The Impact of Taproot and Schnorr on Address Clustering Analysis of Bitcoin Transactions

09 March 2020

Authors: *Alexi Anania* *(alexi@synenesi.com),* *Ken Hodler* *(ken@citizenhodler.com)*
(Special thanks to a contributor involved in cybercrime law enforcement, who opted to remain anonymous).

Abstract

Bitcoin Core developers have two new technological improvements planned for the Bitcoin Core client in 2020: Taproot and Schnorr. These upgrades were formally suggested in BIPs 340, 341, 342 and provide solutions to improve the privacy and anonymity of bitcoin transactions. Thus far, a technique called address clustering has been successfully used by law enforcement organizations and other forensic investigators to trace pseudonymous bitcoin transactions to a real world identity. In this paper, we examine the implications of Taproot and Schnorr on clustering analysis, to conclude that the aforementioned BIPs are anticipated to have minor impacts. Thus address clustering analysis will continue to be a useful heuristic for forensic investigators, once Taproot and Schnorr are fully implemented.

Anonymity of Bitcoin and clustering

Bitcoin is often pointed out for its properties of anonymity,[1] however, it publicly publishes the amount, sending address(es) and receiving address(es) of every transaction in the blockchain. Bitcoin addresses are alphanumeric[2] in the same way that bank account numbers are. Put differently, it is a pseudonymous routing address used instead of the owner's legal name.

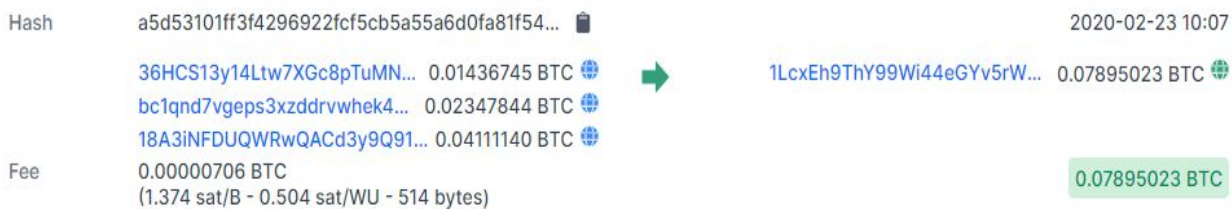

Figure 1: Example of a transaction with multiple inputs

There are three primary types of addresses in Bitcoin, as seen in Figure 1 and 2. The analysis changes based on the type of address.

- P2PKH type starts with the number 1: 18A3iNFDUQWRwQACd3y9Q914hL8kbgdCAp.
- Bech32 type starts with bc1: bc1qnd7vgeps3xzddrvwhek448uq938995mulk9dz3. P2PKH and Bech32 are different encoding schemes for public keys. In cases where the public key is known, they can be converted to a common format for analysis.
- P2SH type starts with the number 3: 36HCS13y14Ltw7XGc8pTuMNX7E3pQrebNk. A P2SH address might have additional addresses embedded in the script. So with P2SH transactions, the root address and the addresses contained in the scripts can be analyzed together.

On the topic of privacy in the Bitcoin whitepaper, Satoshi Namamoto wrote that a new public key should be used for every transaction in order to maintain elevated privacy, however

[1] Richard Holden and Anup Malani. Why the I.R.S. Fears Bitcoin. 22 January, 2018 . Accessed from: https://www.nytimes.com/2018/01/22/opinion/irs-bitcoin-fear.html
[2] Bitcoin address formats and prefixes. Accessed from: https://allprivatekeys.com/bitcoin-address-format.

he also wrote that linkage is possible with multi-input transactions as the owner of all those inputs must be the same:[3] the private key for every input-address is needed to sign the transaction.

The multi-input linkage heuristic seen in figure 1, in addition with several others,[4] exposes the potential to form (and trace) clusters of addresses belonging to the same owner. Another heuristic often used is based on Bitcoin Script whereby pubkeys are revealed for a multi-signature transaction, connecting the various parties holding those keys.

Regulation prevents cryptocurrency exchanges (operating as legitimate businesses) from using CoinJoins.[5] As a result, exchanges have unique deposit addresses for each user and take advantage of multiple inputs, hence saving on transaction fees. This in combination with Open-Source Intelligence (OSINT)[6] of interactions with exchanges, creates the required exposure for big clusters to become identifiable. Blockchain analysis companies, like Chainalysis, Elliptic and CipherTrace,[7] maintain a database with identified clusters allowing their clients to fulfill a better AML/CTF (source of funds) and KYC (identity). Law enforcement uses the same software to track stolen coins, money laundering, ransomware payments and darknet traders.[8] [9]

---

[3] Satoshi Nakamoto. *Bitcoin: A peer-to-peer electronic cash system*. Consulted, 1:2012, 2008. Accessed from: https://bitcoin.org/bitcoin.pdf.

[4] CryptoQuant Team, Introduction to Bitcoin Heuristics. (30 July, 2019), Accessed from: https://medium.com/cryptoquant/introduction-to-bitcoin-heuristics-487c298fb95b.

[5] Anonymiser according to the Financial Action Task Force: Documents - Financial Action Task Force. Accessed from: https://www.fatf-gafi.org/publications/fatfrecommendations/documents/public-statement-virtual-assets.html. FinCEN. Application of FinCEN's Regulations to Certain Business Models Involving Convertible Virtual Currencies, page 19. 09 May 2019 Accessed from: https://www.fincen.gov/sites/default/files/2019-05/FinCEN%20Guidance%20CVC%20FINAL%20508.pdf.

[6] Wikipedia Article: Open Source Intelligence. Accessed from https://en.wikipedia.org/wiki/Open-source_intelligence. Last edited on 7 February, 2020

[7] Sedgwick, Kai, A Forensic Analysis of Blockchain Surveillance Companies. 05 March 2019. From Bitcoin.com 's new page. Access from https://news.bitcoin.com/a-forensic-analysis-of-blockchain-surveillance-companies/.

[8] Europol and Chainalysis Reinforce Their Cooperation in The Fight Against Cybercrime. 19 Feb 2016. Europol News Webpage. Accessed from https://www.europol.europa.eu/newsroom/news/europol-and-chainalysis-reinforce-their-cooperation-in-fight-against-cybercrime.

[9] Inside Chainalysis' Multimillion-Dollar Relationship With the US Government. Accessed from: https://www.coindesk.com/inside-chainalysis-multimillion-dollar-relationship-with-the-us-government.

Example of clustering: multi-input heuristic for exchanges

Using a full Bitcoin Core node with txindex and RPC enabled, all blocks and transactions can be queried. This is done by querying the blockhash for every block number. Subsequently, by using this blockhash, the block of transactions is retrieved. Then for every transaction hash in the retrieved block, the raw transaction is queried and decoded. Since the inputs are provided as a transaction hash,[10] the raw transaction has to be queried and decoded to find any corresponding addresses. Thereafter, the input addresses are matched against a database of addresses in (already identified) clusters and appended to a discovered cluster.

Once a new address is found, the full blockchain is scanned to determine whether that address previously performed a multi-input transaction. Relying on only a node is suboptimal. The API of block explorers like Blockchain.info can be used to retrieve all transactions of the particular address.

In our example we use a combination of the Blockchain.info API for history look-ups and a Bitcoin Core node for monitoring new transactions. Starting with our exchange deposit addresses as well as certain addresses found on Twitter and BitcoinTalk, multi-input clustering was initiated.

To illustrate the effectiveness of address clustering, a user posted several transaction IDs on BitcoinTalk that were withdrawals from Paxful.com.[11] The first transaction had 102 inputs.[12] Given those 102 addresses belonging to Paxful.com, the multi-input clustering algorithm looks for historic transaction inputs from these addresses. As a result 5807 new addresses were found. By visiting 2550 of these addresses, we found over 310 thousand addresses belonging to Paxful.com. Many of these addresses are used multiple times, indicating that Paxful allows for multiple deposits on the same address. Interestingly, we noticed a specific address has even been used in 3776 transactions and referred 8921 new addresses to the algorithm.[13]

By using this single heuristic we were ultimately able to identify more than 36 million

[10] Unknown, Daniel. What's a UTXO? A Guide To Unspent Transaction Output (UTXO). 26 July 2018. Komodo Website: Education page. Accessed from https://komodoplatform.com/whats-utxo/.
[11] https://bitcointalk.org/index.php?topic=1694776.0.
[12] https://www.blockchain.com/btc/tx/90abcc4892f36d14753eb7c2ce1f1a21ee569b0a00203693f894253255e538f4.
[13] https://www.blockchain.com/nl/btc/address/36ztavSsPjeBUcYgHVaqRbiTmQL7xUG97r.

addresses from 33 exchanges. Professional services such as Chainalysis use more heuristics than the aforementioned example. As a result, they are able to identify many more addresses by actually interacting with services themselves.

To counter the heuristic of address clustering analysis, CoinJoins have been used where multiple parties combine their transactions in order to create one transaction with multiple inputs and outputs of the same amount.[14] CoinJoins are an effective way of disrupting an address clustering analysis because it associates multiple addresses that are not part of the same wallet. There are two downsides for a CoinJoin though. Namely that it adds to the cost of a typical transaction because it requires additional transaction fees and it must be coordinated offchain by its participants. Some automation of the coordination process is possible, but isn't widely used due to additional cost of sending a transaction.

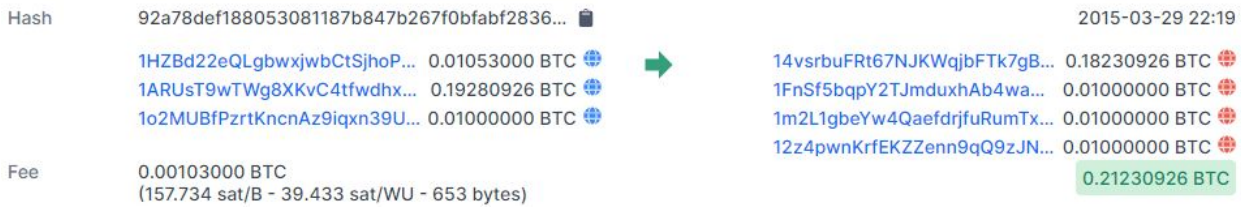

Figure 2: Example of a CoinJoin Transaction

The effect of clustering on privacy

There are a total of over 460 million Bitcoin addresses created thus far,[15] evidently the number of identified addresses is quite significant. The vast majority of Bitcoin users buy and sell in this way (at similar exchanges) and therefore expose their address to further blockchain analysis. In the event that a criminal or bad actor undertakes forbidden or illicit activity, law enforcement is able to subpoena the exchange for the user's identity.

Combined with other heuristics, multiple addresses belonging to the same owner can be identified. This allows blockchain analysis companies to find the amount that an identified

[14] CoinJoin. (Last edited on 30 June 2019). In Bitcoin Wiki. Accessed from: https://en.bitcoin.it/wiki/CoinJoin.
[15] Chainalysis Team. Mapping the Universe of Bitcoin's 460 Million Addresses. 19 December 2018. Chainalysis Blog. Accessed from https://blog.chainalysis.com/reports/bitcoin-addresses.

individual invested in cryptocurrency as well as any blockchain related services they interacted with.

Taproot for scalability and privacy

Bitcoin Script is a small stack-based program with simple yes/no type outcomes.[16] With just a few simple instructions embedded in each transaction, Script defines the cryptographic signature algorithms on the blockchain and how any coin in a transaction can be spent. Bitcoin Script is very limited in what it can do by design. It was specifically designed so that complex programs are impossible to implement.

Two of Bitcoin's most significant and complicated criticisms are a lack of scalability and the lack of privacy that arise from complex smart contracts using Bitcoin Script. Taproot was initially conceived in 2018 to address these problems as we explore passim. Proposed by former Blockstream CTO, Gregory Maxwell, a respected Bitcoin Core contributor, it is anticipated as a major innovation.[17]

Taproot was made possible due to SegWit (Segregated Witness). SegWit introduced the Script versioning, an extension of the Bitcoin protocol, which is an intrinsic requirement in order to define or add these new opcodes. The Taproot BIP has several changes included in one proposal.[18] These changes include Merkle Abstract Syntax Tees (MAST), merging of pay-to-pubkey and pay-to-scripthash policies, disabling the OP_CHECKMULTISIG and OP_CHECKMULTISIGVERIFY opcodes, and a few other minor, mostly technical adjustments to make Bitcoin scripting more efficient. Most of these changes don't impact privacy or address clustering analysis. In the paper we focus on MAST.

Currently, P2SH requires that the entire script is revealed when signing a transaction. A complex script might have multiple possible execution paths, but only the path that is actually

[16] Script. Accessed from: https://en.bitcoin.it/wiki/Script.
[17] Maxwell, Gregory. Taproot: Privacy preserving switchable scripting. 23 Jan 2018. Accessed from: https://bitcoin-development.narkive.com/wyni1HaG/taproot-privacy-preserving-switchable-scripting.
[18] Wuille, Pieter; Nick, Jonas; Towns, Anthony. BIP-341, Design section. Bitcoin Improvement Proposals github repository. 19 Jan 2020; last updated: 25 Jan 2020. Accessed from: https://github.com/bitcoin/bips/blob/master/bip-0341.mediawiki#design.

executed by the signature is important. The unexecuted parts of the script make the transaction heavier (more bytes) and may reveal private details that can be used to cluster addresses.

A proposed solution, called MAST, uses Merkle trees to represent more complex scripts as simpler atomic units. Taproot is built directly on MAST. By using this compact data structure, all contract conditions are individually hashed, and when a condition is revealed to the blockchain it can be verified using the Merkle root hash and the Merkle path. Without compromising the privacy of any other conditions of the transactions, most Script code included in the Merkle tree remains hashed and hidden. Only the condition which was met is revealed.

As an example, assume we have the following timelocked recovery script where Bob is able to recover Alice's funds after 3 months:

```
OP_IF
      <Alice's Pubkey> OP_CHECKSIG
OP_ELSE
      "3 months from now" OP_CHECKSEQUENCEVERIFY OP_DROP
      <Bob's Pubkey> OP_CHECKSIG
OP_ENDIF
```

The script represents two scenarios: Scenario A is that Alice can spend her funds at any time. In scenario B, Bob can spend Alice's funds after 3 months have elapsed. With MAST, these two scenarios are broken into separate scripts:

Script for scenario A:

```
<Alice's Pubkey> OP_CHECKSIG
```

Script for scenario B:

```
"3 months from now" OP_CHECKSEQUENCEVERIFY OP_DROP
<Bob's Pubkey> OP_CHECKSIG
```

Next, a merkle tree is constructed:

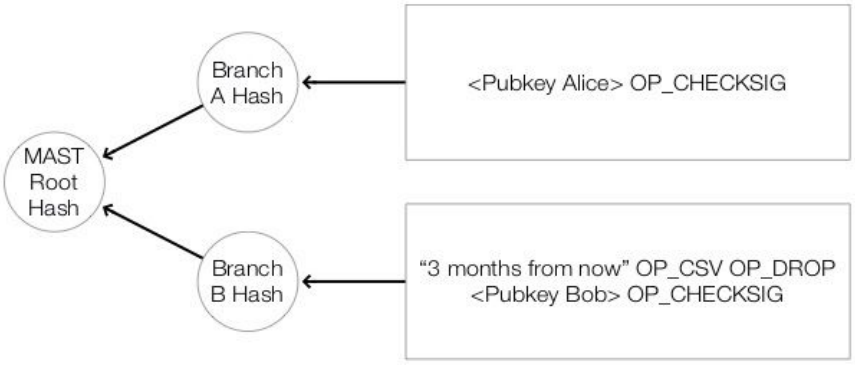

Figure 3: MAST Script Layout

The locking script added to the UTXO is a single sequence of bytes that represents an internal key (described below) and the MAST root hash. It is indistinguishable from a pubkey. When Alice executes scenario A, she needs to send an signature script:

```
<Alice's signature>
```

She also provides the merklized redeem script:

```
<MAST root hash> <Alice's Pubkey> OP_CHECKSIG <Branch B hash>
```

When Bob executes scenario B instead, the signature script is:

```
<Bob's signature>
```

And the merklized script is:

```
<MAST root hash> <Branch A hash>
"3 months from now" OP_CHECKSEQUENCEVERIFY OP_DROP
<Bob's Pubkey> OP_CHECKSIG
```

In both signing scenarios, a Merkle proof is provided that the executed code was part of the original Script while revealing only the details of the Script that are necessary for the scenario. Alice only provides the Script details for scenario A or Bob only for scenario B. This ensures that only the details of the executed scenario are publicly revealed on the blockchain. It should also be noted that not even the signers of the transaction need to know the full details of the script. They only need to know the scripts for the scenarios that they can participate in and the corresponding Merkle proofs.

Taproot also offers a way of hiding the existence of the script altogether. A public key can be specified that can be used to sign the transaction without invoking the script. This is called the 'internal key'. With a signature that corresponds to the internal key, the spending condition is satisfied and the script does not need to execute. It can be thought of as the default scenario or the scenario where everyone agrees. In the example above, the scenario where Alice can sign at any time would most likely be the default scenario and the script for scenario A can be replaced by an internal key. The remaining script is organized into a MAST with only one node (see figure 4).

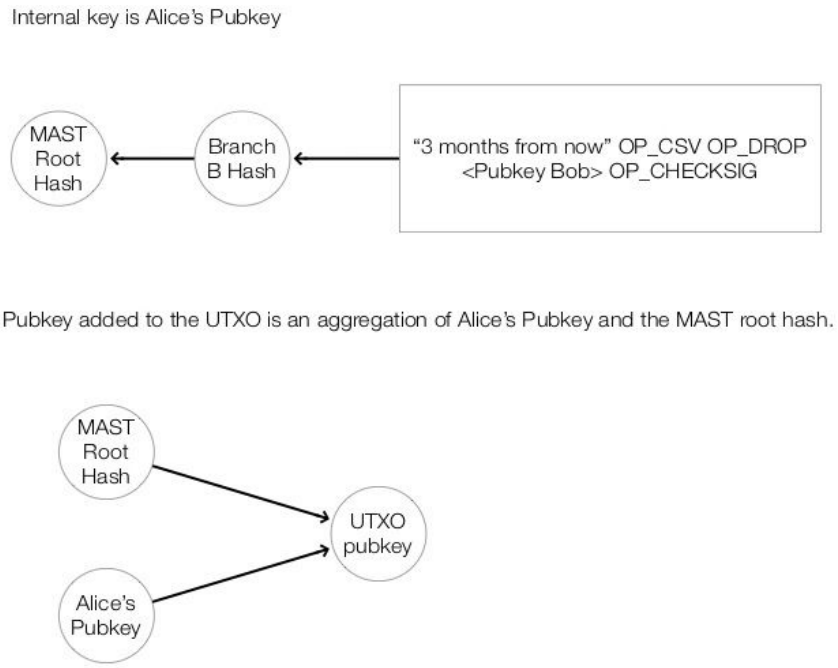

Figure 4: Taproot Script with an internal key

Alice is now able to sign the transaction by sending a signature and not specifying a redeem script. When Bob executes scenario B, he will provide a signature using his personal secret key and the following redeem script:

```
<MAST root hash>
"3 months from now" OP_CSV OP_DROP
<Bob's Pubkey> OP_CHECKSIG
```

There are ongoing lessons being learned from the security failures of public, transparent contracts wherein all states are visible and verifiable. Taproot expands the Bitcoin network's smart contract flexibility and offers a higher degree of privacy by making complex smart contracts indistinguishable from any regular transaction, in the case that the primary scenario is executed.[19]

Effects of Taproot on Cluster Analysis

MAST has only a minor impact on cluster analysis. Since it doesn't impact the ability of users to mix multiple UTXOs into a single transaction, cluster analysis will still be effective. There are two small but notable impacts:

1. Since MAST hides sections of the scripts that aren't executed, it also hides any pubkeys that those sections contain. An address clustering analysis that considers pubkeys embedded in scripts will be limited by the implementation of MAST.
2. MAST may enable a new clustering vector. The "Security" section of BIP340[20] describes a strategy where different wallets may have different implementation quirks that can be fingerprinted and used as a vector for clustering addresses. As an example, if we have a script that has 10 branches, one wallet might organize the branches as a complete tree and another wallet could organize them as a balanced tree.[21] By analyzing the Merkle tree that is partially revealed in the MAST redemption script, it is possible to identify which wallets may have signed the transaction. This 'fingerprint' can be used to associate transactions and the addresses they contain into address clusters.

[19] Van Wirdum, Aaron. Taproot Is Coming: What It Is, and How It Will Benefit Bitcoin. 24 Jan 2019. Bitcoin Magazine website. Accessed from: https://bitcoinmagazine.com/articles/taproot-coming-what-it-and-how-it-will-benefit-bitcoin.
[20] Wuille, Pieter; Nick, Jonas; Ruffing, Tim. BIP-340. Last updated: 23 Feb 2020. Bitcoin Improvement Proposals github repository. Accessed from https://github.com/bitcoin/bips/blob/master/bip-0340.mediawiki.
[21] Types of Binary Trees. Last updated 13 Feb 2020, Wikipedia. Accessed from: https://en.wikipedia.org/wiki/Binary_tree#Types_of_binary_trees

Schnorr signatures

Another proposed soft fork, by prominent blockchain developer and CEO of Blockstream Pieter Wuille, is a Bitcoin Improvement Proposal for Schnorr signatures.[22] Claus P. Schnorr first described the benefits of Schnorr signatures. These complement those of Taproot, namely: an efficiency of space, cheaper transactions, as well as privacy properties for scripting leading to enhanced security.

Presently, Bitcoin works with Elliptic Curve Digital Signature Algorithm (ECDSA). Schnorr signatures existed under patent at the time of Bitcoin's inception,[23] and even predate ECDSA. It is believed that had the patent not existed, Satoshi Nakamoto might have initially chosen Schnorr signatures instead, as opposed to ECDSA.[24] Schnorr signatures have advantages in that they are extremely easy to verify meaning fast transaction times and also multi-signature (multisig) capabilities. Today the patent has expired and Schnorr remains an active area of research. Many innovations can be done on Bitcoin, built on top of Schnorr.

Schnorr algorithms allow for certain types of aggregation, whereby most of the computation is done before actually transmitting to the network. In this way, it decreases the transaction latency and saves space. SegWit makes it a lot easier to implement Schnorr in Bitcoin. Schnorr signatures achieve incremental results by allowing almost any smart contract to enable the functionality for participants to agree on an outcome, therefore they can cooperate and use multisig transactions.

Bitcoin Core developers have been studying various multi-signature schemes which provide the groundwork for the current Schnorr signature proposal.[25] These schemes work by

---

[22] ibid.

[23] Schnorr, Claus P. Method for identifying subscribers and for generating and verifying electronic signatures in a data exchange system. Patent application US4995082, USPTO, Accessed from https://patents.google.com/patent/US4995082.

[24] Boudjemaa, Adam. The Future of Bitcoin: Schnorr Signatures, Key Aggregation & Interactive Aggregate Signatures. 7 Nov 2019. Hackernoon Blog. Accessed from: https://hackernoon.com/the-future-of-bitcoin-schnorr-signatures-key-aggregation-and-interactive-aggregate-signatures-ias-wbk36po.

[25] Maxwell, Gregory et. al. Simple Schnorr Multi-Signatures with Applications to Bitcoin. 20 May 2018. Accessed from: https://eprint.iacr.org/2018/068.pdf.

aggregating multiple signing keys into a single public key which acts as a destination address. When the group wants to spend the funds, each individual signer signs a spending transaction and the signatures are aggregated into one signature that corresponds to the group's public key (see figure 6).

Several aggregation schemes are possible, but the scheme that is likely to be favored is MuSig.[26] It allows aggregation of the public keys into a single public key. This saves space on-chain because all of the public keys are compressed into one. It also leads to large privacy gains: since it isn't possible to distinguish between a single key and a group of compressed keys, multi-signature transactions are indistinguishable from single signature transactions. Additionally, since the individual public keys are compressed, they are not exposed on-chain.

Effects of Schnorr Signatures on Address Cluster Analysis

Implementing Schnorr signatures will not impact the effectiveness of address cluster analysis. Like Taproot, Schnorr does not affect the ability of users to mix UTXOs paid to multiple addresses into a single transaction, which is the key heuristic used by address clustering analysis.

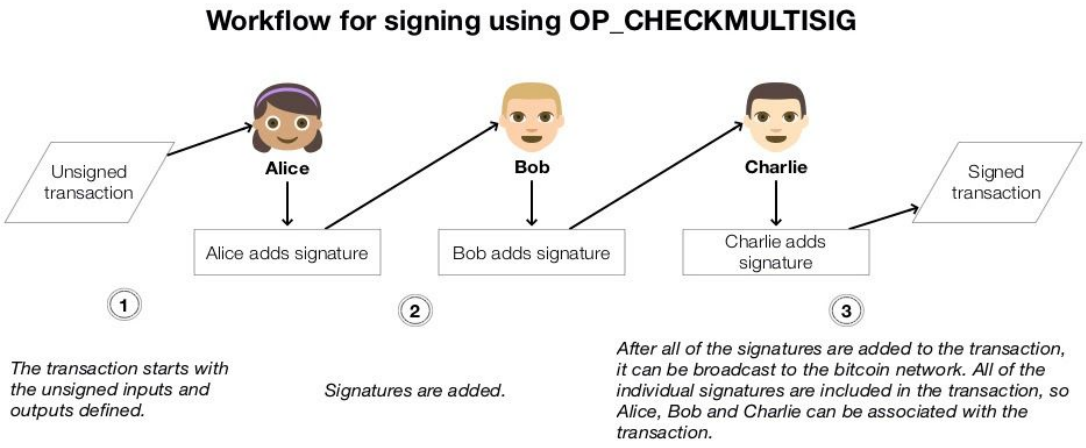

Figure 5: Workflow for signing using OP_CHECKMULTISIG

[26] ibid.

Schnorr offers a privacy improvement for users of multi signature transactions. Since the individual signatures are cryptographically aggregated to form a single published signature, they are not exposed for analysis (see figure 6). By contrast, the current mechanism of using OP_MULTISIG exposes the public keys of the signers and the potential signers of a transaction (see figure 5). Since Schnorr aggregates signatures, address cluster analysis is prevented from associating transactions that have overlapping signatories.

Similarly, another minor improvement to privacy comes from the fact that multisignature transactions will be indistinguishable from single signer transactions. Since the pubkeys in UTXOs and the signatures attached to them are each cryptographically aggregated into a single value in multisig transactions, it isn't possible to distinguish them from single signer transactions. This applies to both key signature spends and script based spends. This will impact address clustering analysis that uses the distinction between single signer and multisig transactions in their analysis.

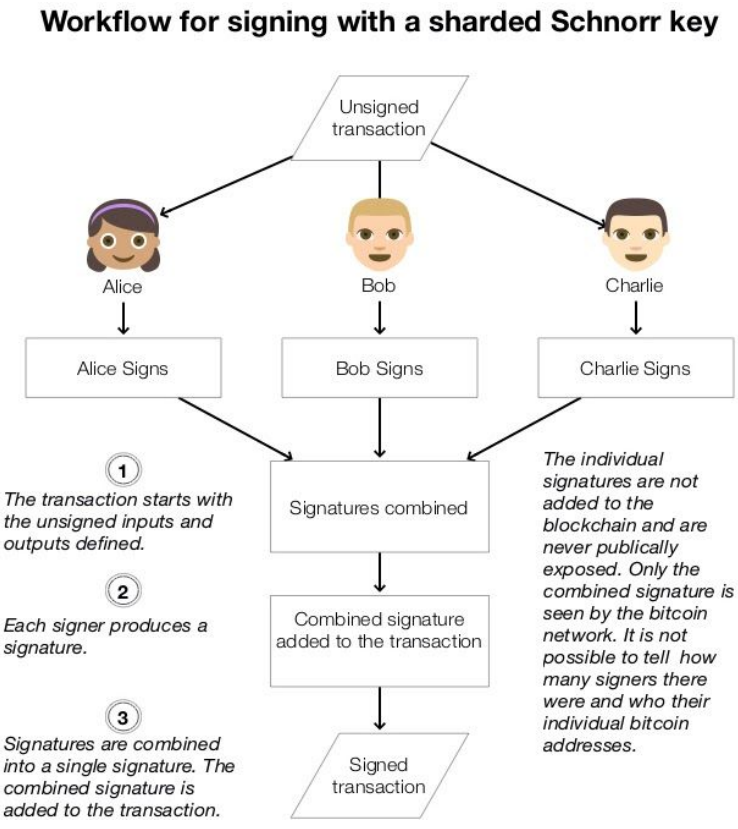

Figure 6: Workflow for signing with a sharded Schnorr key

When considering CoinJoins, these will be less expensive to do with Schnorr than with OP_MULTISIG. Since Schnorr aggregates the signatures, CoinJoins will take up less space in a block and have lower fees. This may lead to increased use among users in the Bitcoin community that place a higher value on privacy. The fees for CoinJoins will still be higher than a direct send, thus most users would possibly continue using direct transactions. It would come down to who is prepared to pay the premiums for privacy.

The heuristic for identifying CoinJoin transactions remains the same. A transaction that consists of more than 3 inputs from multiple address clusters and a number of outputs equal to the number of inputs can be treated as a CoinJoin transaction. Once a CoinJoin transaction is detected, the address clustering system can no longer associate the addresses from the transaction's outputs with the addresses from the inputs, which are part of known clusters.

It is probable that users who place a high value on privacy will build systems that allow them to easily use CoinJoins for every transaction sent. Address clustering would be much less effective if use of such a system becomes widespread. Schnorr doesn't improve privacy directly, but rather it encourages the more prolific use of CoinJoins.

Conclusion

Bitcoin transactions are exposed by different heuristics in order to effectively cluster addresses from the same owner. Schnorr and Taproot offer improvements in the privacy and efficiency of the Bitcoin blockchain. However, as we have shown, neither will substantially change the effectiveness of address clustering analysis. There are minor privacy improvements that prevent address clustering from considering addresses contained in more complex scripts and multi-signature transactions. There is also a notable case where Taproot reveals new information that enables a new clustering vector. In the case when the primary scenario of a complex script is executed, script transactions are indistinguishable from the more common single signature transactions. However, these don't change the underlying source of effectiveness of address clustering as it stands today, which is the inclusion of multiple UTXOs targeted at different addresses in a single transaction.