P2P & Blockchain

Survey on Mixing Techniques

Bitcoin Background

In 2008, **Satoshi Nakamoto** published the Bitcoin white paper, describing it as a **peer-to-peer (P2P) electronic cash system**. Bitcoin users connect directly with each other over a P2P network and exchange digital money (bitcoins), which are linked to cryptographic addresses. These bitcoins can be spent by using the **private key** linked to that address.

Bitcoin is made up of a chain of blocks. Each block is identified by a block header and refers to the block before it (called the **parent block**). This creates a **chain** going back to the very first block — the **genesis block**.

All transactions are stored in a way that is **distributed, permanent, and verifiable**. The blockchain **cannot be changed or deleted**.

Address

Bitcoin uses **asymmetric cryptography**, where users have a **public key** and a **private key**. The **address** is a hash (short digital version) of the public key. To spend bitcoins, the user must sign the transaction with their private key.

There are also **script hash** addresses, which let users send transactions to a more complex script instead of a regular address. Bitcoin uses the **Elliptic Curve Digital Signature Algorithm (ECDSA)** to create digital signatures.

Transaction

A Bitcoin **transaction** is the process of sending coins from one address (input) to another address (output). An **unspent transaction output (UTXO)** is used as the

input for a new transaction. The output is the recipient's address. Any leftover amount goes to a **new change address**, which belongs to the sender.

A transaction includes:

- transaction ID,
- version,
- inputs,
- outputs,
- and **nLockTime** — which sets the earliest time when the transaction can be added to a block.

Each input connects to a previous transaction's output. To prevent **double-spending**, Bitcoin keeps a list of UTXOs. Once an output is used, it is removed from the list.

Transaction Fee

To avoid spam or flooding attacks, Bitcoin charges a **transaction fee**. This fee goes to the miners. It's calculated as the **difference between the total input value and the total output value**. Larger transactions usually need **higher fees** to be confirmed quickly.

Transaction Scripts

Bitcoin uses a simple, stack-based scripting language called **Script**. The commands (called **opcodes**) usually start with "OP_". Scripts define the **rules for spending bitcoins**.

Most Bitcoin transactions use the **Pay-to-Public-Key-Hash (P2PKH)** format. Some others use more advanced scripts like:

- **Pay-to-Script-Hash (P2SH)**, or
- **Multisig** (which needs multiple signatures to spend coins).

Timelock Transaction

A **timelock transaction** prevents the coins from being spent until a certain time. This is useful for refunds. The **nLockTime** field in the transaction sets when the transaction becomes valid.

Hashlock Transaction

A **hashlock transaction** can only be spent by providing a secret value (called the **pre-image**) that matches a hash included in the transaction.

Many transactions can use the same hash, but they are only activated if someone reveals the correct pre-image. When one transaction is used, the pre-image becomes public, and then **all related transactions** can be unlocked.

To stop others from using the same pre-image to take the coins, **hashlock transactions** are usually locked by both the **signature** and the **pre-image**.

Hash Time Locked Contracts (HTLC)

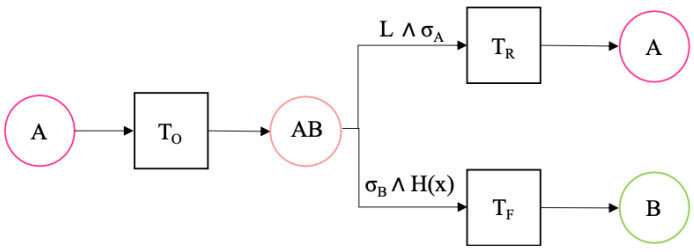
HTLCs are special scripts that use **both a hashlock and a timelock**.

- The output is locked by a hash.
- If the recipient doesn't unlock it within a certain time, the coins go back to the sender.

Example (Figure 1):

Bob can complete the transaction by providing the **secret (x)** and his **signature**.

If he doesn't, Alice can get her coins back using a **refund transaction** after the **locktime**.



σ_A: Alice's signature. σ_B: Bob's signature. L: Lock time.
T_O: Offer transaction. T_R: Refund transaction. T_F: Fulfill transaction.

Figure 1: Hash time locked contracts (HTLC)

2.1. Bitcoin Privacy Threats

Since all Bitcoin transactions are publicly visible on the blockchain, this creates **privacy problems** for users. Research has shown that people can use **heuristics** (rules of thumb) along with extra information from websites,

forums, or online stores to group transactions together and **figure out who owns what**.

2.2. De-anonymization in Bitcoin

There is some confusion about what exactly **“anonymity”** means in blockchain. One definition says:

> “A person is anonymous if you can't tell who they are among a group of people (called the anonymity set)” [58].

Bitcoin is **not fully anonymous**. Many studies [59, 52, 36, 22, 43] have shown that it is possible to **link Bitcoin addresses to real people**. Below are some of the most common techniques used for this.

Common Input Ownership

If a transaction has **multiple inputs**, each input needs to be signed with its own signature. Usually, these inputs are assumed to belong to **one person**, since it’s uncommon for many users to cooperate to create one transaction. This is called the **common input ownership heuristic** — it guesses that all inputs in a transaction are from the same user. According to [36], this method can correctly identify about **69% of wallet addresses**.

Detecting Change Addresses

When someone sends bitcoins, the total input amount is often more than the output amount. So the leftover coins are sent back to the sender through a **new address**, called a **change address**. This new address is usually controlled by the **same person** who made the transaction.

To detect change addresses, researchers use several clues (heuristics), such as:

- **Address reuse**: If an output address was used before, it might be a change address.
- **Script types**: If all inputs use the same script type, the output with the same script might be a change address. etc

Transaction Graphs

A **transaction graph** shows how bitcoins move between users.

Each **address** is a node in the graph, and each **transaction** is an edge (a link between addresses). Because each input is connected to an earlier output, you can trace **connections between transactions**.

Also, since most transactions use **change addresses** (controlled by the sender), many addresses can be grouped as belonging to one user. If someone interacts with a service (like an exchange) that knows their real identity, their address can be linked to them. Then, all related transactions can be linked as well. As mentioned in [39], even if users create a new address every time, this **does not fully protect privacy**. Also, since the sender knows the recipient's address, they can **track the recipient's future transactions** and see who else they deal with.

Linking Similar Addresses / Address Reuse

If a person **reuses** the same address in multiple transactions, those addresses can be **linked together** and assumed to belong to the same person.

Side-Channel Attacks

Side-channel attacks use indirect information to uncover identities, such as:

Time correlation: Linking transactions that happened close in time.

Amount correlation: Matching transactions with the same amounts.

Network information: Tracking IP addresses or network timing to guess where the transaction came from.

Implementation in practice. Most of the techniques mentioned before have not been used widely in real life, or there is a big delay between when the protocol is created and when it is actually used. Table 2 shows which techniques have been used in practice. Most of these are centralized mixing websites. According to [54], FairExchange transactions cannot be found on the blockchain, and most stealth address applications go back to the time of Darkwallet. Dumplings [56] shows that the number of CoinJoin transactions has increased in the last two years (since 2018). It is important to know that many CoinJoin transactions happen because of multiple mixing rounds to get better privacy. Joinmarket, Wasabi, and Samourai are wallets that use CoinJoin. Joinmarket works with a taker-maker model: the taker asks to create a CoinJoin transaction and makers join by paying fees. In this way, only the taker’s privacy is protected [34]. Wasabi uses Chaumian CoinJoin where participants register their inputs and sign the outputs blindly with the coordinator to create the CoinJoin. Samourai uses Whirlpool, which has special

pools where users mix their coins with others. SharedCoin was a CoinJoin service by Blockchain.info, but Blockchain.info could find connections between inputs and outputs. Darkwallet, which offered CoinJoin and stealth addresses, stopped working—likely for legal reasons. In 2020, BTCpay added PayJoin so merchants can accept PayJoin transactions in their stores. Currently, Wasabi, Samourai, Joinmarket, and Bluewallet support PayJoin.

Creating PayJoin transactions between users was done earlier in Joinmarket and Samourai wallets (called Stowaway). Shufflepuff [72] is an early version on Github last updated in 2016, and Nxt [42] has been active since block 621,000 (March 9, 2020) on the main network. At the time of writing, atomic swap techniques have no commercial use yet. Recently [6] proposed a new CoinSwap/PaySwap wallet design. There are some alpha versions of TumbleBit on Github (NTumbleBit and Breeze), but they are not commercial yet.

Future research. Future studies can focus on three areas: usability, law enforcement, and how practical these techniques are.

- **Usability.** Some important questions are: How much do users know about extra or built-in privacy techniques and their real use? I. Do they trust third-party privacy services? II. Do users prefer using add-on techniques from wallets and services or built-in privacy coins that offer stronger privacy? Are users okay with paying extra fees and waiting longer to get better privacy? I. Is there a difference between privacy-aware and privacy-unaware users in paying for privacy? Which privacy features do users want most (like preventing address reuse, hiding amounts, hiding sender, hiding sender and receiver, sending directly to the receiver, or not interacting with others)? Can users understand what to do with current privacy wallets and how to use them? Do privacy wallets increase the number of private transactions?
- **Law enforcement.** Although privacy techniques are sometimes used in the dark web and for illegal activities, some users use them to avoid serious problems caused by deanonymization on the blockchain. There is no research that classifies the destination addresses of CoinJoin transactions, which are popular privacy techniques in wallets like JoinMarket, Wasabi, and Samourai. Even though it is hard to find the exact recipient of CoinJoin transactions, classifying these addresses using verified data can help understand how CoinJoin is used in the Bitcoin blockchain. A good research question is: Can

we categorize the destinations of CoinJoin transactions to find out how many are used for illegal activities? Privacy and law enforcement are always in tension in cryptocurrencies. Finding a balance to protect most users' privacy while stopping criminals is still an open challenge. [45] suggested a model where law enforcement can work with participants in CoinJoin transactions to catch criminals, which could be a good start.

- **Practicality.** Accepting PayJoin in the market can give good privacy because it breaks the common input ownership rule. But these transactions should be made so they cannot be easily identified as PayJoin. Also, unnecessary input detection and wallet fingerprinting need to be considered in the protocol's design. More research is needed to see how well these methods can detect PayJoin transactions. Research can also look at CoinJoin transactions with unequal amounts. Right now, equal-size CoinJoin transactions can be identified, which can cause problems, like some services refusing their outputs. The Knapsack method [49] and Wabisabi [23] could help improve how hard it is to identify these types of transactions in the future.

6. Conclusion

This study aimed to review and evaluate mixing techniques used in Bitcoin. It compared many different proposals based on several criteria. These proposals offer different levels of privacy, security, and efficiency. Strong privacy usually affects how efficient, scalable, and easy to use the technique is. Among atomic swap techniques, New CoinSwap and earlier versions meet most of the criteria, but they need more transactions, which means more time and higher fees. CoinJoin-based techniques are the most commonly used in practice. However, these techniques face problems like transactions being easy to recognize because of equal-sized outputs and attacks that can disrupt the system (DoS attacks). The new PayJoin method, which is based on CoinJoin, can solve the problem of easy recognition and improve anonymity. One big advantage of CoinJoin-based methods is that they need fewer transactions to work, which makes them cheaper. But most of these techniques cannot provide a large group of users for anonymity (called anonymity set) or the ability to deny involvement (plausible deniability). Confidential transactions, like those in ValueShuffle, can fix

this problem and make CoinJoin transactions harder to tell apart. Mixing techniques often need a minimum number of transactions to hide the links between senders and receivers. While having more transactions can improve privacy, it also costs more in fees. Even if mixing fees are small, the extra transaction fees might stop users from using these techniques. Our results show that except for centralized mixers and threshold-based techniques, most methods are resistant to theft. Although the main goal of these privacy techniques is to protect users from criminals and attackers, these methods can also be used for illegal activities. Therefore, new ways are needed to tell apart transactions used for illegal actions from normal privacy transactions (like financial privacy).

Presentation Slides & Words:

Slide 1: Intro

Today, I'll walk you through key **techniques for enhancing privacy** in Bitcoin transactions, known as **mixing techniques**.

Slide 2: UTXO

Let's begin with the foundation:

When Bitcoin's creator Satoshi Nakamoto launched the protocol in 2009, it became the first operational digital currency system to implement the UTXO model. Blockchains like Bitcoin use UTXO model. Where is blockchains like Ethereum use account-based models.

UTXO stand for Unspent Transaction Output. It is an amount of Bitcoin that your wallet controls. You can think of UTXOs as loose change leftover from previous bitcoin transactions. The funds are considered "unspent" because they are freely available for you to send to someone or move to another wallet. They are called "transaction outputs" because they were created from previous transactions.

Your wallet can have multiple addresses and each address can hold multiple UTXOs. And the total sum of these UTXOs is you Bitcoin balance. Each UTXO is like a separate piece of Bitcoin you can spend.

UTXOs use public key cryptography to prove and transfer ownership. Specifically, the recipient's public key is included in the UTXO, so only the person who has the matching private key can spend it. To spend the UTXO, a valid digital signature connected to the public key must be provided.

To find the balance of an address, you scan the network for all UTXOs linked to that address and add up all the unspent outputs.

Additional:

A **UTXO (Unspent Transaction Output)** is a certain amount of cryptocurrency that a sender has approved and that the recipient can spend.

Each transaction input points to a UTXO from a previous transaction, and each transaction output creates a new UTXO. These new UTXOs are stored in the user's wallet and can be spent later. When a UTXO is used as an input in a new transaction, it is spent and no longer counts as a UTXO.

Slide 3: UTXO & Cash

UTXOs is similar to cash transactions in a physical wallet, which helps you protect your privacy by preventing others from seeing you entire balance. If your UTXO is larger than the amount that you are spending, the leftover balance is returned to you as change in a new UTXO. When a UTXO is spent, it is considered "consumed" and is technically removed from circulation.

Another key comparison between physical cash transactions and UTXOs is that both have to be **spent in full** and **cannot be subdivided**. If you have a 5 bitcoin UTXO and want to send someone 1 BTC, you would have to send the entire UTXO worth 5 bitcoin and receive a new UTXO in return worth 4 BTC minus any fees. This part of the UTXO system is how Bitcoin solves the **double-spend problem**. When a person attempts to spend the same UTXO twice, the two transactions end up in a mempool — a sort of waiting room for pending transactions. They remain in here until successful miners that win the proof-of-work competition bundle them into new blocks.

This is another important thing to understand about bitcoin transactions. When a transaction is made, bitcoin isn't digitally moved from one account to another. Instead, they are unlocked, reassigned to a new owner and then locked back up again.

The main differences between the physical bills analogy and the UTXO model is that bitcoin and other UTXO-based cryptoassets aren't bound by set amounts, i.e. $5, $10, $15, etc. Any amount of bitcoin (up to eight decimal places) can be an unspent transaction output.

For instance, you may have 0.0003847 BTC leftover from a transaction. This amount would become a new UTXO which must be spent in full if used and cannot be divided into smaller amounts.

Additional:

UTXOs under the hood: At a more technical level, there are four main parts to a transaction:

Version: This informs network nodes what version of client software is being used. Different versions follow different rules for verifying transaction data.

Locktime: This is the amount of time that determines how fast a transaction is added to the blockchain. This input dictates what the earliest possible time is for the transaction to be processed by mining nodes.

Input: Information pointing to the source of funds or previous transaction where the UTXO was produced. The input also contains something called the "unlocking script".

Output: Information regarding the value being transferred, the wallet where ownership of funds is being reassigned to and new UTXOs formed. The output also contains a "locking script".

Slide 4: UTXO – Example

You have a wallet which controls 2 Addresses (Address A, Address B). Address A has a single UTXO worth 0.5 BTC. While Address B has two UTXOs: one worth 0.2 BTC, other 0.3 BTC. If you wanna send someone 0.4 BTC, you can choose which Address to use, this helps protect your privacy because recipient sees only the balance associated with the address used, rather than your full wallet balance. Let's say we use Address A: Your wallet sends 0.4 BTC to the recipient and returns the remaining 0.1 BTC as a change, to a newly generated address your wallet controls. Minus of course a small amount of fees paid to the miners who process the transaction.

Now Address A is empty, and we have a new change address Address C, with a little less than 0.1 BTC. This makes it harder for others to track your activity on the blockchain. If you need to spend more than single UTXO, will combine multiple UTXOs until total covers you payment. This can be done containing UTXOs in a single address or multiple addresses. Any extra is then returned to you as a change.

Unlike cash UTXOs aren't restricted to set denominations. They can be any size, this means you can also pay from Address b by combining two UTXOs, 0.2 and 0.3 BTC. The change will still go to a new address, leaving Address B empty after.

However, because you spending multiple UTXOs transaction would be more expensive in terms of fees.

Slide 5: UTXO Management

UTXO management or miss-management affects you on chain in two ket ways: **Privacy and Fees.**

Privacy-wise, each UTXO is tied to a specific address. Reusing addresses or combining the wrong UTXOs can make it easy for others to trace your transactions. Simply, if someone knows your identity is tied to one address, they can uncover how much BTC you own, if you aren't careful with how you spend your UTXOs.

When it comes to fees, the size of Bitcoin transaction depends on the number of UTXOs involved. More UTXOs means a larger transaction in terms of data, which means higher fees, especially when the network is busy. If your transaction includes many small UTXOs, you may end up paying significant fees.

UTXO Consolidation is the process of combining multiple smaller UTXOs into one larger UTXO by sending a transaction yo yourself. Consolidation makes you wallet more efficient and can reduce transaction fees for future payments. However consolidating UTXOs can impact your privacy, because you are linking UTXOs that may have come from different addresses. It's crucial to choose carefully which UTXOs to consolidate.

If the person you paid 0.4 BTC knows your identity, and you merge all you UTXOs into a single address, they could see your entire balance. UTXO consolidation can be step to avoid high fees later. Its usually best to consolidate UTXOs when

network fees are low. So its always tradeoff between your privacy and fees that you pay.

Slide 6: Mixer

*A **cryptocurrency mixer** is a specialized service designed to increase the privacy and anonymity of blockchain transactions.*

Bitcoin is pseudonymous, not fully anonymous. Unlike traditional financial transactions, which are private by default, most cryptocurrencies such as Bitcoin and Ether operate on public blockchains. This means every transaction is permanently recorded and accessible to anyone, making it possible for blockchain analysts or malicious actors to trace the flow of funds between wallets.

A crypto mixer's primary function is to break the link between the sender's wallet and the recipient's wallet. It does so by pooling together coins from many users and then redistributing them in a way that makes it difficult to track which coins went where.

Think of it like a digital version of shuffling cards in a deck. After mixing, your cryptocurrency is returned to you or a recipient's address, but it's "cleaned" of any direct transaction history. This privacy-enhancing feature is why some people rely on mixers, especially those seeking to keep their financial activities confidential in an open-ledger world.

Slide 7: Mixers Work

Here's a typical workflow of how a cryptocurrency mixer operates:

- **Deposit:** You send your Bitcoin to the mixer's wallet address. Multiple users do the same, creating a large pool of coins.
- **Mixing/shuffling:** The mixer's system pools and shuffles these coins together, breaking any visible connection between deposited and withdrawn funds.
- **Redistribution:** After mixing, the service sends back an equivalent amount of coins to your specified address, but these aren't the same coins you deposited. They come from the pooled coins of all participants.

- **Fees:** The mixer usually deducts a small fee, generally ranging from 1% to 3%, to cover operational costs.

This process effectively disrupts blockchain analysis, making it extremely difficult for anyone to trace the coins back to their original owners.

Slide 8: Types of Cryptocurrency Mixers

Not all mixers are created equal. They can broadly be divided into two categories: centralized and decentralized mixers.

Decentralized mixers use blockchain technology and smart contracts to automate the mixing process without a trusted third party. They rely on cryptographic methods such as zero-knowledge proofs to mix coins in a trustless environment. Users pool their coins into a smart contract, which then redistributes coins in a way that ensures privacy.

Slide 9: Centralized Mixers - Mixing Websites

Centralized mixers are the most common and operate similarly to traditional services. You send your coins to a company or entity that controls the mixing process, and then they send back “clean” coins after mixing. These services are relatively easy to use, often providing a simple user interface. However, centralized mixers require you to trust the service operator with your coins, at least temporarily.

Mixing websites are websites that run centralized mixers. So they are a subset of centralized mixers.

Slide 10: Mixing Websites Example

If **Alice, Bob, and Carol** want to send coins to **A′, B′, and C′**, they all send the **same amount** of coins to the mixer. The mixer then **shuffles** the coins and sends them to the recipients. After this, it's hard to tell which sender sent money to which recipient.

Usually, users have to **fill a form** on the mixing website with:

- the recipient’s address,

- a preferred time delay (to increase anonymity),
- and agree to the **mixing fee** and **transaction fee**.

The mixer then gives the user a **new address** to send their coins to.

Issues:

- The mixer can **steal the coins** and not send anything to the recipient.
- The mixer can **record the user's data** and **link sender to recipient**.
- Since users can send **different amounts**, it becomes easier to track them on the blockchain.
- To get **good privacy**, users must often wait longer — which is inconvenient.
- **Mixing fees** are often **high** (from 1% to 5% of the total amount).
- Also, [15] pointed out that if the mixer always takes a **fixed fee percentage**, it can leak **information** about the transaction — which can weaken privacy.

Slide 11: Atomic Swaps

Each cryptocurrency is supported by a blockchain, designed only to accept transactions in specific tokens. For example, the Bitcoin and Ethereum blockchains each have a native token that cannot be transferred to the other. You first need to convert them to fiat currency and then buy the other using other cryptocurrencies and exchanges to get the one you want. Depending on the cryptocurrency, this can take several trades. Atomic swaps allow you to exchange tokens from different blockchains in one trade.

Atomic Swaps are simply peer to peer crypto trades across different blockchains without using centralized exchange or liquidity pool, meaning that you could for example exchange your BTC for ETH with a stranger directly. Atomic swaps are generally initiated by users and executed by a smart contract. The smart contract can be programmed in many ways, but most tend to lock up the tokens being swapped or burn them, then issue the new tokens to the transferees.

What Are the Benefits of Atomic Swaps?

When two entities want to trade tokens, they can use an atomic swap to ensure no third parties are involved. This technique is faster and generally cheaper than

going through exchanges or other token swap service providers.

Is an Atomic Swap Anonymous?

In most cases, the only publicly available information is the token amounts and the users' public addresses. However, if other information has been made available, these addresses can be traced back to their owners, so they are realistically pseudonymous.

Atomic swaps executed using type of smart contract technology known as **Hash Time locked Contracts or HTLC** to automate exchange of tokens.

HTLC stands for Hashed TimeLock Contract, and is a contract that enforces a transaction to be completed within a certain time frame, or the funds are refunded.

Think of a HTLC as virtual lock box that only opens with two unique keys, that is the **HashLock** and the **TimeLock** key.

HashLock key basically opens the virtual lock box so that the trading parties can get their exchanged assets after the swap is complete.

TimeLock key as its name suggests is a time-based feature that acts as an additional security measure such that in case of unfulfulled swap conditions, both parties get their funds back after the set time window closes. This is because atomic swaps are designed to either happen exactly as planned or not happen at all.

Slide 12: Atomic Swap Example

- Alice agrees to swap 10 X tokens with Bob for 10 Y tokens. They create an HTLC that will expire in one hour.
- Alice creates a contract address and deposits her 10 X tokens to it. This generates a private key that only Alice has access to. Alice creates a cryptographic hash of the private key and sends it to Bob.
- Bob uses the hash to verify that Alice has deposited 10 X tokens to the contract address. Bob cannot access the funds because he only has the hash, not the actual private key.

- Bob uses the hash to generate a new contract address where he deposits his 10 Y tokens. Now both parties have deposited their funds to the contract.
- Because Bob created the address using the hash of Alice's private key, Alice is able to get the 10 Y tokens Bob deposited. She does this and, in the process, reveals the private key to Bob. If Bob does not complete the transaction before the timelock expires, the tokens got by Alice will revert to Bob.
- Now Bob uses the private key to withdraw the 10 X tokens and finalize the transaction.
- With the swap completed within one hour, the contract cannot revert and Alice has successfully swapped her 10 X tokens with Bob for his 10 Y tokens.

Additional:

Disadvantages of Atomic Swaps

DEXs and centralized exchanges remain in high demand due to some of the cons of using atomic swaps to trade tokenized assets.

- **Ease-of-use**—Each counterparty must agree on the amount and price of the transaction, the length of the timelock, exchange data, and hashes, and wait for transactions to be processed. This may become a timely and complex process that can be especially difficult for beginners.
- **Privacy concerns**—Atomic swaps take multiple blocks to be completed, which can alert malicious actors, giving them time to track addresses and target traders.
- **Compatibility**—You cannot perform atomic swaps across all blockchain networks. Each blockchain must use the same hashing algorithm for atomic swaps to work.

Advantages of Atomic Swaps

Atomic swaps offer traders benefits that are unavailable from some other solutions.

- **Reduce counterparty risks**—There is no need to entrust funds to a centralized third party to facilitate the transaction. Traders maintain complete control over their assets.

- **Deeper liquidity**—Making assets tradeable across different blockchain networks makes those assets more liquid.
- **Direct asset-to-asset swaps**—Users can buy crypto assets directly without needing to first swap to a highly liquid stablecoin and make multiple transactions. Users can trade any token they like in a decentralized environment.
- **Guaranteed outcome**—Traders have guarantees that their contract will execute as described or they will receive their funds back.
- **Lower costs**—Peer-to-peer swaps can incur lower fees than relying on a third-party administrator.

Slide 13: Fair Exchange

FairExchange was proposed by **Barber et al. [4]**. It allows **two users** (like Alice and Bob) to swap their coins directly.

A fair exchange protocol consists of three types of transactions:

- **Commitment Transaction:** This is where someone locks up their money, showing they're serious about the deal.
- **Refund Transaction:** If the deal doesn't go through (for example, if the other person disappears), this transaction lets the first person get their money back after some time.
- **Claim Transaction:** This lets a person take the other person's locked money—**but only if** they share a secret. The first person to make a claim must publish a special code (a secret), and that code helps the second person make their claim too.

Alice sends coins to **Bob's recipient address**, and Bob sends coins to **Alice's recipient address**.

Steps of the protocol:

1. **Alice and Bob** each create **two key pairs** (for different parts of the process).
2. They each create **secret values** — called **a(i)** and **b(i)**.

3. They use a method called **cut-and-choose** to exchange **hashes** of the secrets:
    - H(a(i)+b(i))
    - H(b(i))

      These are added to the **offer transactions**.

Steps:

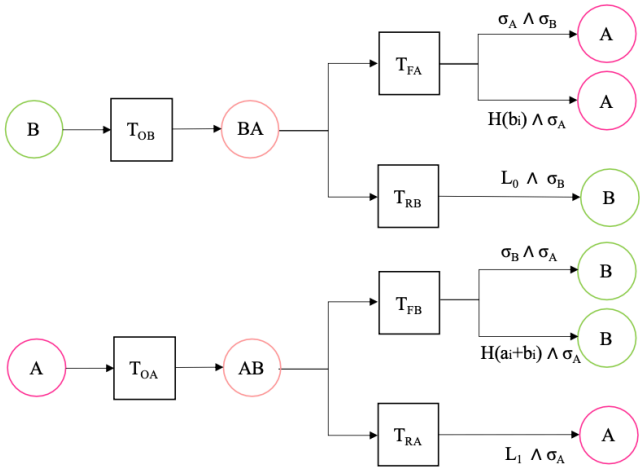
Commitment Phase:

- **Bob** creates a offer transaction **T(OB)** that can be used in two ways T(FA):
    - Either **both Alice and Bob sign it**, or
    - Alice signs it and provides the secret **b(i)**.
- **Bob** also creates a partial **refund transaction, with locktime set (T(RB))** in case something goes wrong and he needs his coins back.
    - Bob send this partial **(T(RB))** waits for Alice to sign it
    - Alice checks Locktime signs it and then sends the completed refund transaction back to Bob.
    - Bob checks the signed **(T(RB))** and then **publishes** both **T(OB)** and **T(RB)**.

- **Alice** does the same on her side:
    - She creates **T(OA)**, which can be redeemed by either:
        - **Both Bob and Alice's signatures**, or
        - **Bob's signature and the value a(i)+b(i)**

Claim Phase:

- **Bob** completes Alice's transaction by revealing his **signature** and **a(i)+b(i)**.
- **Alice** subtracts her own secret (a(i)) from the total to find **b(i)**, and then uses it to complete Bob's transaction.



σ_A: Alice's signature. σ_B: Bob's signature. T_{RA} & T_{RB}: Refund transactions. T_{OA} & T_{OB}: Offer transactions. T_{FA} & T_{FB}: Fulfill transactions. L_0 & L_1: Lock times.

Figure 2: FairExchange

Issues:

- The protocol uses **four transactions**, which means **extra cost and longer time**.
- The transactions have **special formats** that stand out and can be recognized on the blockchain.
- Both final (fulfill) transactions **contain the same values**, so it's easier to track them.
- Only **two people** are involved in the swap, so to get strong anonymity, you need **many repeated swaps**.
- Using **timelocks** (a delay condition) also **limits the size of the anonymity set**.

Additional:

Anonymity:

The anonymity of Fair Exchange is based on two key things. First, Alice and Bob actually swap their coins. If they use different key pairs for the claim transactions, there is no direct link between the transactions on the public blockchain. Second, the transactions cannot be directly linked using the preimages *a* and *b*, because the link is hidden inside the hash and the total of the values.

However, the system uses two special types of output scripts. Even though these scripts are not directly linked to each other, they clearly show that the Fair Exchange method was used when looking at the blockchain. Also, the claimed transaction outputs will likely have about the same value. Because of this, the anonymity is limited to only those transactions that have similar spending conditions and values.

Since the refund transactions have a locktime, the coins must be claimed shortly after the funding transactions are added to the blockchain. This further limits the possible size of the anonymity group. So, Fair Exchange can only provide good anonymity if it is used by many people who are swapping coins of similar value at around the same time.

Slide 14: CoinJoin

CoinJoin was introduced by **Gregory Maxwell** in 2013 [50]. In Bitcoin, each input in a transaction must be signed by its **owner's private key**. Using this idea, multiple users can **work together** to create **one big transaction** with all their inputs.

CoinJoin Basics: Group multiple user's inputs into ONE transaction with multiple EQUAL outputs. This makes it difficult to trace who is sending where, since outputs can't be distinguished.

CoinJoin is a type of anonymous transaction, primarily used by Bitcoin mixers or wallets, that helps Bitcoin users maintain their privacy while sending Bitcoin. CoinJoin is not a single-party transaction, which means that the transaction does not go from one sender to one receiver. Instead, **it is a multiparty transaction at the end of which it is unclear who owns which coin.** This ambiguity is secured by matching the size of inputs or outputs. While this may seem a bit complicated, a good example can help understand this logic.

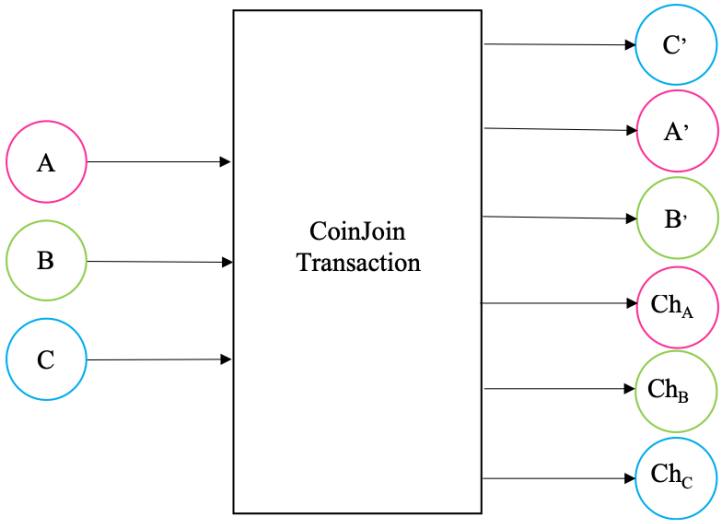
By doing this:

- They **break the common input ownership heuristic** (because not all inputs belong to one user).

- If all users send **similar amounts**, it becomes harder to tell which input is linked to which output.

To create the transaction:

- Each user provides:
  - their **input address**,
  - **output address**, and
  - **change address**,
- and then **signs** their part of the transaction.
- One user **collects all signatures** and sends the **full transaction** to the Bitcoin network.



Additional: While searching gregory Maxwell information about Coinjoin, I saw his reddits in the internet. And he said even he is not planning to further work on this technique. And on his website there is no, publishes related to his CoinJoin work.

Slide 15: CoinJoin Issues

Issues:

- There may still be some **internal links** between users.

- Bitcoin has **limits on transaction size**, so only a small number of users can join — meaning a **small anonymity set**.
- Large groups **increase risk** of Sybil or DoS attacks and require more communication between users.
- Users must spend the **same coin amount**, which makes it **less practical** in real life.
- Outputs with **identical values** can still make change addresses **easy to identify**.
- This reduces **plausible deniability** (users can't deny their participation).
- The protocol doesn't clearly explain how **users are selected** for the mixing process [7].

Slide 16: Threshold Signatures

https://youtu.be/4DFfZovCBB0?si=YR78PkyKNL08XrPL

https://medium.com/@yehudalindell/threshold-signature-schemes-mpc-for-cryptocurrency-protection-48ea2453ecb5

https://academy.binance.com/en/articles/threshold-signatures-explained This one i more clear i think

Threshold signature techniques use a shared (joint) signature system. A transaction can only be signed and completed when a **minimum number of participants (the threshold)** agree and provide their signatures.

ECDSA key & signature, SMC, PRNG, MultiSig

Slide 17: CoinParty

CoinParty uses a decentralized method for Bitcoin mixing. Instead of trusting a single third party, it uses a group of **mixing peers** that work together and act like a trusted third party by using **Secure Multi-Party Computation (SMC)**. This setup allows users to mix their Bitcoins securely and anonymously. In this paper, we call the privacy peers who run the SMC protocol **mixing peers**.

Figure shows an example of how the CoinParty protocol works with three users. CoinParty works in **three main phases**:

(1) the **commitment phase** (explained in Section 4.1),

(2) the **shuffling phase** (explained in Section 4.2), and

(3) the **transaction phase** (explained in Section 4.3).

There is also a **fourth phase**, called the **error and reversion protocol** (Section 4.4), which is used when something goes wrong or when someone behaves dishonestly during the earlier phases.

Step-by-step process:

1. **Escrow (Commitment) Phase**:
    - Mixing peers work together to generate a set of **escrow addresses** (T1, T2, T3) using **threshold ECDSA**.
    - These addresses are **jointly controlled** by the mixing peers.
    - Each input user (the one mixing coins) gets one of these addresses and sends their coins to it.
    - These coins can **only be spent** if the **majority** of mixing peers sign the transaction.
2. **Shuffling Phase**:
    - Input peers (user) **encrypt** their **output addresses** using **layered encryption** with the public keys of all mixing peers.
    - They also attach a **hash** of their output address to help verify the final result.
    - Each mixing peer:
        - Decrypts the message,
        - **Shuffles** the addresses,
        - Sends them to the next peer.
    - The **last mixing peer** does the final shuffle and **broadcasts the list** to all other mixing peers.

- All mixing peers check the final list and use a **pseudo-random number generator (PRNG)** to randomize the order — this prevents the last peer from controlling the final result.

3. **Final (Transaction) Phase**:
   - Mixing peers use the **threshold ECDSA** to **sign the transactions** and send the coins to the shuffled output addresses.
   - The **private keys** needed to sign are **shared among the peers**, and only a group of them can complete the transaction.

Error Phase:

When an error is detected during the commitment or shuffling phase, the mixing peers transfer all funds from the escrow addresses back to the input addresses. The necessary steps are the same as in the transaction phase. As we analyze these transaction are guaranteed to succeed even in the presence of malicious adversaries such that no funds can be stolen, diverted, or lost.

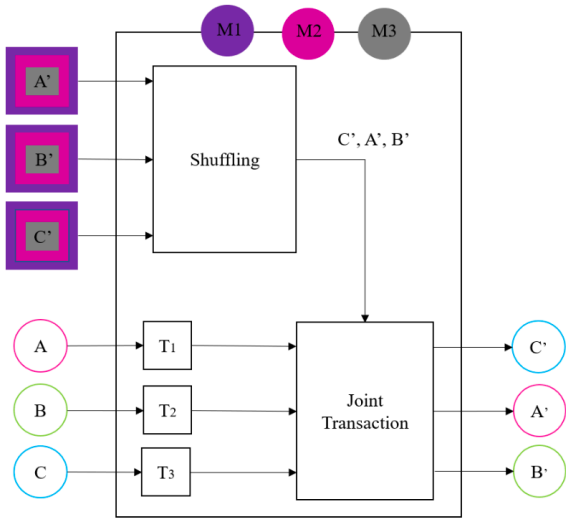


Figure 6: CoinParty, inspired by [83]

Issues:

- The protocol is secure **only if at least 2/3 of the mixing peers are honest**. This is hard to guarantee in a peer-to-peer network where users don't have **strong identities** [63].

- Because of this, it is vulnerable to **theft** or **DoS attacks** [20].
- Mixing peers don't benefit directly from the transaction, so they might **leave the process halfway** when coins are already locked in escrow.
- To keep them engaged, they can be **paid with mixing fees**, but that **increases the total fee** paid by users [40].
- Every transaction requires:
  - **Generating a threshold ECDSA key**, and
  - **Creating a threshold ECDSA signature**,
  - Which causes **high computation cost**.

Additional:

Escrow Phase:

We show how to create escrow addresses **Ti** and their matching key pairs in a distributed way, so that no central trusted party is needed:

(C1) Using **Pseudo-Random Secret Sharing (PRSS)**, each mixing peer **Mj** gets a share **[di]j** of a secret random value **di**, which acts as the private key.

(C2) Each **Mj** locally calculates their part of the public key **Qi**:

[Qi]j = [di]j · G, where **G** is the base point of the elliptic curve (for Bitcoin, this is **secp256k1**).

(C3) Each mixing peer **Mj** sends her public key share **[Qi]j** to the other mixing peers.

(C4) Each **Mj** receives the shares from the other peers and rebuilds the full public key:

Qi = Σ λj [Qi]j = Σ λj [di]j · G = di · G,

where **λj** is the **Lagrange coefficient** used to combine the shares correctly (evaluated at **x = 0**).

(C5) Each **Mj** creates the Bitcoin address **Ti** from the public key **Qi**, following Bitcoin's standard procedure.

The mixing peers prepare a list of escrow addresses ahead of time and give a unique escrow address **Ti** to each input peer **i** when needed.

If an input peer sees two different escrow addresses (**Ti ≠ Ti'**), for example from a malicious mixing peer trying to steal funds, the peer stops the protocol immediately and alerts the other mixing peers about the cheating.

Otherwise, the input peer sends the required funds **v** from their address **Ii** to the escrow address **Ti** — this means they make a transaction:

Ii →v Ti.

The mixing peers then wait for the usual six confirmations before the transaction is considered final by the Bitcoin network.

Transaction phase:

the mixing peers create transactions of the form **Ti →v Oπ(i)**. To do this, they need to generate one **ECDSA signature** for each transaction.

Because the **private key di** for an escrow address **Ti** is **shared** among the mixing peers, the normal ECDSA algorithm can't be used. Instead, the mixing peers must work together to sign the transaction. They use a **threshold version** of the ECDSA algorithm from **Ibrahim et al. [17]**, and follow these steps to create and sign a Bitcoin transaction:

(T1) The mixing peers use **PRSS** [13] to generate shares **[k]i** of a random secret number **k**. Then they use the reciprocal protocol from [17] to calculate $\mathbf{[k^{-1}]i}$, the inverse of **k**.

(T2) Each mixing peer **Mj** calculates the hash value **e = SHA-1(Ti →v Oπ(i))**, following Bitcoin's rules.

(T3) Using the elliptic curve generator **G**, each peer computes their part of **kG** as **[kG]j = [k]j · G** and shares this with the others. After collecting all the shares, they combine them to reconstruct the full **kG**.

(T4) From **kG = (x, y)**, they calculate **R = x mod n**. Then, the peers calculate the second part of the signature as $\mathbf{[S] = [k^{-1}] \cdot (e + [d] \cdot R)}$ and reconstruct the full **S**.

(T5) The mixing peers now have the complete **ECDSA signature (R, S)**. They use it to create and broadcast the transaction **Ti →v Oπ(i)**.

To make this process faster and more reliable than in [17], we introduce some improvements:

- The mixing peers **precompute the first part of the signature R**, when they also precompute the escrow addresses **Ti**.
- This means they can precompute **[k]**, **[k^{-1}]** (T1), and then **kG** (T3).
- They also **split the second part S** (from T4) into two parts:

  [S] = [k^{-1}] · e + [k^{-1}] · [d] · R.
- This allows the term **[k^{-1}] · [d] · R** to be **precomputed in advance**.

So, when it's time to actually sign the transaction, the mixing peers just need to compute the hash **e** (T2), do the simple multiplication **[k^{-1}] · e** (T4), and then reconstruct **[S]**.

This makes signing **faster** and more **resistant to failures**, like if a peer stops responding.

Slide 18: Summary Table – Privacy

Privacy Criteria

- **Anonymity set**: This means the **number of users** taking part in a mixing transaction. A **larger set** makes it **harder to tell who sent what**.

  Most techniques (except some CoinJoin-based ones) can offer a large anonymity set and can blend in with regular blockchain transactions.

  In CoinJoin-based techniques, the anonymity set is often limited by the size of the transaction, and it’s hard to coordinate many users to join one transaction. Bigger groups mean more risk of DoS and Sybil attacks, and more communication is needed.
- **Unlinkability**: Given **two transactions**, if it's **impossible or very hard** to tell if both went to the **same recipient**, we say they are unlinkable. This means that even if someone receives coins in different transactions, **no one can link them to one person** [69].
- **Untraceability**: Given a transaction, **all senders look equally likely**. You **can’t tell** which input address actually sent the coins [69].

All techniques try to improve **untraceability** — making it hard to tell who sent the coins.

Some techniques still have **internal (pertial) traceability**, meaning that within the group of users participating, it might be possible to **match inputs and outputs**. Even if traceability **exists among participants**, these techniques still protect against **outside observers** or **blockchain analysts** [54].

- **Payment value privacy**: The **amount of the transaction** is hidden or protected from being analyzed using blockchain data.

  Among the techniques, **ValueShuffle** suggests using **Confidential Transactions (CT)**, which hides the coin values. But CT would require a **soft fork** (an upgrade) to be added to Bitcoin. If CT is added to Bitcoin:

  - All mixing techniques could **benefit** from it.
  - Users wouldn't need to use **fixed coin amounts**, making mixing **more flexible** and **user-friendly**.

  Study [57] compared CT in **TumbleBit** and **CoinJoin**:

  - It found that CT can **reduce costs** for **large-value transactions**.
  - But it **increases fees** for **small-value transactions**.
  - On average, CT would **increase Bitcoin fees 9 times**.

Slide 19: Summary Table – Security

One of the most important goals in payment systems is to **make sure the coins are not stolen or lost**. This is especially important in **blockchain**, where there's **no easy way to get coins back** once they're lost

Security Criteria

- **Theft resistance**: The coins **cannot be stolen** while the protocol is running.

  Mixing websites do **not** offer theft protection.

  Threshold signature techniques also don't fully prevent theft. They require a **majority of honest users**, which is **hard to guarantee** in peer-to-peer

networks.

Atomic swap and **CoinJoin-based techniques** can provide better **theft protection**, as designed in their protocols.

- **DoS resistance** (Denial of Service): A user **can't just stop** in the middle of the process and prevent the transaction from completing. *(This is only checked in decentralized systems where users work together to create the transaction.)*

  Most **CoinJoin** and **threshold signature techniques** are **vulnerable**, because they depend on users behaving properly. Some methods try to fix this by: **Detecting and removing malicious users**, **Restarting the protocol**, or **Locking up the attacker's coins (UTXO)**.

  Centralized mixers and **atomic swaps** are generally **DoS-resistant**, since no participant can **stop the whole process** once it starts.

- **Sybil resistance**: An attacker **cannot join the protocol multiple times** using **fake identities** to try to find out who is receiving the coins.

  Most techniques defend against Sybil attacks by **requiring users to pay a fee upfront**.

Slide 20: Summary Table – Efficiency

Efficiency Criteria

- **No interaction with input users**: The technique doesn't require users to **communicate with each other** to build the transaction.
- **Bitcoin compatible**: The technique works with the **current Bitcoin system** and doesn't break compatibility with features like **blockchain pruning** (removing old, unused data).
- **Direct send to the recipient**: Coins can be sent **directly to the final recipient** — not first to a temporary address controlled by the sender and then forwarded. If not, users must first receive coins to their own address and then forward them — which adds **extra steps and fees**.

  This issue happens in **CoinJoin** and **threshold signature techniques**.

Using **stealth addresses** could solve this problem by letting users send directly without revealing identity.

- **Number of transactions**: The **fewer the transactions**, the better. We look at how many transactions are needed to finish the mixing. In practice, **extra transaction fees** (on top of the mixing fee) can be a big **barrier** for users.

  Even in **CoinJoin-based techniques**, each user has to pay:

  At least **one extra fee** for mixing,

  And another for **sending coins to the final address**.

  Also, **one round of CoinJoin** may not give enough anonymity, so users need to **mix multiple times**, which increases cost.

  Atomic swap techniques also need **four transactions** and at least **two blocks**, which leads to **higher fees and longer delays**.

- **Minimum mixing time (Blocks)**: This is the **minimum number of blocks** needed to complete the protocol. *(One Bitcoin block takes about 10 minutes.)*

Slide 21: Conclusion

In summary, mixing techniques enhance **Bitcoin privacy**, but no method is perfect. These techniques offer different levels of privacy, security, and efficiency. Strong privacy usually affects how efficient, scalable, and easy to use the technique is.

Mixing techniques often need a minimum number of transactions to hide the links between senders and receivers. While having more transactions can improve privacy, it also costs more in fees.

CoinJoin-based techniques are the most commonly used in practice. Except for centralized mixers and threshold-based techniques, most methods are resistant to theft.

Although the main goal of these privacy techniques is to protect users from criminals and attackers, these methods can also be used for illegal activities.

Slide 22: References

These are the sources I used. If you have any questions or would like to discuss further, feel free to email me at: **z.zadagerey@studenti.unipi.it**. Thank you for your attention!

MAIN PDF