SUAVE could be used to build a programmable coprocessor for Ethereum mainnet as well as rollups like Base. Nobody to this date has made a crypto coprocessor that is able to do private compute over shared state, while A LOT of applications depend on it.

Here we describe one possible architecture for it based on typical crypto coprocessor model.

Architecture

The design is illustrated by the graph below. The green boxes represent program running inside SGX.

[

image

2705×1034 428 KB

](https://collective.flashbots.net/uploads/default/original/2X/6/69d446544b60049994f657f3b705f86dc5bcc387.png)

Lifecycle of a user:

1. People register compute jobs on Base (e.g., call a contract that emits an event SUAVECoprocess(job)
2. an event loop calls a SUAVE-geth node running inside an SGX, sending a transaction that calls the BaseCoprocessingContract

on the function MonitorBase

1. the function MonitorBase

fetches the latest block on Base, get all the transactions, goes through the events they emit, upon seeing an event that registers a job, do the job (by routing to the best Service Subnet), then post the result back by sending a transaction to Base

1. enjoy applications enabled by private credible compute today

Contract

contract MonitorBase { uint256 public lastBlock; // Last block number that was processed

```
// Event definitions
event JobProcessed(bytes indexed job, bytes result);
event TransactionSent(bytes indexed txn);

// Main monitoring function
function monitorBase() external {
    // Fetch events using a precompiled function (or an oracle)
    bytes[] memory events = Suave.doHTTPRequest(etherscan, base, lastBlock);

    for (uint i = 0; i < events.length; i++) {
        // Process each event to get a co-processing job
        bytes memory job = getCoprocessingJob(events[i]);

        // Execute the job using a precompiled function (or an oracle)
        bytes memory result = Suave.doHTTPRequest(subnetAPIRouter(job), job);

        // Send the result as a transaction
        sendTxn(result, events[i].info);

        emit JobProcessed(job, result);
    }

    // Update the last processed block
    lastBlock = block.number;
}

// Function to send a transaction
function sendTxn(bytes memory data, bytes memory contractInfo) internal {
    // Create and sign a transaction using precompiled functions
    bytes memory txn = Suave.signEthTransaction(Suave.createTxn(data, contractInfo), baseChainId);

    // Send the transaction using a precompiled function (or an oracle)
    Suave.doHTTPRequest(baseRPC, txn);

    emit TransactionSent(txn);
}
```

}

Roadmap

Here we use the most degen example of AI services for the roadmap.

1. PoC on Rigil testnet + ChatGPT as a Subnet
2. PoC of SUAVE-geth inside SGX
3. Open source LLM as a Subnet
4. Open source LLM subnet on SGX
5. attestation of all SGX parts on Ethereum mainnet