I'm not the author of this EIP, but I can't find a thread for it. Also, I would like to know if Ed25519 support is something that is still under consideration, or it has been rejected.

https://eips.ethereum.org/EIPS/eip-665

Simple Summary

Support performant and cheap verification of Ed25519 cryptographic signatures in smart contracts in general by adding a precompiled contract for Ed25519 signature verification to the EVM.

Abstract

Verification of Ed25519 cryptographic signatures is obviously possible in EVM bytecode. However, the gas cost will be very high, and computationally expensive, as such tight, wide word operations intensive code as required for Ed25519 is not a good fit for the EVM bytecode model.

The addition of a native compiled function, in a precompiled contract, to the EVM solves both cost and performance problems.

Motivation

Ed25519 and Ed448 (that is, EdDSA using Curve25519 or Curve448) are IETF recommendations RFC7748) with some attractive properties:

- Ed25519 is intended to operate at around the 128-bit security level and Ed448 at around the 224-bit security level
- EdDSA uses small public keys (32 or 57 octets) and signatures (64 or 114 octets) for Ed25519 and Ed448, respectively
- Ed25519/Ed448 are designed so that fast, constant-time (timing attack resistant) and generally side-channel resistant implementations are easier to produce

Despite being around only for some years, post-Snowden, these curveshave gained wide use quickly in various protocols and systems:

- TLS / ECDH(E) (session keys)
- TLS / x.509 (client and server certificates)
- DNSSEC (zone signing)
- OpenSSH (user keys)
- GNUPG/OpenPGP (user keys)
- OpenBSD Signify (software signing)

One motivation for Ed25519 signature verification in smart contracts is to associate

existing off-chain systems, records or accounts that use Ed25519 (like above) with blockchain addresses or delegate

by allowing to sign data with Ed25519 (only), and then submit this Ed25519-signed data anonymously (via any Eth sender address) to the blockchain, having the contract check the Ed25519 signature of the transaction.

Another motivation is the processing of external, Ed25519 proof-of-stake based blockchains within Ethereum smart contracts.

When a transactions contains data that comes with an Ed25519 signature, that proves that the sender of the Ethereum transaction was also in control of the private key (and the data), and this allows the contract to establish an association between the blockchain and the external system or account, and the external system establish the reverse relation.

For example, a contract might check a Ed25519 signed piece of data submitted to the Ethereum transaction like the current block number. That proves to the contract, that the sender is in possession of both the Ethereum private key and the Ed25519 private key, and hence the contract will accept an association between both. This again can be the root anchor for various powerful applications, as now a potentially crypto holding key owner has proven to be in control of some external off-chain system or account, like e.g. a DNS server, a DNS domain, a cluster node and so on.