We (Tas Dienes, Dan Shaw, and Andreas Freund), have been exploring ways to make Ethereum (primarily Mainnet) more appealing to the enterprise market.

This post is designed to both stimulate discussion and poll the community on the need to enable the dynamic usage of a given set of cryptographic algorithms, also often called “pluggable crypto”, by both Dapps and Clients at runtime within the Ethereum stack, apart from the current ones utilized: secp256k1, keccak256, BN256 (as precompiles). “Pluggable crypto” refers to the ability of Dapps to choose more than one type of cryptographic primitive for use in their smart contracts, and for Clients to do the same, not only in the EVM but also account management, transaction-validation, and in elements of a Block such as a Blockhash.

It is motivated by two things:

- Future-proofing: With the rise of L2, the need for cryptographic aggregator schemes in Eth2, and the active research on stateless clients using novel cryptographic approaches such as recursive zero-knowledge proofs, there is a rising need to be able to use more cryptographic algorithms and use them more flexibly.
- Compliance Considerations: secp256k1 and keccak256 are not NIST compliant. This makes the use of Ethereum Mainnet or Enterprise Ethereum clients by government and enterprises significantly more difficult and, often, impossible. Implementing a “pluggable crypto” approach in Ethereum, at least with Eth2, would avoid the lengthy and uncertain process of convincing NIST to accept the current Ethereum cryptographic algorithms. and, thus, make adoption of Mainnet beyond the current ecosystem, in particular by regular companies, significantly easier.

Hence, the provocative hypothesis: Ethereum, at least with Eth2, should have “pluggable crypto” built-in.

Please, let us know your thoughts on this hypothesis/discussion points and why you are in favor or against such a development effort.

cc @tasd