcoin requires anonymity and privacy improvements. As a result, numerous studies exist that include proposals for improving anonymity and privacy in Bitcoin-like digital cash systems. We examined these proposals as the second category and provided a taxonomy for them, as well. Many cryptographic protocols, like zero knowledge proofs, ring signatures, and homomorphic commitments, are utilized in these proposals. However, we have not encountered any studies utilizing differential privacy for improving anonymity in Bitcoin-like financial distributed ledgers. To remedy this absence, first, we present an examination of Bitcoin in terms of differential privacy. Our motivation arises from the fact that differential privacy approaches can be used for improving the privacy of the public Bitcoin blockchain. The differential privacy methods offer the prevention of anonymization and privacy breaches by direct queries and the preservation of checkability of the integrity of the blockchain. We first examine the current Bitcoin implementation using the differential privacy formulation. Then, we examine the application of noise addition to transaction amounts and user graph perturbation as differential privacy mechanisms. Furthermore, we demonstrate an empirical study for practical utilization of the noise addition approach

and compare four differential privacy mechanisms according to mean absolute error for varying ε and δ values. In addition, we introduce a new metric called *mean ranking offset* and use it for the comparison, as well. In Section 4.5, we summarize our observations. It is observed that the noise addition and the graph perturbation mechanisms decrease the fraction of the cases violating differential privacy, therefore they can be used for improving anonymity and privacy in Bitcoin. The noise addition method decreases the fraction of the cases violating differential privacy by half for the three query functions, whereas the graph perturbation method decreases the fraction of the cases violating differential privacy by half for all of the four query functions considered. When the differential privacy mechanisms are compared practically for the noise addition, it is demonstrated that the Geometric mechanism adds a marginal amount of noise for all considered ε values and this mechanism is ineffective at hiding the ranks of the amounts in the dataset. This allows an observer, searching for a transaction with a specific amount, to detect the transaction by finding the nearest noisy value even if the noises are added. Our experiments show that the Laplace mechanism outperforms other mechanisms with high MAE and MRO values, and it can be opted with ε equal or less than 0.5 for improving differential privacy in Bitcoin. Although the results that are obtained in this paper are promising, none of the proposed methods achieved perfect differential privacy. As another contribution of this dissertation, we propose a block-chain based differentially-private federated smart utility metering framework. We utilize noise addition approach for improving differential privacy. We detail key requirements, entities and roles, data structures, noise addition and reconciliation and billing algorithms of the proposed framework. We examine the framework in terms of information leakage and differential privacy. We theoretically show that differential privacy is provided, thus, noise addition approach can be used to improve privacy in blockchain-based smart utility metering scenario.

As the further research topics, modification of the Bitcoin verification mechanism according to the perturbation of differential privacy improving mechanisms, and examining the effect of the perturbation on the degradation of utility can be counted. Applying these differential privacy mechanisms to other blockchain-based cryptocurrencies may be investigated, as well. For the proposed blockchain-based differentially-private smart utility metering framework, determination of the correct amount of noise, and choice of the most differential privacy parameter ε can be investigated. Methods of improving anonymity of smart home identities can be studied

as another future work. Moreover, the proposed blockchain-based differentially-private smart utility metering framework can be tested using a machine learning model that predicts human absence or presence, size of household, or categorizes household like young or elder. Besides, noise addition algorithm of the framework can be diversified. One approach may be grouping smart homes and adding noise values to smart homes that are in the same group correlatively. In this case, reconciliation and billing algorithms must be modified accordingly, as well. In addition, smart homes may have different privacy requirements, therefore, the proposed framework may be extended to provide different privacy levels to answer these different privacy requirements.

BIBLIOGRAPHY

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., & Zhang, L. (2016b). Deep Learning with Differential Privacy. *Cornell University - ArXiv*. https://doi.org/10.1145/2976749.2978318

Abbas, Q. E., & Sung-Bong, J. (2019). A Survey of Blockchain and Its Applications. *International Conference on Artificial Intelligence*. https://doi.org/10.1109/icaiic.2019.8669067

Abdo, J. B., & Zeadally, S. (2020). Multi-utility framework: blockchain exchange platform for sustainable development. *International Journal of Pervasive Computing and Communications*, *18*(4), 388–406. https://doi.org/10.1108/ijpcc-06-2020-0059

Acs, G., & Castelluccia, C. (2012). DREAM: DiffeRentially privatE smArt Metering. *ArXiv: Cryptography and Security*.

Aitzhan, N. Z., & Svetinovic, D. (2018). Security and Privacy in Decentralized Energy Trading Through Multi-Signatures, Blockchain and Anonymous Messaging Streams. *IEEE Transactions on Dependable and Secure Computing*, *15*(5), 840–852. https://doi.org/10.1109/tdsc.2016.2616861

Aklilu, Y. T., & Ding, J. (2021). Survey on Blockchain for Smart Grid Management, Control, and Operation. *Energies*, *15*(1), 193. https://doi.org/10.3390/en15010193

Alsalami, N., & Zhang, B. (2019). SoK: A Systematic Study of Anonymity in Cryptocurrencies. *IEEE Conference Dependable and Secure Computing*. https://doi.org/10.1109/dsc47296.2019.8937681

Amarasinghe, N., Boyen, X., & McKague, M. (2019). A Survey of Anonymity of Cryptocurrencies. *Proceedings of the Australasian Computer Science Week Multiconference*. https://doi.org/10.1145/3290688.3290693

Andoni, M., Robu, V., Flynn, D., Abram, S., Geach, D., Jenkins, D., McCallum, P., & Peacock, A. (2019). Blockchain technology in the energy sector: A systematic review of challenges and opportunities. *Renewable & Sustainable Energy Reviews*, *100*, 143–174. https://doi.org/10.1016/j.rser.2018.10.014

Androulaki, E., & Karame, G. (2014). Hiding Transaction Amounts and Balances in Bitcoin. *Springer International Publishing EBooks*, 161–178. https://doi.org/10.1007/978-3-319-08593-7_11

Androulaki, E., Karame, G., Roeschlin, M., Scherer, T., & Capkun, S. (2013). Evaluating User Privacy in Bitcoin. *Springer Berlin Heidelberg EBooks*, 34–51. https://doi.org/10.1007/978-3-642-39884-1_4

Andrychowicz, M., Dziembowski, S., Malinowski, D., & Mazurek, L. (2014). Secure Multiparty Computations on Bitcoin. *IEEE Symposium on Security and Privacy*. https://doi.org/10.1109/sp.2014.35

Averin, A., Samartsev, A., & Sachenko, N. (2020). Review of Methods for Ensuring Anonymity and De-Anonymization in Blockchain. *2020 International Conference Quality Management, Transport and Information Security, Information Technologies (IT&QM&IS)*. https://doi.org/10.1109/itqmis51053.2020.9322974

Back, A. (2012). Hashcash - A Denial of Service Counter-Measure. http://www.hashcash.org/hashcash.pdf

Backes, M., & Meiser, S. (2013). Differentially Private Smart Metering with Battery Recharging. *Lecture Notes in Computer Science*, 194–212. https://doi.org/10.1007/978-3-642-54568-9_13

Banasik, W., Dziembowski, S., & Malinowski, D. (2016). Efficient Zero-Knowledge Contingent Payments in Cryptocurrencies Without Scripts. *Springer International Publishing EBooks*, 261–280. https://doi.org/10.1007/978-3-319-45741-3_14

Bao, H., & Lu, R. (2015). A New Differentially Private Data Aggregation with Fault Tolerance for Smart Grid Communications. *IEEE Internet of Things Journal*, *2*(3), 248–258. https://doi.org/10.1109/jiot.2015.2412552

Barbosa, P., Brito, A., & Almeida, H. (2016). A Technique to provide differential privacy for appliance usage in smart metering. *Information Sciences*, *370–371*, 355–367. https://doi.org/10.1016/j.ins.2016.08.011

Barber, S., Boyen, X., Shi, E., & Uzun, E. (2012). Bitter to Better — How to Make Bitcoin a Better Currency. *Lecture Notes in Computer Science*, 399–414. https://doi.org/10.1007/978-3-642-32946-3_29

Baumann, A., Fabian, B., & Lischke, M. (2014). Exploring the Bitcoin Network. *International Conference on Web Information Systems and Technologies*. https://doi.org/10.5220/0004937303690374

Ben-Sasson, E., Chiesa, A., Tromer, E., & Virza, M. (2013a). Succinct Non-Interactive Zero Knowledge for a von Neumann Architecture. *IACR Cryptology EPrint Archive*.

Ben-Sasson, E., Chiesa, A., Genkin, D., Tromer, E., & Virza, M. (2013b). SNARKs for C: Verifying Program Executions Succinctly and in Zero Knowledge. *Lecture Notes in Computer Science*, 90–108. https://doi.org/10.1007/978-3-642-40084-1_6

Ben-Sasson, E., Chiesa, A., Garman, C., Green, M., Miers, I., Tromer, E., & Virza, M. (2014). Zerocash: Decentralized Anonymous Payments from Bitcoin. *IEEE Symposium on Security and Privacy*. https://doi.org/10.1109/sp.2014.36

Bergman, K., & Rajput, S. (2021). Revealing and Concealing Bitcoin Identities: A Survey of Techniques. *Proceedings of the 3rd ACM International Symposium on Blockchain and Secure Critical Infrastructure*. https://doi.org/10.1145/3457337.3457838

Bernabe, J. B., Cánovas, J. L., Hernandez-Ramos, J. L., Moreno, R. T., & Skarmeta, A. F. (2019). Privacy-Preserving Solutions for Blockchain: Review and Challenges. *IEEE Access*, *7*, 164908–164940. https://doi.org/10.1109/access.2019.2950872

Bhutta, M. N. M., Khwaja, A. A., Nadeem, A., Ahmad, H. F., Khan, M. K., Hanif, M., Song, H., Alshamari, M. A., & Wang, T. (2021). A Survey on Blockchain Technology: Evolution, Architecture and Security. *IEEE Access*, *9*, 61048–61073. https://doi.org/10.1109/access.2021.3072849

Biryukov, A., Khovratovich, D., & Pustogarov, I. (2014). Deanonymisation of Clients in Bitcoin P2P Network. *Computer and Communications Security*. https://doi.org/10.1145/2660267.2660379

Biryukov, A., & Pustogarov, I. (2015). Bitcoin over Tor isn't a Good Idea. *IEEE Symposium on Security and Privacy*. https://doi.org/10.1109/sp.2015.15

Biryukov, A., & Tikhomirov, S. (2019a). Security and privacy of mobile wallet users in Bitcoin, Dash, Monero, and Zcash. *Pervasive and Mobile Computing*, *59*, 101030. https://doi.org/10.1016/j.pmcj.2019.101030

Biryukov, A., & Tikhomirov, S. (2019b). Transaction Clustering Using Network Traffic Analysis for Bitcoin and Derived Blockchains. *Conference on Computer Communications Workshops*. https://doi.org/10.1109/infcomw.2019.8845213

Bissias, G., Ozisik, A. P., Levine, B. N., & Liberatore, M. (2014). Sybil-Resistant Mixing for Bitcoin. *Workshop on Privacy in the Electronic Society*. https://doi.org/10.1145/2665943.2665955

Bistarelli, S., Mercanti, I., Faloci, F., & Santini, F. (2021). Highlighting Poor Anonymity and Security Practice in the Blockchain of Bitcoin. *The 36th ACM/SIGAPP Symposium on Applied Computing (SAC '21)*. https://doi.org/10.1145/3412841.3441909

Bittau, A., Erlingsson, Ú., Maniatis, P., Mironov, I., Raghunathan, A., Lie, D., Rudominer, M., Kode, U., Tinnes, J., & Seefeld, B. (2017). Prochlo. *ArXiv: Cryptography and Security*. https://doi.org/10.1145/3132747.3132769

Bleumer, G. (2005). Threshold Signature. In: van Tilborg, H.C.A. (eds) Encyclopedia of Cryptography and Security. Springer, Boston, MA . https://doi.org/10.1007/0-387-23483-7_429

Blum, M. (1983). Coin flipping by telephone a protocol for solving impossible problems. *SIGACT News*, *15*(1), 23–27. https://doi.org/10.1145/1008908.1008911

Bokhari, S. T., Aftab, T., Nadir, I., & Bakhshi, T. (2019). Exploring Blockchain-Secured Data Validation in Smart Meter Readings. *2019 22nd International Multitopic Conference (INMIC)*. https://doi.org/10.1109/inmic48123.2019.9022772

Boneh, D., Gentry, C., Lynn, B., & Shacham, H. (2003). Aggregate and Verifiably Encrypted Signatures from Bilinear Maps. *Lecture Notes in Computer Science*, 416–432. https://doi.org/10.1007/3-540-39200-9_26

Bonneau, J., Miller, A., Clark, J., Narayanan, A., Kroll, J. A., & Felten, E. W. (2015). SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies. *IEEE Symposium on Security and Privacy*. https://doi.org/10.1109/sp.2015.14

Bonneau, J., Narayanan, A., Miller, A., Clark, J., Kroll, J. A., & Felten, E. W. (2014). Mixcoin: Anonymity for Bitcoin with Accountable Mixes. *Lecture Notes in Computer Science*, 486–504. https://doi.org/10.1007/978-3-662-45472-5_31

Boudot, F. (2000). Efficient Proofs that a Committed Number Lies in an Interval. *Lecture Notes in Computer Science*, 431–444. https://doi.org/10.1007/3-540-45539-6_31

Bradbury, D. (2014). Anonymity and privacy: a guide for the perplexed. *Network Security*. https://doi.org/10.1016/s1353-4858(14)70102-3

Buterin, V. (2014). Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform. https://ethereum.org/669c9e2e2027310b6b3cdce6e1c52962/Ethereum_Whitepaper_-_Buterin_2014.pdf

Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., & Maxwell, G. (2018). Bulletproofs: Short Proofs for Confidential Transactions and More. *IEEE Symposium on Security and Privacy*. https://doi.org/10.1109/sp.2018.00020

Bünz, B., Agrawal, S., Zamani, M., & Boneh, D. (2020). Zether: Towards Privacy in a Smart Contract World. *Springer International Publishing EBooks*, 423–443. https://doi.org/10.1007/978-3-030-51280-4_23

Bürer, M. J., De Lapparent, M., Pallotta, V., Capezzali, M., & Carpita, M. (2019). Use cases for Blockchain in the Energy Industry Opportunities of emerging business models and related risks. *Computers & Industrial Engineering*, *137*, 106002. https://doi.org/10.1016/j.cie.2019.106002

Campanelli, M., Gennaro, R., Goldfeder, S., & Nizzardo, L. (2017). Zero-Knowledge Contingent Payments Revisited. *Computer and Communications Security*. https://doi.org/10.1145/3133956.3134060

Chaum, D. (1981). Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, *24*(2), 84–90. https://doi.org/10.1145/358549.358563

Chaum, D. (1983). Blind Signatures for Untraceable Payments. *Springer US EBooks*, 199–203. https://doi.org/10.1007/978-1-4757-0602-4_18

Chaum, D. (1988). The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology*, *1*(1), 65–75. https://doi.org/10.1007/bf00206326

Chaum, D., Das, D., Javani, F., Kate, A., Krasnova, A., De Ruiter, J., & Sherman, A. T. (2017). cMix: Mixing with Minimal Real-Time Asymmetric Cryptographic Operations. *Lecture Notes in Computer Science*, 557–578. https://doi.org/10.1007/978-3-319-61204-1_28

Chen, X., Ji, J., Luo, C., Liao, W., & Li, P. (2018). When Machine Learning Meets Blockchain: A Decentralized, Privacy-preserving and Secure Design. *International Conference on Big Data*. https://doi.org/10.1109/bigdata.2018.8622598

Chen, X., Zhang, D., Cui, Z., Gu, Q., & Ju, X. (2019). DP-Share: Privacy-Preserving Software Defect Prediction Model Sharing Through Differential Privacy. *Journal of Computer Science and Technology*, *34*(5), 1020–1038. https://doi.org/10.1007/s11390-019-1958-0

Conti, M., Kumar, E. S., Lal, C., & Ruj, S. (2018). A Survey on Security and Privacy Issues of Bitcoin. *IEEE Communications Surveys and Tutorials*, *20*(4), 3416–3452. https://doi.org/10.1109/comst.2018.2842460

Corrigan-Gibbs, H., & Ford, B. (2010). Dissent. *Computer and Communications Security*. https://doi.org/10.1145/1866307.1866346

Coutu, O. (2014). Privacy in Bitcoin through decentralized mixers. *Master's Thesis, Université de Montréal*.

Crépeau, C. (2005). Cut-and-choose protocol. In: van Tilborg, H.C.A. (eds) Encyclopedia of Cryptography and Security. Springer, Boston, MA . https://doi.org/10.1007/0-387-23483-7_92

Damgård, I., Geisler, M., Krøigaard, M., & Nielsen, J. B. (2009). Asynchronous Multiparty Computation: Theory and Implementation. *Lecture Notes in Computer Science*, 160–179. https://doi.org/10.1007/978-3-642-00468-1_10

Danezis, G., Fournet, C., Kohlweiss, M., & Parno, B. (2013). Pinocchio coin. *Privacy Enhancing Technologies*. https://doi.org/10.1145/2517872.2517878

Dankar, F. K., & Emam, K. E. (2012). The application of differential privacy to health data. *EDBT/ICDT Workshops*. https://doi.org/10.1145/2320765.2320816

Dave, D., Parikh, S., Patel, R., & Doshi, N. (2019). A Survey on Blockchain Technology and its Proposed Solutions. *Procedia Computer Science*, *160*, 740–745. https://doi.org/10.1016/j.procs.2019.11.017

Davenport, D. (2002). Anonymity on the Internet: why the price may be too high. *Communications of the ACM*, *45*(4), 33–35. https://doi.org/10.1145/505248.505267

Dingledine, R., Mathewson, N., & Syverson, P. (2004). Tor: The Second-Generation Onion Router. *USENIX Security Symposium*. https://doi.org/10.21236/ada465464

Dorri, A., Luo, F., Kanhere, S. S., Jurdak, R., & Zhang, R. (2019b). SPB: A Secure Private Blockchain-Based Solution for Distributed Energy Trading. *IEEE Communications Magazine*, *57*(7), 120–126. https://doi.org/10.1109/mcom.2019.1800577

Duan, H., Zheng, Y., Du, Y., Zhou, A., Wang, C., & Au, M. H. (2019). Aggregating Crowd Wisdom via Blockchain: A Private, Correct, and Robust Realization. *IEEE International Conference on Pervasive Computing and Communications*. https://doi.org/10.1109/percom.2019.8767412

Dupont, J., & Squicciarini, A. (2015). Toward De-Anonymizing Bitcoin by Mapping Users Location. *Conference on Data and Application Security and Privacy*. https://doi.org/10.1145/2699026.2699128

Dwork, C. (2006). Differential Privacy. *Lecture Notes in Computer Science*, 1–12. https://doi.org/10.1007/11787006_1

Dwork, C., McSherry, F., Nissim, K., & Smith, A. (2006). Calibrating Noise to Sensitivity in Private Data Analysis. *Lecture Notes in Computer Science*, 265–284. https://doi.org/10.1007/11681878_14

Dwork, C., & Roth, A. (2014). The Algorithmic Foundations of Differential Privacy. *Foundations and Trends® in Theoretical Computer Science*. https://doi.org/10.1561/9781601988195

Eberhardt, J., Peise, M., Kim, D., & Tai, S. (2020). Privacy-Preserving Netting in Local Energy Grids. *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. https://doi.org/10.1109/icbc48266.2020.9169440

Eberhardt, J., & Tai, S. (2018). ZoKrates - Scalable Privacy-Preserving Off-Chain Computations. *Green Computing and Communications*. https://doi.org/10.1109/cybermatics_2018.2018.00199

Eckhoff, D., & Wagner, I. (2018). Privacy in the Smart City—Applications, Technologies, Challenges, and Solutions. *IEEE Communications Surveys and Tutorials*, *20*(1), 489–516. https://doi.org/10.1109/comst.2017.2748998

Eibl, G., & Engel, D. (2017). Differential privacy for real smart metering data. *Computer Science - Research and Development*, *32*(1–2), 173–182. https://doi.org/10.1007/s00450-016-0310-y

Eisele, S., Laszka, A., Schmidt, D. C., & Dubey, A. (2020). The Role of Blockchains in Multi-Stakeholder Transactive Energy Systems. *Frontiers in Blockchain*, *3*. https://doi.org/10.3389/fbloc.2020.593471

Erdin, E., Cebe, M., Akkaya, K., Bulut, E., & Uluagac, A. S. (2021). A scalable private Bitcoin payment channel network with privacy guarantees. *Journal of Network and Computer Applications*, *180*, 103021. https://doi.org/10.1016/j.jnca.2021.103021

Erdin, E., Zachor, C., & Gunes, M. H. (2015). How to Find Hidden Users: A Survey of Attacks on Anonymity Networks. *IEEE Communications Surveys and Tutorials*, *17*(4), 2296–2316. https://doi.org/10.1109/comst.2015.2453434

Erlingsson, Ú., Pihur, V., & Korolova, A. (2014). RAPPOR. *ArXiv: Cryptography and Security*. https://doi.org/10.1145/2660267.2660348

Fabian, B., Ermakova, T., Krah, J., Lando, E., & Ahrary, N. (2018). Adoption of Security and Privacy Measures in Bitcoin Stated and Actual Behavior. *Social Science Research Network*. https://doi.org/10.2139/ssrn.3184130

Fanti, G., & Viswanath, P. (2017). Anonymity Properties of the Bitcoin P2P Network. *ArXiv: Cryptography and Security*.

Farokhi, F. (2020). Review of results on smart-meter privacy by data manipulation, demand shaping, and load scheduling. *IET Smart Grid*, *3*(5), 605–613. https://doi.org/10.1049/iet-stg.2020.0129

Fei, T., Chang, Y., Wang, J., Lu, N., & Shi, W. (2020). Anonymous Bitcoin Mixing Scheme Based on Semi-Trusted Supervisor. *2020 IEEE 3rd International Conference on Electronics Technology (ICET)*. https://doi.org/10.1109/icet49382.2020.9119602

Feld, S., Schönfeld, M., & Werner, M. (2014). Analyzing the Deployment of Bitcoin's P2P Network under an AS-level Perspective. *Procedia Computer Science*, *32*, 1121–1126. https://doi.org/10.1016/j.procs.2014.05.542

Feng, Q., He, D., Zeadally, S., Khan, M. K., & Kumar, N. (2019). A survey on privacy protection in blockchain system. *Journal of Network and Computer Applications*, *126*, 45–58. https://doi.org/10.1016/j.jnca.2018.10.020

Ferrag, M. A., Maglaras, L. A., & Ahmim, A. (2017). Privacy-Preserving Schemes for Ad Hoc Social Networks: A Survey. *IEEE Communications Surveys and Tutorials*, *19*(4), 3015–3045. https://doi.org/10.1109/comst.2017.2718178

Firoozjaei, M. D., Ghorbani, A. A., Kim, H., & Song, J. (2020). Hy-Bridge: A Hybrid Blockchain for Privacy-Preserving and Trustful Energy Transactions in Internet-of-Things Platforms. *Sensors*, *20*(3), 928. https://doi.org/10.3390/s20030928

Fleder, M., Kester, M. S., & Pillai, S. (2015). Bitcoin Transaction Graph Analysis. *ArXiv: Cryptography and Security*.

Florian, M., Walter, J., & Baumgart, I. (2015). Sybil-Resistant Pseudonymization and Pseudonym Change without Trusted Third Parties. *Workshop on Privacy in the Electronic Society*. https://doi.org/10.1145/2808138.2808145

Fotiou, N., Pittaras, I., Siris, V. A., Polyzos, G. C., & Anton, P. (2021). A privacy-preserving statistics marketplace using local differential privacy and blockchain: An application to smart-grid measurements sharing. *Blockchain: Research and Applications*, *2*(1), 100022. https://doi.org/10.1016/j.bcra.2021.100022

Franklin, M., & Zhang, H. (2012). A Framework for Unique Ring Signatures. *Cryptology ePrint Archive, Paper 2012/577.*

Fuchsbauer, G. (2009). Automorphic Signatures in Bilinear Groups and an Application to Round-Optimal Blind Signatures. *IACR Cryptology EPrint Archive*, *2009*, 320.

Fujisaki, E., & Suzuki, K. (2007). Traceable Ring Signature. *Springer Berlin Heidelberg EBooks*, 181–200. https://doi.org/10.1007/978-3-540-71677-8_13

Gai, K., Wu, Y., Zhu, L., Qiu, M., & Shen, M. (2019). Privacy-Preserving Energy Trading Using Consortium Blockchain in Smart Grid. IEEE Transactions on Industrial Informatics, 15(6), 3548–3558. https://doi.org/10.1109/tii.2019.2893433

Geng, Q., & Viswanath, P. (2016). Optimal Noise Adding Mechanisms for Approximate Differential Privacy. *IEEE Transactions on Information Theory*, *62*(2), 952–969. https://doi.org/10.1109/tit.2015.2504972

Genkin, D., Papadopoulos, D., & Papamanthou, C. (2018). Privacy in decentralized cryptocurrencies. *Communications of the ACM*, *61*(6), 78–88. https://doi.org/10.1145/3132696

Ghesmati, S., Fdhila, W., & Weippl, E. (2022). SoK: How private is Bitcoin? Classification and Evaluation of Bitcoin Privacy Techniques. *Proceedings of the 17th International Conference on Availability, Reliability and Security*. https://doi.org/10.1145/3538969.3538971

Ghosh, A., Roughgarden, T., & Sundararajan, M. (2009). Universally utility-maximizing privacy mechanisms. *Symposium on the Theory of Computing*. https://doi.org/10.1145/1536414.1536464

Goldwasser, S., Micali, S., & Rackoff, C. (1989). The Knowledge Complexity of Interactive Proof Systems. *SIAM Journal on Computing*, *18*(1), 186–208. https://doi.org/10.1137/0218012

Golle, P., & Juels, A. (2004). Dining Cryptographers Revisited. *Springer Berlin Heidelberg EBooks*, 456–473. https://doi.org/10.1007/978-3-540-24676-3_27

Gough, M., Santos, S. F., Van Sark, W., Javadi, M. S., Castro, R. E., & Catalao, J. P. S. (2022). Preserving Privacy of Smart Meter Data in a Smart Grid Environment. *IEEE Transactions on Industrial Informatics*, *18*(1), 707–718. https://doi.org/10.1109/tii.2021.3074915

Guan, Z., Si, G., Zhang, X., Wu, L., Guizani, N., Du, X., & Ma, Y. (2018). Privacy-Preserving and Efficient Aggregation Based on Blockchain for Power Grid Communications in Smart Communities. *IEEE Communications Magazine*, *56*(7), 82–88. https://doi.org/10.1109/mcom.2018.1700401

Guo, Y., Wan, Z., & Cheng, X. (2022). When blockchain meets smart grids: A comprehensive survey. *High-Confidence Computing*, *2*(2), 100059. https://doi.org/10.1016/j.hcc.2022.100059

Hassan, M. U., Rehmani, M. H., Kotagiri, R., Zhang, J., & Chen, J. (2019). Differential privacy for renewable energy resources based smart metering. *Journal of Parallel and Distributed Computing*, *131*, 69–80. https://doi.org/10.1016/j.jpdc.2019.04.012

Hassan, M. U., Rehmani, M. H., & Chen, J. (2020a). Differential Privacy Techniques for Cyber Physical Systems: A Survey. *IEEE Communications Surveys and Tutorials*, *22*(1), 746–789. https://doi.org/10.1109/comst.2019.2944748

Hassan, M. U., Rehmani, M. H., & Chen, J. (2020b). Performance Evaluation of Differential Privacy Mechanisms in Blockchain based Smart Metering. *Cornell University - ArXiv*. https://doi.org/10.48550/arxiv.2007.09802

Hassan, M. U., Rehmani, M. H., & Chen, J. (2020c). Differential privacy in blockchain technology: A futuristic approach. *Journal of Parallel and Distributed Computing*, *145*, 50–74. https://doi.org/10.1016/j.jpdc.2020.06.003

Hassan, M. U., Rehmani, M. H., & Chen, J. (2020d). DEAL: Differentially Private Auction for Blockchain based Microgrids Energy Trading. *IEEE Transactions on Services Computing*, 1. https://doi.org/10.1109/tsc.2019.2947471

Heilman, E., Alshenibr, L., Baldimtsi, F., Scafuro, A., & Goldberg, S. (2017). TumbleBit: An Untrusted Bitcoin-Compatible Anonymous Payment Hub. *Network and Distributed System Security Symposium*. https://doi.org/10.14722/ndss.2017.23086

Heilman, E., Baldimtsi, F., & Goldberg, S. (2016). Blindly Signed Contracts: Anonymous On-Blockchain and Off-Blockchain Bitcoin Transactions. *Lecture Notes in Computer Science*, 43–60. https://doi.org/10.1007/978-3-662-53357-4_4

Herrera-Joancomartí, J. (2014). Research and Challenges on Bitcoin Anonymity. *Lecture Notes in Computer Science*, 3–16. https://doi.org/10.1007/978-3-319-17016-9_1

Holohan, N., Braghin, S., Mac Aonghusa, P., & Levacher, K. (2019). Diffprivlib: The IBM Differential Privacy Library. *ArXiv: Cryptography and Security*.

Hong, Y., Kwon, H., Lee, J., & Hur, J. (2018). A Practical De-mixing Algorithm for Bitcoin Mixing Services. *Proceedings of the 2nd ACM Workshop on Blockchains, Cryptocurrencies, and Contracts - BCC '18*. https://doi.org/10.1145/3205230.3205234

Hossain, M. B., Natgunanathan, I., Xiang, Y., & Zhang, Y. (2021). Cost-Friendly Differential Privacy of Smart Meters Using Energy Storage and Harvesting Devices. *IEEE Transactions on Services Computing*, *15*(5), 2648–2657. https://doi.org/10.1109/tsc.2021.3081170

Hynes, N., Dao, D., Yan, D., Cheng, R., & Song, D. (2018). A demonstration of sterling. *Proceedings of the VLDB Endowment*, *11*(12), 2086–2089. https://doi.org/10.14778/3229863.3236266

Ibrahim, M. H. (2017). SecureCoin: A Robust Secure and Efficient Protocol for Anonymous Bitcoin Ecosystem. *International Journal of Network Security*, *19*(2), 295–312. https://doi.org/10.6633/ijns.201703.19(2).14

Jawaheri, H. A., Sabah, M. A., Boshmaf, Y., & Erbad, A. (2020). Deanonymizing Tor hidden service users through Bitcoin transactions analysis. *Computers & Security*, *89*, 101684. https://doi.org/10.1016/j.cose.2019.101684

Jayasinghe, D., Markantonakis, K., & Mayes, K. (2014). Optimistic Fair-Exchange with Anonymity for Bitcoin Users. *International Conference on E-Business Engineering*. https://doi.org/10.1109/icebe.2014.20

Jogenfors, J. (2016). Quantum Bitcoin: An Anonymous, Distributed, and Secure Currency Secured by the No-Cloning Theorem of Quantum Mechanics. *ArXiv: Quantum Physics*. https://doi.org/10.1109/bloc.2019.8751473

Johnson, D. H., Menezes, A., & Vanstone, S. A. (2001). The Elliptic Curve Digital Signature Algorithm (ECDSA). *International Journal of Information Security*, *1*(1), 36–63. https://doi.org/10.1007/s102070100002

Jorgensen, Z., Yu, T., & Cormode, G. (2016). Publishing Attributed Social Graphs with Formal Privacy Guarantees. *International Conference on Management of Data*. https://doi.org/10.1145/2882903.2915215

Jourdan, M., Blandin, S., Wynter, L., & Deshpande, P. (2018). Characterizing Entities in the Bitcoin Blockchain. *International Conference on Data Mining*. https://doi.org/10.1109/icdmw.2018.00016

Kabalci, Y. (2016). A survey on smart metering and smart grid communication. *Renewable & Sustainable Energy Reviews*, *57*, 302–318. https://doi.org/10.1016/j.rser.2015.12.114

Kelly, D., Raines, R. A., Baldwin, R. O., Grimaila, M. R., & Mullins, B. E. (2012). Exploring Extant and Emerging Issues in Anonymous Networks: A Taxonomy and Survey of Protocols and Metrics. *IEEE Communications Surveys and Tutorials*, *14*(2), 579–606. https://doi.org/10.1109/surv.2011.042011.00080

Knirsch, F., Unterweger, A., Eibl, G., & Engel, D. (2018). Privacy-Preserving Smart Grid Tariff Decisions with Blockchain-Based Smart Contracts. *Springer International Publishing EBooks*, 85–116. https://doi.org/10.1007/978-3-319-62238-5_4

Kosba, A. E., Miller, A., Shi, E., Wen, Z., & Papamanthou, C. (2016). Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts. *IEEE Symposium on Security and Privacy*. https://doi.org/10.1109/sp.2016.55

Koshy, P., Koshy, D., & McDaniel, P. (2014). An Analysis of Anonymity in Bitcoin Using P2P Network Traffic. *Lecture Notes in Computer Science*, 469–485. https://doi.org/10.1007/978-3-662-45472-5_30

Kserawi, F., Al-Marri, S., & Malluhi, Q. (2022). Privacy-Preserving Fog Aggregation of Smart Grid Data Using Dynamic Differentially-Private Data Perturbation. *IEEE Access*, *10*, 43159–43174. https://doi.org/10.1109/access.2022.3167015

Kumar, E. S. (2020). Preserving Privacy in Ethereum Blockchain. *Annals of Data Science*. https://doi.org/10.1007/s40745-020-00279-9

Kus, M. C., & Levi, A. (2022). Investigation and Application of Differential Privacy in Bitcoin. *IEEE Access*, *10*, 25534-25554. https://doi.org/ 10.1109/ACCESS.2022.3151784

Kus-Khalilov, M. C., & Levi, A. (2018). A Survey on Anonymity and Privacy in Bitcoin-Like Digital Cash Systems. *IEEE Communications Surveys and Tutorials*, *20*(3), 2543–2585. https://doi.org/10.1109/comst.2018.2818623

Ladd, W. (2012). Blind Signatures For Bitcoin Transaction Anonymity. https://wbl.github.io/bitcoinanon.pdf

Lamport, L., Shostak, R. E., & Pease, M. C. (1982). The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems*, *4*(3), 382–401. https://doi.org/10.1145/357172.357176

Lazar, D., Gilad, Y., & Zeldovich, N. (2018). Karaoke: distributed private messaging immune to passive traffic analysis. *Operating Systems Design and Implementation*, 711–725. https://doi.org/10.5555/3291168.3291221

Lee, N., Yang, J., Onik, M. H., & Kim, C. (2019). Modifiable Public Blockchains Using Truncated Hashing and Sidechains. *IEEE Access*, *7*, 173571–173582. https://doi.org/10.1109/access.2019.2956628

Li, F., Luo, B., & Liu, P. (2010). Secure Information Aggregation for Smart Grids Using Homomorphic Encryption. *International Conference on Smart Grid Communications*. https://doi.org/10.1109/smartgrid.2010.5622064

Lischke, M., & Fabian, B. (2016). Analyzing the Bitcoin Network: The First Four Years. *Future Internet*, *8*(4), 7. https://doi.org/10.3390/fi8010007

Liu, Y., Liu, X., Tang, C., Wang, J., & Zhang, L. (2018). Unlinkable Coin Mixing Scheme for Transaction Privacy Enhancement of Bitcoin. *IEEE Access*, *6*, 23261–23270. https://doi.org/10.1109/access.2018.2827163

Liu, X., Wang, H., Chen, G., Zhou, B., & Rehman, A. U. (2021). Intermittently differential privacy in smart meters via rechargeable batteries. *Electric Power Systems Research*, *199*, 107410. https://doi.org/10.1016/j.epsr.2021.107410

Lu, N., Chang, Y., Shi, W., & Choo, K. R. (2020). CoinLayering: An Efficient Coin Mixing Scheme for Large Scale Bitcoin Transactions. *IEEE Transactions on Dependable and Secure Computing*, *19*(3), 1974–1987. https://doi.org/10.1109/tdsc.2020.3043366

Lua, E. K., Crowcroft, J., Pias, M., Sharma, R. S., & Lim, S. (2005). A survey and comparison of peer-to-peer overlay network schemes. *IEEE Communications Surveys and Tutorials*, *7*(2), 72–93. https://doi.org/10.1109/comst.2005.1610546

Lukionav, D. (2015). Compact Confidential Transactions for Bitcoin. http://www.voxelsoft.com/dev/cct.html

Marks, J., Montano, B., Chong, J., Raavi, M., Islam, R., Cerny, T., & Shin, D. (2021). Differential privacy applied to smart meters. *ACM Symposium on Applied Computing*. https://doi.org/10.1145/3412841.3442360

Maurer, F. K. (2016). A survey on approaches to anonymity in bitcoin and other cryptocurrencies. *GI-Jahrestagung*, 2145–2150.

Maxwell, G., & Poelstra, A. (2015). Borromean Ring Signatures. http://diyhpl.us/~bryan/papers2/bitcoin/Borromean%20ring%20signatures.pdf

Meiklejohn, S., & Orlandi, C. (2015). Privacy-Enhancing Overlays in Bitcoin. *Springer Berlin Heidelberg EBooks*, 127–141. https://doi.org/10.1007/978-3-662-48051-9_10

Meiklejohn, S., Pomarole, M., Jordan, G., Levchenko, K., McCoy, D., Voelker, G. M., & Savage, S. (2013). A fistful of bitcoins. *Internet Measurement Conference*. https://doi.org/10.1145/2504730.2504747

Mercer, R. (2016). Privacy on the Blockchain: Unique Ring Signatures. *ArXiv: Cryptography and Security*.

Merkle, R. C. (1987). A Digital Signature Based on a Conventional Encryption Function. *Lecture Notes in Computer Science*, 369–378. https://doi.org/10.1007/3-540-48184-2_32

Miers, I., Garman, C., Green, M., & Rubin, A. D. (2013). Zerocoin: Anonymous Distributed E-Cash from Bitcoin. *IEEE Symposium on Security and Privacy*. https://doi.org/10.1109/sp.2013.34

Molina-Markham, A., Shenoy, P., Fu, K., Cecchet, E., & Irwin, D. (2010). Private memoirs of a smart meter. *ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings*. https://doi.org/10.1145/1878431.1878446

Mollah, M. B., Zhao, J., Niyato, D., Lam, K., Zhang, X., Ghias, A. M. Y. M., Koh, L. H., & Yang, L. (2019). Blockchain for Future Smart Grid: A Comprehensive Survey.

IEEE Internet of Things Journal, *8*(1), 18–43. https://doi.org/10.1109/jiot.2020.2993601

Monrat, A. A., Schelen, O., & Andersson, K. (2019). A Survey of Blockchain From the Perspectives of Applications, Challenges, and Opportunities. *IEEE Access*, *7*, 117134–117151. https://doi.org/10.1109/access.2019.2936094

Möser, M., Böhme, R., & Breuker, D. (2013). An inquiry into money laundering tools in the Bitcoin ecosystem. *2013 APWG ECrime Researchers Summit*. https://doi.org/10.1109/ecrs.2013.6805780

Muratori, M. (2018). Impact of uncoordinated plug-in electric vehicle charging on residential power demand. *Nature Energy*, *3*(3), 193–201. https://doi.org/10.1038/s41560-017-0074-z

Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. https://bitcoin.org/bitcoin.pdf

Narayanan, A., Bonneau, J., Felten. E. & Goldfeder, S. (2016). *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton University Press.

Neudecker, T., & Hartenstein, H. (2017). Could Network Information Facilitate Address Clustering in Bitcoin? *Springer International Publishing EBooks*, 155–169. https://doi.org/10.1007/978-3-319-70278-0_9

Nguyen, C. T., Hoang, D. T., Nguyen, D. N., Niyato, D., Nguyen, H. T., & Dutkiewicz, E. (2019). Proof-of-Stake Consensus Mechanisms for Future Blockchain Networks: Fundamentals, Applications and Opportunities. *IEEE Access*, *7*, 85727–85745. https://doi.org/10.1109/access.2019.2925010

Ni, J., Zhang, K., Alharbi, K., Lin, X., Zhang, N., & Shen, X. (2017). Differentially Private Smart Metering With Fault Tolerance and Range-Based Filtering. *IEEE Transactions on Smart Grid*, *8*(5), 2483–2493. https://doi.org/10.1109/tsg.2017.2673843

Nick, J. D. (2015). Data-Driven De-Anonymization in Bitcoin. *Master's Thesis ETH Zürich*. https://doi.org/10.3929/ethz-a-010541254

Noether, S., & Mackenzie, A. (2016). Ring Confidential Transactions. *Ledger*, *1*, 1–18. https://doi.org/10.5195/ledger.2016.34

Ober, M., Katzenbeisser, S., & Hamacher, K. (2013). Structure and Anonymity of the Bitcoin Transaction Graph. *Future Internet*, *5*(2), 237–250. https://doi.org/10.3390/fi5020237

Ortega, M. S. (2013). The Bitcoin Transaction Graph Anonymity. Master's Thesis, Universitat Autònoma de Barcelona, 2013.

Parno, B., Howell, J., Gentry, C., & Raykova, M. (2013). Pinocchio: Nearly Practical Verifiable Computation. *IEEE Symposium on Security and Privacy*. https://doi.org/10.1109/sp.2013.47

Peng, L., Feng, W., Yan, Z., Li, Y., Zhou, X., & Shimizu, S. (2021). Privacy preservation in permissionless blockchain: A survey. *Digital Communications and Networks*, *7*(3), 295–307. https://doi.org/10.1016/j.dcan.2020.05.008

Politou, E., Casino, F., Alepis, E., & Patsakis, C. (2020). Blockchain Mutability: Challenges and Proposed Solutions. *IEEE Transactions on Emerging Topics in Computing*, *9*(4), 1972–1986. https://doi.org/10.1109/tetc.2019.2949510

Reed, M. G., Syverson, P., & Goldschlag, D. M. (1998). Anonymous connections and onion routing. *IEEE Journal on Selected Areas in Communications*, *16*(4), 482–494. https://doi.org/10.1109/49.668972

Reid, F., & Harrigan, M. (2012). An Analysis of Anonymity in the Bitcoin System. *International Conference on Social Computing*. https://doi.org/10.1109/passat/socialcom.2011.79

Rivest, R. L., Shamir, A., & Tauman, Y. (2001). How to Leak a Secret. *Lecture Notes in Computer Science*, 552–565. https://doi.org/10.1007/3-540-45682-1_32

Ron, D., & Shamir, A. (2013). Quantitative Analysis of the Full Bitcoin Transaction Graph. *Lecture Notes in Computer Science*, 6–24. https://doi.org/10.1007/978-3-642-39884-1_2

Ruffing, T., Moreno-Sanchez, P., & Kate, A. (2014). CoinShuffle: Practical Decentralized Coin Mixing for Bitcoin. *Springer International Publishing EBooks*, 345–364. https://doi.org/10.1007/978-3-319-11212-1_20

Ruffing, T., Moreno-Sanchez, P., & Kate, A. (2017). P2P Mixing and Unlinkable Bitcoin Transactions. *Network and Distributed System Security Symposium*. https://doi.org/10.14722/ndss.2017.23415

Ruj, S., & Nayak, A. (2013). A Decentralized Security Framework for Data Aggregation and Access Control in Smart Grids. *IEEE Transactions on Smart Grid*, *4*(1), 196–205. https://doi.org/10.1109/tsg.2012.2224389

Saberhagen, N. v.. (2013) CryptoNote v2.0. https://bytecoin.org/old/whitepaper.pdf

Sala, A., Zhao, X., Wilson, C., Zheng, H., & Zhao, B. Y. (2011). Sharing graphs using differentially private graph models. *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference - IMC '11*. https://doi.org/10.1145/2068816.2068825

Saputro, N., & Akkaya, K. (2012). Performance evaluation of Smart Grid data aggregation via homomorphic encryption. *Wireless Communications and Networking Conference*. https://doi.org/10.1109/wcnc.2012.6214307

Saxena, A., Misra, J., & Dhar, A. (2014). Increasing Anonymity in Bitcoin. *Lecture Notes in Computer Science*. https://doi.org/10.1007/978-3-662-44774-1_9

Schaffner, T. (2014). Bitcoin Anonymity and Security. http://www.cs.tufts.edu/comp/116/archive/fall2014/tschaffner.pdf

Schunter, M. (2005). Fair Exchange. In: van Tilborg, H.C.A. (eds) Encyclopedia of Cryptography and Security. Springer, Boston, MA . https://doi.org/10.1007/0-387-23483-7_155

Shafiq, O. (2019). Bitcoin Transactions Data 2011–2013. *IEEE DataPort.*

Shen Tu, Q., & Yu, J. (2015a). Research on Anonymization and Deanonymization in the Bitcoin system. *arXiv:1510.07782.*

ShenTu, Q., & Yu, J. (2015b). Transaction Remote Release (TRR): A New Anonymization Technology for Bitcoin. *arXiv: 1509.06160.*

ShenTu, Q., & Yu, J. (2015b). A Blind-Mixing Scheme for Bitcoin based on an Elliptic Curve Cryptography Blind Digital Signature Algorithm. *ArXiv: Cryptography and Security*.

Spagnuolo, M., Maggi, F., & Zanero, S. (2014). BitIodine: Extracting Intelligence from the Bitcoin Network. *Lecture Notes in Computer Science*, 457–468. https://doi.org/10.1007/978-3-662-45472-5_29

Tikhomirov, S. (2017). Ethereum: State of Knowledge and Research Perspectives. *Lecture Notes in Computer Science*, 206–221. https://doi.org/10.1007/978-3-319-75650-9_14

Tschorsch, F., & Scheuermann, B. (2016). Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies. *IEEE Communications Surveys and Tutorials*, *18*(3), 2084–2123. https://doi.org/10.1109/comst.2016.2535718

Tong, W., Hua, J., & Zhong, S. (2017). A Jointly Differentially Private Scheduling Protocol for Ridesharing Services. *IEEE Transactions on Information Forensics and Security*, *12*(10), 2444–2456. https://doi.org/10.1109/tifs.2017.2707334

Torra, V., & Salas, J. (2019). Graph Perturbation as Noise Graph Addition: A New Perspective for Graph Anonymization. *Lecture Notes in Computer Science*, 121–137. https://doi.org/10.1007/978-3-030-31500-9_8

Xueshuo, X., Jiming, W., Junyi, Y., Yaozheng, F., Ye, L., Tao, L., & Guiling, W. (2021). AWAP: Adaptive weighted attribute propagation enhanced community detection model for bitcoin de-anonymization. *Applied Soft Computing*, *109*, 107507. https://doi.org/10.1016/j.asoc.2021.107507

Valenta, L., & Rowan, B. (2015). Blindcoin: Blinded, Accountable Mixes for Bitcoin. *Springer Berlin Heidelberg EBooks*, 112–126. https://doi.org/10.1007/978-3-662-48051-9_9

Venkatakrishnan, S. B., Kairouz, P., & Viswanath, P. (2017). Dandelion: Redesigning the Bitcoin Network for Anonymity. *ArXiv: Cryptography and Security*.

Wang, Y., Luo, F., Zhang, R., Tong, Z., & Qiao, Y. (2019). Distributed meter data aggregation framework based on Blockchain and homomorphic encryption. *IET Cyber-Physical Systems*, *4*(1), 30–37. https://doi.org/10.1049/iet-cps.2018.5054

Wang, A., Qin, B., Hu, J., & Xiao, F. (2020). Preserving transaction privacy in bitcoin. *Future Generation Computer Systems*, *107*, 793–804. https://doi.org/10.1016/j.future.2017.08.026

Wijaya, D. A., Liu, J. K., Steinfeld, R., Sun, S., & Huang, X. (2016). Anonymizing Bitcoin Transaction. *Lecture Notes in Computer Science*, 271–283. https://doi.org/10.1007/978-3-319-49151-6_19

Wilcox, Z. (2017). Ethereum Adoption of zk-SNARK Technology. Zcash blog. https://z.cash/blog/ethereum-snarks.html. Accessed: 20-Nov-2017

Williams, A. (2018). Consolidating Multiple Ledgers with Blockchain: A Single Digital Ledger for the Government of Canada Accounts. Blockchain Research Institute.

Wood, G. (2014). Ethereum: a secure decentralised generalised transaction ledger. http://paper.gavwood.com. 2014

Wootters, W. K., & Zurek, W. H. (1982). A single quantum cannot be cloned. *Nature*, *299*(5886), 802–803. https://doi.org/10.1038/299802a0

Wu, L., Hu, Y., Zhou, Y., Wang, H., Luo, X., Wang, Z., Zhang, F., & Ren, K. (2021). Towards Understanding and Demystifying Bitcoin Mixing Services. *Cornell University - ArXiv*. https://doi.org/10.1145/3442381.3449880

Wu, Q., Zhou, X., Qin, B., Hu, J., Liu, J., & Ding, Y. (2017). Secure joint Bitcoin trading with partially blind fuzzy signatures. *Soft Computing*. https://doi.org/10.1007/s00500-015-1997-6

Xiao, Z., & Xiao, Y. (2013). Security and Privacy in Cloud Computing. *IEEE Communications Surveys and Tutorials*, *15*(2), 843–859. https://doi.org/10.1109/surv.2012.060912.00182

Xu, L., Shah, N., Chen, L., Diallo, N., Gao, Z., Lu, Y., & Shi, W. (2017). Enabling the Sharing Economy. *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*. https://doi.org/10.1145/3055518.3055527

Yang, M., Margheri, A., Hu, R., & Sassone, V. (2018). Differentially Private Data Sharing in a Cloud Federation with Blockchain. *IEEE Cloud Computing*, *5*(6), 69–79. https://doi.org/10.1109/mcc.2018.064181122

Yao, A. C. (1982). Protocols for secure computations. *Foundations of Computer Science*, 160–164. https://doi.org/10.1109/sfcs.1982.88

Yi, X., & Lam, K. (2019). A New Blind ECDSA Scheme for Bitcoin Transaction Anonymity. *Computer and Communications Security*. https://doi.org/10.1145/3321705.3329816

Yi, X., Paulet, R., & Bertino, E. (2014). Homomorphic Encryption and Applications. *SpringerBriefs in Computer Science*. https://doi.org/10.1007/978-3-319-12229-8

Zaghloul, E., Li, T., Mutka, M. W., & Ren, J. (2020). Bitcoin and Blockchain: Security and Privacy. *IEEE Internet of Things Journal*, *7*(10), 10288–10313. https://doi.org/10.1109/jiot.2020.3004273

Zhang, Y., Long, Y., Liu, Z., Liu, Z., & Gu, D. (2019). Z-Channel: Scalable and efficient scheme in Zerocash. *Computers & Security*, *86*, 112–131. https://doi.org/10.1016/j.cose.2019.05.012

Zhao, C., & Guan, Y. (2015). A GRAPH-BASED INVESTIGATION OF BITCOIN TRANSACTIONS. *Springer International Publishing EBooks*, 79–95. https://doi.org/10.1007/978-3-319-24123-4_5

Zhao, J., Jung, T., Wang, Y., & Li, X. (2014). Achieving differential privacy of data disclosure in the smart grid. *International Conference on Computer Communications*. https://doi.org/10.1109/infocom.2014.6847974

Zhao, Z., Wang, J., Shi, K., & Zhang, H. (2022). Improving Address Clustering in Bitcoin by Proposing Heuristics. *IEEE Transactions on Network and Service Management (Early Access).* https://doi.org/10.1109/TNSM.2022.3186466

Zheng, Z., Wang, T., Bashir, A. K., Alazab, M., Mumtaz, S., & Wang, X. (2021). A Decentralized Mechanism Based on Differential Privacy for Privacy-Preserving Computation in Smart Grid. *IEEE Transactions on Computers*, *71*(11), 2915–2926. https://doi.org/10.1109/tc.2021.3130402

Zhu, L., Zheng, B., Shen, M., Gao, F., Li, H., & Shi, K. (2020). Data Security and Privacy in Bitcoin System: A Survey. *Journal of Computer Science and Technology*, *35*(4), 843–862. https://doi.org/10.1007/s11390-020-9638-7

Zhu, T., & Yu, P. S. (2019). Applying Differential Privacy Mechanism in Artificial Intelligence. *International Conference on Distributed Computing Systems*. https://doi.org/10.1109/icdcs.2019.00159

Ziegeldorf, J. H., Grossmann, F., Henze, M., Inden, N., & Wehrle, K. (2015). CoinParty. *Conference on Data and Application Security and Privacy*. https://doi.org/10.1145/2699026.2699100

Ziegeldorf, J. H., Matzutt, R., Henze, M., Grossmann, F., & Wehrle, K. (2018). Secure and anonymous decentralized Bitcoin mixing. *Future Generation Computer Systems*, *80*, 448–466. https://doi.org/10.1016/j.future.2016.05.018

Zyskind, G., Nathan, O., & Pentland, A. (2018). Enigma: Decentralized Computation Platform with Guaranteed Privacy. *The MIT Press EBooks*. https://doi.org/10.7551/mitpress/11636.003.0018