Some personal user experiences

2023 Feb 28

See all posts

In 2013, I went to a sushi restaurant beside the Internet Archive in San Francisco, because I had heard that it accepted bitcoin for payments and I wanted to try it out. When it came time to pay the bill, I asked to pay in BTC. I scanned the QR code, and clicked "send". To my surprise, the transaction did not go through; it appeared to have been sent, but the restaurant was not receiving it. I tried again, still no luck. I soon figured out that the problem was that my mobile internet was not working well at the time. I had to walk over 50 meters toward the Internet Archive nearby to access its wifi, which finally allowed me to send the transaction.

Lesson learned: internet is not 100% reliable, and customer internet is less reliable than merchant internet. We need in-person payment systems to have *some* functionality (NFC, customer shows a QR code, whatever) to allow customers to transfer their transaction data directly to the merchant if that's the best way to get it broadcasted.

In 2021, I attempted to pay for tea for myself and my friends at a coffee shop in Argentina. In their defense, they did not *intentionally* accept cryptocurrency: the owner simply recognized me, and showed me that he had an account at a cryptocurrency exchange, so I suggested to pay in ETH (using cryptocurrency exchange accounts as wallets is a standard way to do in-person payments in Latin America). Unfortunately, my first transaction of 0.003 ETH did not get accepted, probably because it was under the exchange's 0.01 ETH deposit minimum. I sent another 0.007 ETH. Soon, both got confirmed. (I did not mind the 3x overpayment and treated it as a tip).

In 2022, I attempted to pay for tea at a different location. The first transaction failed, because the default transaction from my mobile wallet sent with only 21000 gas, and the receiving account was a contract that required extra gas to process the transfer. Attempts to send a second transaction failed, because a UI glitch in my phone wallet made it not possible to scroll down and edit the field that contained the gas limit.

Lesson learned: simple-and-robust UIs are better than fancy-and-sleek ones. But also, most users don't even know what gas limits are, so we really just need to have better defaults.

Many times, there has been a surprisingly long time delay between my transaction getting accepted on-chain, and the service acknowledging the transaction, even as "unconfirmed". Some of those times, I definitely got worried that there was some glitch with the payment system on their side.

Many times, there has been a surprisingly long and unpredictable time delay between sending a transaction, and that transaction getting accepted in a block. Sometimes, a transaction would get accepted in a few seconds, but other times, it would take minutes or even hours. Recently, EIP-1559 significantly improved this, ensuring that most transactions get accepted into the next block, and even more recently the Merge improved things further by stabilizing block times.

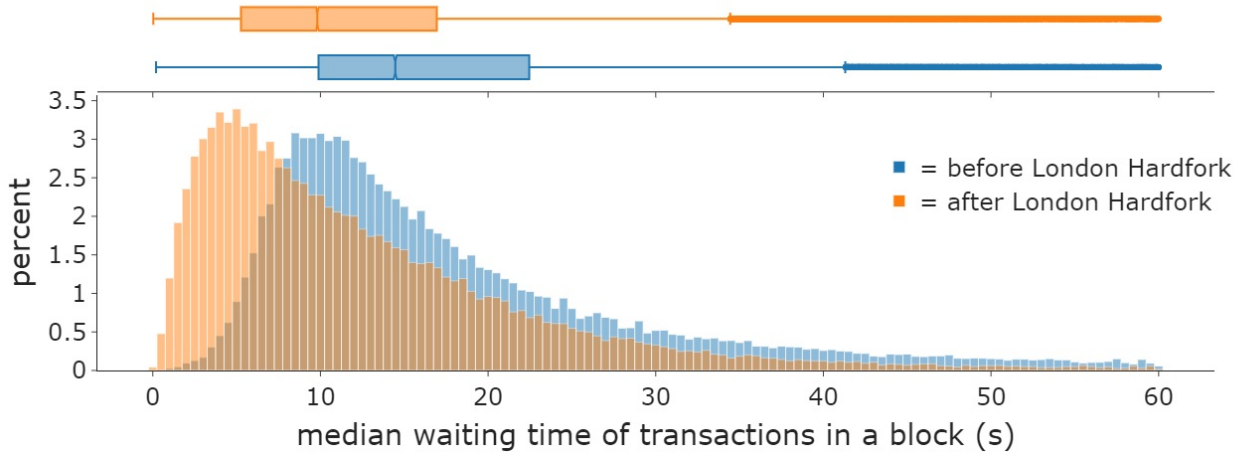

Diagram from this report by Yinhong (William) Zhao and Kartik Nayak.

However, outliers still remain. If you send a transaction at the same time as when many others are sending transactions and the base fee is spiking up, you risk the base fee going too high and your transaction not getting accepted. Even worse, wallet UIs suck at showing this. There are no big red flashing alerts, and very little clear indication of what you're supposed to do to solve this problem. Even to an expert, who knows that in this case you're supposed to "speed up" the transaction by publishing a new transaction with identical data but a higher max-basefee, it's often not clear where the button to do that actually is.

Lesson learned: UX around transaction inclusion needs to be improved, though there are fairly simple fixes. Credit to the Brave wallet team for taking my suggestions on this topic seriously, and first increasing the max-basefee tolerance from 12.5% to 33%, and more recently exploring ways to make stuck transactions more obvious in the UI.

In 2019, I was testing out one of the earliest wallets that was attempting to provide social recovery. Unlike my preferred approach, which is smart-contract-based, their approach was to use Shamir's secret sharing to split up the private key to the account into five pieces, in such a way that any three of those pieces could be used to recover the private key. Users were expected to choose five friends ("**guardians**" in modern lingo), convince them to download a separate mobile application, and provide a confirmation code that would be used to create an encrypted connection from the user's wallet to the friend's application through Firebase and send them their share of the key.

This approach quickly ran into problems for me. A few months later, something happened to my wallet and I needed to actually use the recovery procedure to recover it. I asked my friends to perform the recovery procedure with me through their apps - but it did not go as planned. Two of them lost their key shards, because they switched phones and forgot to move the recovery application over. For a third, the Firebase connection mechanism did not work for a long time. Eventually, we figured out how to fix the issue, and recover the key. A few months after that, however, the wallet broke again. This time, a regular software update somehow accidentally reset the app's storage and deleted its key. But I had not added enough recovery partners, because the Firebase connection mechanism was too broken and was not letting me successfully do that. I ended up losing a small amount of BTC and ETH.

Lesson learned: secret-sharing-based off-chain social recovery is just really fragile and a bad idea unless there are no other options. Your recovery guardians should not have to download a separate application, because if you have an application *only* for an exceptional situation like recovery, it's too easy to forget about it and lose it. Additionally, requiring separate centralized communication channels comes with all kinds of problems. Instead, the way to add guardians should be to provide their ETH address, and recovery should be done by smart contract, using ERC-4337 account abstraction wallets. This way, the guardians would only need to not lose their Ethereum wallets, which is something that they already care much more about not losing for other reasons.

In 2021, I was attempting to save on fees when using Tornado Cash, by using the "self-relay" option. Tornado Cash uses a "relay" mechanism where a third party pushes the transaction on-chain, because when you are withdrawing you generally do not yet have coins in your withdrawal address, and you don't want to pay for the transaction with your deposit address because that creates a public link between the two addresses, which is the whole problem that Tornado Cash is trying to prevent. The problem is that the relay mechanism is often expensive, with relays charging a percentage fee that could go far above the

actual gas fee of the transaction.

To save costs, one time I used the relay for a first small withdrawal that would charge lower fees, and then used the "self-relay" feature in Tornado Cash to send a second larger withdrawal myself without using relays. The problem is, I screwed up and accidentally did this while logged in to my deposit address, so the deposit address paid the fee instead of the withdrawal address. Oops, I created a public link between the two.

Lesson learned: wallet developers should start thinking much more explicitly about privacy. Also, we need better forms of account abstraction to remove the need for centralized or even federated relays, and commoditize the relaying role.

Miscellaneous stuff

- Many apps still do not work with the Brave wallet or the Status browser; this is likely because they didn't do their homework properly and rely on Metamask-specific APIs. Even Gnosis Safe did not work with these wallets for a long time, leading me to have to write my own mini Javascript dapp to make confirmations. Fortunately, the latest UI has fixed this issue.
- The ERC20 transfers pages on Etherscan (eg. https://etherscan.io/address/0xd8da6bf26964af9d7eed9e03e53415d37aa96045#tokentxns) are very easy to spam with fakes. Anyone can create a new ERC20 token with logic that can issue a log that claims that I or any other specific person sent someone else tokens. This is sometimes used to trick people into thinking that I support some scam token when I actually have never even heard of it.
- Uniswap used to offer the functionality of being able to swap tokens and have the output sent to a different address. This was really convenient for when I have to pay someone in USDC but I don't have any already on me. Now, the interface doesn't offer that function, and so I have to convert and then send in a separate transaction, which is less convenient and wastes more gas. I have since learned that Cowswap and Paraswap offer the functionality, though Paraswap... currently does not seem to work with the Brave wallet.
- Sign in with Ethereum is great, but it's still difficult to use if you are trying to sign in on multiple devices, and your Ethereum wallet is only available on one device.

Conclusions

Good user experience is not about the average case, it is about the worst case. A UI that is clean and sleek, but does some weird and unexplainable thing 0.723% of the time that causes big problems, is worse than a UI that exposes more gritty details to the user but at least makes it easier to understand what's going on and fix any problem that does arise.

Along with the all-important issue of high transaction fees due to scaling not yet being fully solved, user experience is a key reason why many Ethereum users, especially in the Global South, often opt for centralized solutions instead of on-chain decentralized alternatives that keep power in the hands of the user and their friends and family or local community. User experience has made great strides over the years - in particular, going from an average transaction taking minutes to get included before EIP-1559 to an average transaction taking seconds to get included after EIP-1559 and the merge, has been a night-and-day change to how pleasant it is to use Ethereum. But more still needs to be done.