WARNING : THIS SCHEME IS HALF BAKED AND PLACED HERE EXCLUSIVELY FOR OPEN DISCUSSION / FEEDBACK

Background

As part of Project Chicago for Cryptocommodities (https://projectchicago.io/), we intend to analyze a range of blockchain resources and the cryptocommodities that they represent (and therefore the higher-order financial instruments they are able to create).

Part of the goals of such a project include the creation of fully incentive-compatible systems operating robustly in the rational actor model, with fair pricing for both providers and consumers of blockchain resources. Unfortunately, several such resources, including the current peer-to-peer network, follow an unincentivized commons model which simply fails to achieve these properties.

Recent work (https://fc18.ifca.ai/bitcoin/papers/bitcoin18-final18.pdf) has identified three base resources of note in Ethereum transactions: computation, network, and storage. To build efficient markets for all three resources, some price discovery mechanism is required. In Ethereum, of these, computation currently enjoys the most robust such mechanism, with a two-sided gas market between miners and clients and relatively trustless derivative instruments enabling more advanced forms of speculation and risk management (eg https://gastoken.io/). In this post, we seek to extend such an incentivization model to the p2p network layer.

Strawman Scheme

For the sake of discussion and to establish feasibility, let us consider a scheme not optimized for efficiency / overhead / etc. Optimizing and improving the scheme will continue in future work.

For simplicity, let's also assume the current miner-based PoW blockchain architecture pre-sharding. There are natural analogues to this scheme for the PoS and/or sharded blockchain model.

For each transaction, a sender generates some number (say, 4) fresh keys, SK_1, ..., SK_n

, PK_1, ..., PK_n

.

With each transaction, the sender s

signs _{PK_s}

. A transaction is considered invalid if a user's balance is not sufficient to pay for gas + hopfee * tx_size. When relaying, the sender includes the vector SK_1, ..., SK_n

to the relayer. The first relayer adds an onion-like layer to the transaction, signing <*{PK_s}, A_1*>{PK_1}

and sending the remaining data SK_2, ..., SK_n

to all potential next-hops in the relay path. In the case of an invalid SK, the relayer discards the transaction and bans the corresponding peer, bounding the denial of service vector involved with sending fake keys or signatures and performing the corresponding, somewhat expensive ECDSA operations. This process is repeated until a transaction reaches a miner. When included in a block, as part of ``miner-like" fees, each included relayer address is paid the hopfee from the user's balance. Validity of the process is checked, including that all sequential keys exist and are appropriately used in order. An honest miner will accept only the first such routed transaction, adding this transaction to its mempool as usual. Any transaction that is signed by k<n

public keys pays the miner (n-k)*hopfee*tx_size

in additional fees.

Relayers can now choose which transactions to relay based on their included hopfee, statically and efficiently rejecting transactions with low hopfees. Relayers can also efficiently reject requests to which there are no corresponding secret keys through which they are paid. Each relayer (including potential miners) can only be paid for hops after the first hop at which they received the transaction, as they have no knowledge of the secret keys before that point. Relayers can double sign, but if n

is chosen carefully to only slightly exceed the expected number of hops in the network, they risk some future relayer potentially running out of secret keys and rejecting the transaction, potentially losing them the relay race and thus profit on expectation. n

is chosen by the user carefully, with an economically rational user choosing an n

that is high enough for routing to succeed, but as low as possible to avoid excessive fee payment. The network also enforces a maximum n

.

It is worth noting that by pre-sigining a set of addresses, a user can also enforce a given routing path to the miner, never distributing keys that enable miners to choose a route on which they are an earlier hop. This gives users relatively high levels of granularity in their choice of route. Furthermore, each relayer can potentially pre-sign for the next relayer, requiring a p2p-level public key exchange. This may however be undesirable for privacy, as it encourages address reuse over time.

Incentives and Analysis

(this analysis is not exhaustive, cursory, and likely flawed, but my intuition is that any major incentive incompatibilities here are fairly easy solvable)

Users who wish to have their transactions relayed are incentivized to obey all validity rules, providing an appropriately sized valid vector of secret/private keys, obeying the network-wide limit on n

, and ensuring they hold enough balance to pay hop fees. Users who wish to perform a denial of service attack have several choices of griefing vectors, though the most expensive verification of validity is the check that the private keys and previous signatures are valid. This is a relatively bounded amplification vector, though some careful analysis is required on this vector (if it turns out to be bad, one option is requiring deposits for incentivized/prioritized relays, and having some quota of free relays to allow users to bootstrap, or simply considering such invalid transactions as throws that pay full gas and can be included in blocks).

Relayers are incentivized to relay transactions as quickly as possible, winning the race for miner fees (probabilistically, under honest miner majority). Any attempts to modify previous relayers' onion layers will make the transaction invalid without knowledge of the original secret keys vector, and will delay propagation, potentially causing a loss in expected income. Relayers can double sign, but likely will not, as miners earn more from transactions with shorter relay paths and will prefer such transactions for inclusion (and double singing is also an operation which may delay propagation).

Miners are incentivized to include all transactions without modification, as they earn fees from any remaining hops between k

and n

. This directly incentivizes miners to build and support relay networks with maximally short paths, increasing both their per-transaction take and relayer fee take. Miners are also incentivized to be peers in this relay network, allowing them knowledge of earlier secret keys in the vector and increasing their overall take. In general, if miners participate in the relay network, they will have a competitive advantage over other users proportional to their hashpower, meaning that it is possible that the system will essentially devolve to relaying transactions directly to miners. This, however, is OK, as if miners are not available to some part of the network, other relayers can take up the task of building the missing paths that may be suffering from censorship or unavailability.

An open question is the value of relaying; if the value of each relay is really small, the cost of the blockspace to include relayers' signatures may exceed the value of the payments they secure. This is likely less of an issue as scalability proceeds to make block space far cheaper, but worth thinking about.

Current full nodes are incentivized to be relayers if they are generally on the critical relay path of at least some transactions. If they are not, their overhead to the p2p layer may mean that it is preferable for such nodes to exclusively receive transactions, saving overall network overhead.

Building Futures

An important property of both efficient price discovery and efficient marketplaces for cryptocommidities, including relayer bandwidth, is the existence of a robust market for speculation on the underlying commodity.

This scheme allows for such a marketplace to arise. For example, cash-settled oracle-based futures that settle based on the average hop fees in a given period can be created, as can in-protocol futures using this approach.

The exploration of the full range of financial instruments enabled by this approach is left to future work.

Acknowledgments

Thanks to Ari Juels for proposing the original version of this scheme, and for helpful discussions that lead to exploration of

this problem.