[Blockcore- Wallet]

Summer of Bitcoin 2024

API to parse and sign PSBT

Mentors: sondreb dangershony

Name and Contact Information

Name: Tirth Bhayani

Github username: tirth2004

Discord username: @tirth8290

Email: ttirth1234518@gmail.com

Alternative Mail: tirth.bhayani.cse22@itbhu.ac.in

University info

- **University Name:** Indian Institute of Technology (BHU) Varanasi
- **Major:** Computer science and Technology
- **Current year and expected graduation date:** 2nd year (sophomore). Expected graduation will be around 30th May 2026

Degree: Bachelor of Technology (4 Years)

LinkedIn: tirthbhayani1

Country: India

Time Zone: IST (UTC+05:30)

Table of Contents

Title

Implement API endpoints to enable **PSBT** support in blockcore wallet extension

Synopsis

Partially signed bitcoin transactions (or **PSBTs**) were introduced in **BIP 174** which were upgraded in **BIP 370**. It introduced a standard format to perform multisig transactions, coinjoin transactions and even offline transactions. Blockcore wallet does not support PSBT yet. I propose to implement it.

Project Plan

Theoretical Details:

Once people started using bitcoin for multiple purposes, the need for different types of transactions also arose. One of the important

types of transaction was **multi-sig**. When a token is held by multiple people at once, all or some of them must approve when any **UTXO** has to be spent. Until 2017, such types of transactions were implementation based. Then Andrew Chow released **BIP 174** for Partially Signed Bitcoin Transactions or PSBT. It introduced a proper structure to deal transactions involving multiple parties. The figure given below explains a general flow for PSBT.

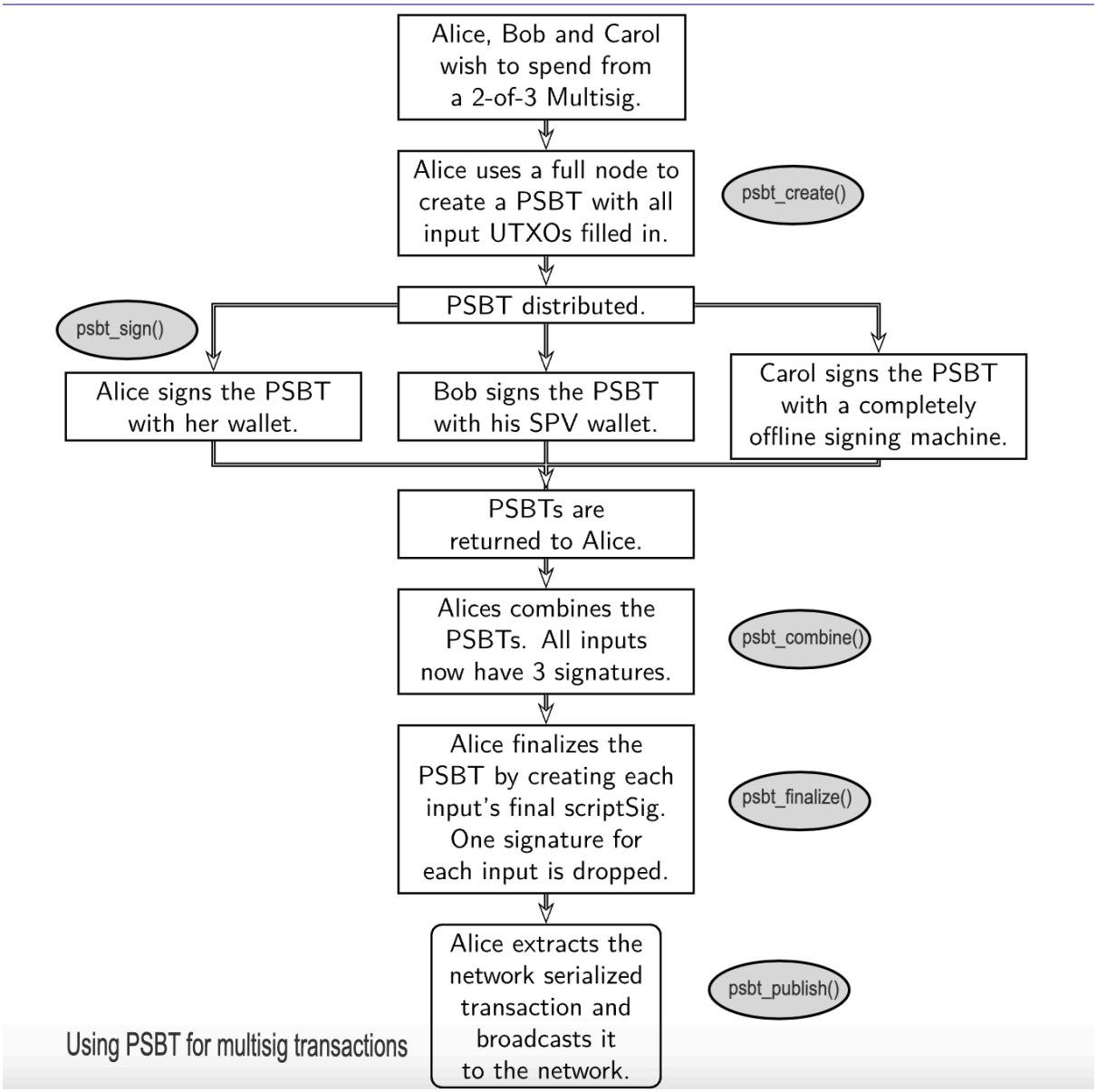

Fig-1 - Flow of PSBT

Initially a **creator** initiates a transaction by creating an empty PSBT. Then an **updater** collects input and output information for all the parties involved, searches for relevant UTXOs and adds all the data to the PSBT. This PSBT is then passed on to the **signers**. Signers sign the transactions with only the UTXOs provided in the raw PSBT. After every one signs, the **combiner** combines everyone's PSBT. **Finalizer** makes the final call to publish the PSBT on chain.

BIP 370 then proposed a change. `PSBT_GLOBAL_UNSIGNED_TX` which contained the entire transaction excluding the signatures was excluded from PSBT. As it was defined by the updater, once even a single signature is made, `PSBT_GLOBAL_UNSIGNED_TX` cannot be changed, as it will make the existing signatures invalid. Thus providing a way to add inputs and outputs at later stages and eliminating the need to send UTXO and input/output information to the creator beforehand. This made certain transactions such as **coinjoin** much easier. Below figure illustrates it.

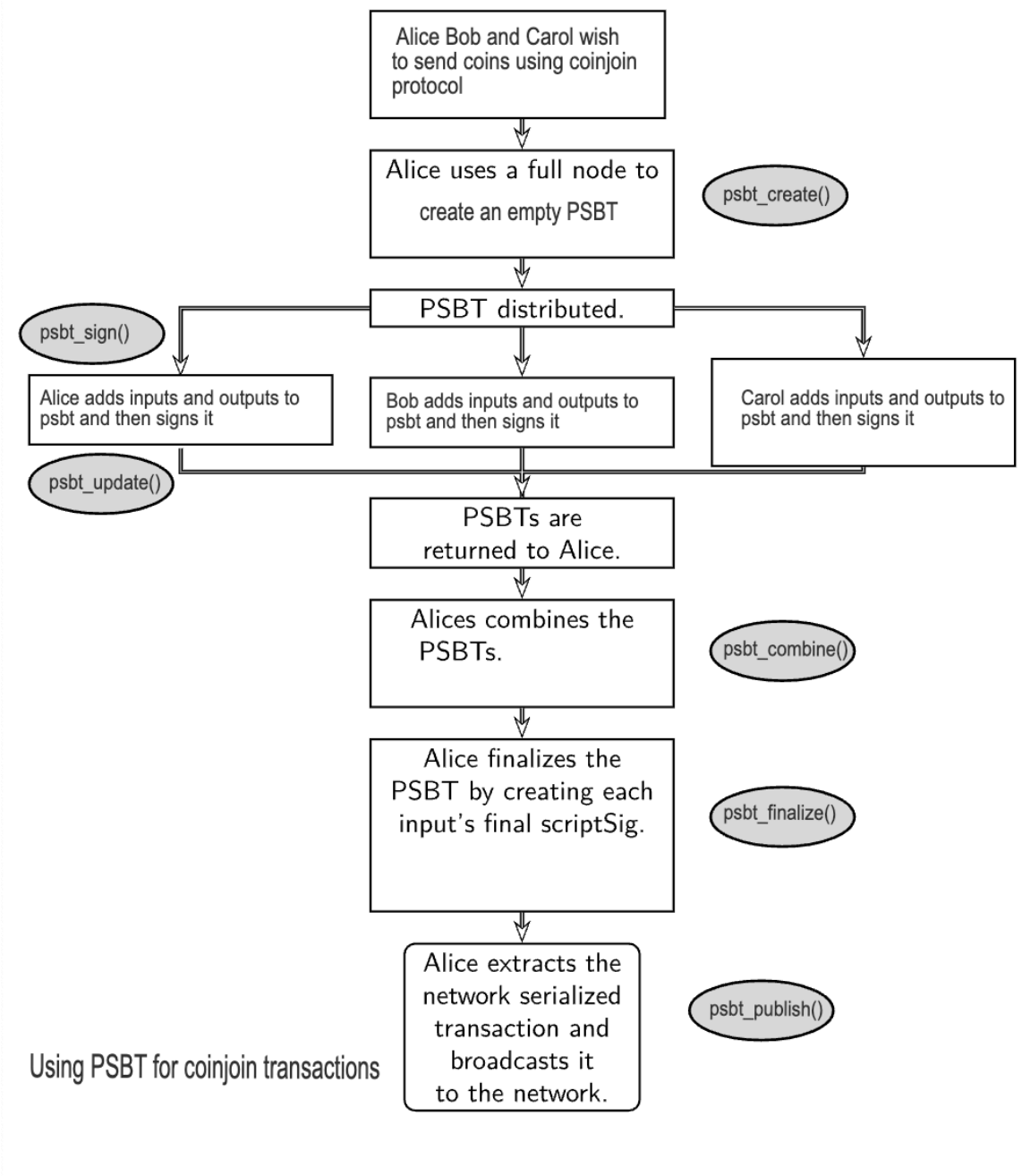

Fig-2 - Coinjoin transaction with v2_psbt

Technical Details:

- The first step is to understand the library currently used in wallet, and find relevant functions that our API will need.
- Next, we need to import the functions which exist, and plan the executions if any functionality is not present.

- Then, we will start working on each of the APIs. Usage of each API is demonstrated in ***Fig-1*** and ***Fig-2***.
- We will define 5 APIs which will perform all the requirements stated in BIP 174. Implementation of `psbt_update()` is kept as an additional deliverable as the implementation of BIP 370 has not been completed yet.
    1. `psbt_create()`: Creates a raw transaction.
       `Parameters: List of inputs and outputs`
       `Response: Base64 PSBT.`
    2. `psbt_sign():` Signs the relevant inputs
       `Parameters: Raw PSBT, Private key`
       `Response: Base64 PSBT with one/all inputs signed.`
    3. `psbt_combine():` Combines multiple versions of the same PSBT into one, to be passed on to the finalizer.
       `Parameters: Multiple versions of the same PSBT`
       `Response: One single Base64 PSBT`
    4. `psbt_finalize():` Finalises any partial signatures, and if all inputs are finalised, converts the result to a fully signed transaction which can be broadcast.
       `Parameters: Base64 PSBT with signed inputs.`
       `Response: Fully signed transaction ready to be broadcast.`
    5. `psbt_publish():` Publish the fully signed transaction to the blockchain
       `Parameters: Fully signed transaction`
       `Response: txid`

Project Timeline

Community Bonding and Onboarding(May 9, 2024 - May 23, 2024)	
`May 9, 2024 - May 23, 2024	• Get familiar with the codebase and the mentors. • Discuss my approaches, logic, and milestones with the mentor. • Attend webinars related to bitcoin.
Project Period(May 23, 2024 - Aug 15, 2024)	
May 24, 2024 - May 31, 2024	• Setting up the required environment of the Blockcore wallet. • Code the functions which are unavailable in the libraries, and will be necessary for the implementation. • Start documenting the project along with the code.
Jun 1, 2024 - Jun 15, 2024	• Start parsing the PSBT in different scenarios. This will be cornerstone for `psbt_create()` and `psbt_combine()` • Implement `psbt_create()`, `psbt_combine()`

	• Keep on documenting things.
Jun 16, 2024 - Jul 3, 2024	• Implement `psbt_sign()` • Implement `psbt_finalise()` and `psbt_publish()`
Jul 4, 2024 - Jul 8, 2024	• Discuss the stretch goal with the mentor and finalise what to do. • Keep testing the APIs for possible bugs and errors
Jul 9, 2024 - Jul 31, 2024	• Work on the stretch goal. • Extensive testing, catching bugs, fixing them and further improving the code coverage
Aug 1, 2024 - Aug 7, 2024	• Finish up the documentation.
Aug 8, 2024 - Aug 15, 2024	• The final week has been left free for completing any remaining work (if any). This provides sufficient cushion for making sure that the timeline is followed. If everything gets completed smoothly before this period then this will be utilised for optimising the code.

Demo App for Blockcore wallet:

As suggested by dangershony, I implemented a demo app using the blockcore API which can connect to your blockcore wallet, sign messages as well as send coins. I implemented the task in reactJs. I utilised 3 APIs of Blockcore web3 provider, namely `wallets` (to connect to wallet) , `signMessage` (to sign a message) and `payment` (to send coins). This helped me get a hold of APIs already used by wallet, as well as helped me understand the codebase better.

Github Repo can be found here. (I have also included a demo video in the Readme)

Future Deliverables

I intend to stay with the community after the Summer of Bitcoin and will try to optimise the blockcore wallet. I will also try updating the code with coming versions of PSBTs. I'll keep on contributing to improve the features. It would give me immense pleasure to be a part of the Blockcore community and improve the lives of those who use the products of Blockcore.

Benefits to Community

Platforms such as **Angor** will benefit from PSBTs. People will be able to send coinjoin transactions, resulting in more security and privacy. Currently, very few wallets support this protocol. It is important to step up, so more secure practices get adopted. It will also simplify and give better user experience for multisig transactions.

Biographical Information

I'm **Tirth Bhayani**, a sophomore currently pursuing **Computer science and Technology** at the **Indian Institute of Technology (BHU)** Varanasi and am expected to graduate in the year 2026.

I have cracked some prestigious exams such as JEE Advanced (All India Rank **598** of 0.9 Million aspirants) and KVPY (All India Rank **1088** of 250k aspirants) in my high school.

I then started coding in my freshman year. I started doing competitive programming with C++ (Codeforces handle: ttirth1234518), and later explored Machine learning to become a part of my college's programming club : **COPS**. I also pursued a semester-long project in third semester on anomaly detection.

My introduction to web3 and blockchain development happened during a competition (Inter-IIT). I was part of the blockchain cohort, where we developed a decentralised exchange for future trading. Though I did not represent the team ultimately, I learned a lot in the process, and started to explore web3 more.

Along with that, I also pursued a 3 month internship with **LaunchX Labs**, a Bengaluru based start up. With them I developed a chrome extension which helps you send a personalised connect request based on your profile, your prospects profile as well as a given context. It was react based.

And my most recent project was a semester long project where I worked on optimising deadline based cloud scheduling. Our aim was to increase **reliability** while reducing energy consumption.

References

- https://guide.summerofbitcoin.org/
- https://github.com/bitcoin/bips/blob/master/bip-0174.mediawiki
- https://github.com/bitcoin/bitcoin/blob/master/doc/psbt.md
- https://github.com/bitcoin/bips/blob/master/bip-0370.mediawiki
- https://docs.xverse.app/sats-connect/bitcoin-methods/signpsbt
- https://leather.gitbook.io/developers/bitcoin-methods/signpsbt#broadcast-psbt