Suppose that it is announced tomorrow that quantum computers are available, and bad actors already have access to them and are able to use them to steal users' funds. Preventing such a scenario is the goal of quantum-resistant cryptography

(eg. Winternitz signatures, STARKs), and once account abstraction is in place, any user can switch to using a quantum-resistant signature scheme on their own schedule. But what if we don't have that much time, and a sudden quantum transition happens long before that?

I argue that actually, we are already

well-positioned to make a pretty simple recovery fork to deal with such a situation

. The blockchain would have to hard fork and users would have to download new wallet software, but few users would lose their funds.

The main challenge with quantum computers is as follows. An Ethereum address is defined as keccak(priv_to_pub(k))[12:]

, where k

is the private key, and priv_to_pub

is an elliptic curve multiplication to convert the privkey into a pubkey. With quantum computers, elliptic curve multiplications become invertible (because it's a discrete-log problem), but hashes are still safe. If a user has not made any transactions with their account, then only the address is publicly visible and they are already safe. But if a user has made even one transaction, then the signature of that transaction reveals the public key, which in a post-quantum world allows revealing the private key. And so most users would be vulnerable.

But we can do much better. The key realization is that in practice, most users' private keys are themselves the result of a bunch of hash calculations

. Many keys are generated using BIP-32, which generates each address through a series of hashes starting from a master seed phrase. Many non-BIP-32 methods of key generation work similarly: eg. if a user has a brainwallet, it's generally a series of hashes (or medium-hard KDF) applied to some passphrase.

This implies the natural structure of an EIP to hard-fork the chain to recover from a quantum emergency:

1. Revert all blocks after the first block where it's clear that large-scale theft is happening
2. Traditional EOA-based transactions are disabled
3. A new transaction type is added to allow transactions from smart contract wallets (eg. part of RIP-7560), if this is not available already
4. A new transaction type or opcode is added by which you can provide a STARK proof which proves knowledge of (i) a private preimage x

, (ii) a hash function ID 1 <= i < k

from a list of k

approved hash functions, and (iii) a public address A

, such that keccak(priv_to_pub(hashes[i]))[12:] = A

. The STARK also accepts as a public input the hash of a new piece of validation code for that account. If the proof passes, your account's code is switched over to the new validation code, and you will be able to use it as a smart contract wallet from that point forward.

For gas efficiency reasons (after all, STARKs are big), we can allow the STARK to be a batch proof, proving N STARKs of the above type (it has to be a STARK-of-STARKs rather than a direct proof of multiple claims, because each user's x

needs to be kept private from the aggregator).

The infrastructure to implement a hard fork like this could in principle start to be built tomorrow, making the Ethereum ecosystem maximally ready in case a quantum emergency does actually come to pass.