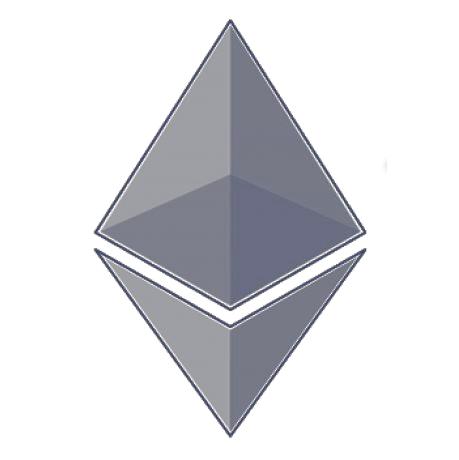
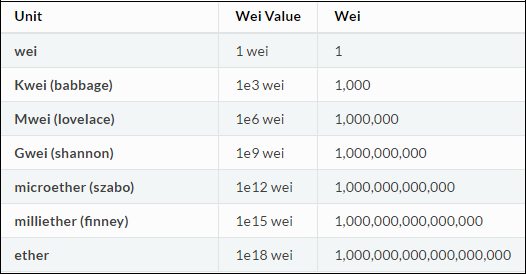
Ethereum?

* **Open source** public Blockchain network
  + Value token = **Ether**
  + De-centralized Turing-complete Virtual Machine
  + Smart contracts platform
  + Execution requires payment - gas



Ethers (ETH)

* Ethereum : Value token
* Denominations:



Ethers Supply

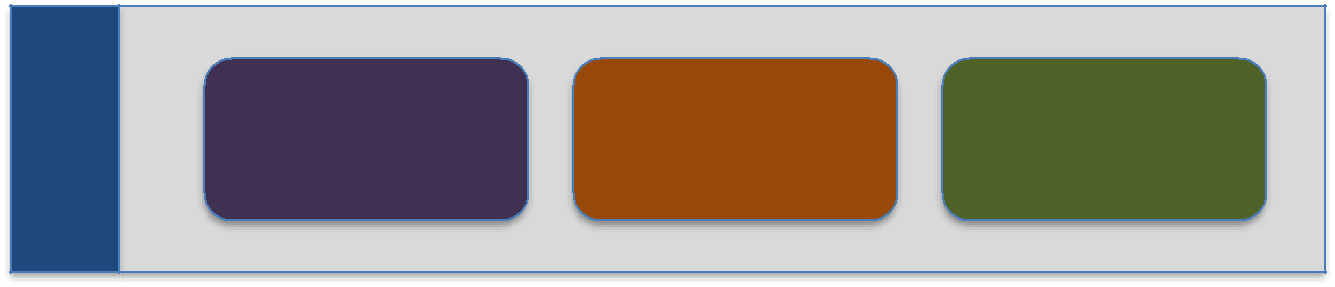
* Ether creation
  + Presale (2014): 60 Million
  + 12 Million created to fund the development
  + 5 Ethers created as reward for every block; roughly ~14 seconds
  + Sometimes 2-3 Ethers for non-winning miners (*uncle rewards*)
* Contract invocation – Users pay by *Ethers*
* Incentive for the miners

EVM

* An software that can execute Ethereum Bytecode

• Follows the EVM specifications *(Ethereum protocol)*

* Runs as a process on a computer/sever



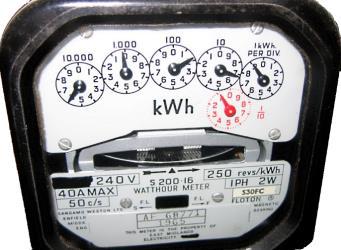
|  |
| --- |
| EVM |

|  |  |  |  |
| --- | --- | --- | --- |
| Memory | Stacks | Execution |  |
| Area | Engine |  |
|  |  |

* EVM implemented in multiple languages

Gas

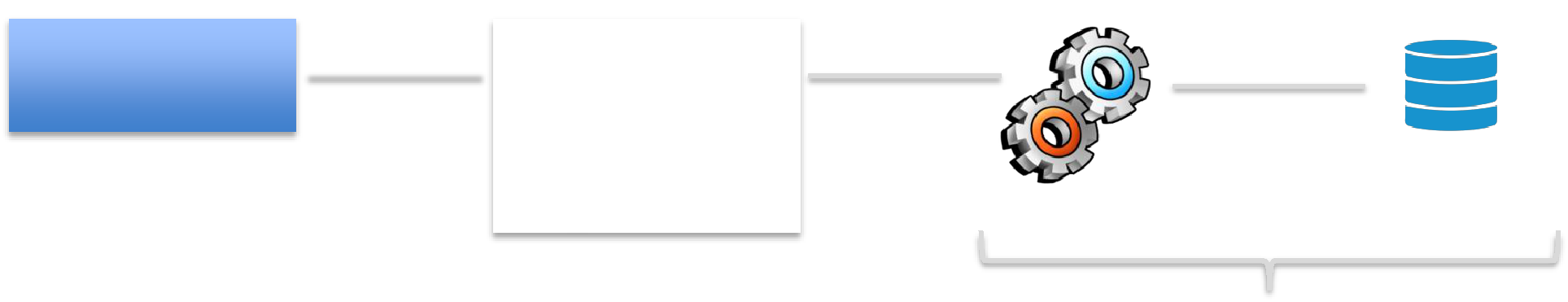
* User invoking the transaction pays for the execution



**Measures:** kWHused **Measures:** Gallonsof water used

* Gas is the unit in which EVM resource usage is measured

Gas Calculations

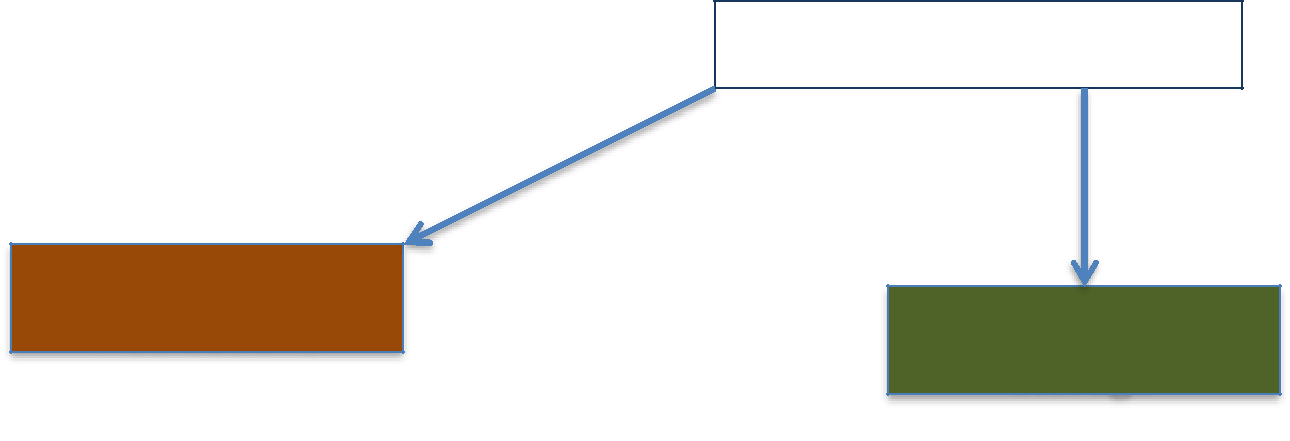


|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Transaction |  |  |  | Instructions | |  |
|  |  |  | • | ADD |  |
|  |  |  |  |
|  |  |  |  | • | MUL |  |
|  |  |
|  |  |  |  | • | JMP |  |
|  |  |  |  | • | … |  |

Type/Number of

instructions

Execution Storage



Fee paid by originator

Amount of

storage

Opcodes & Gas



Fee Calculation

**gasUsed** = Instructions executed (summed up gas)

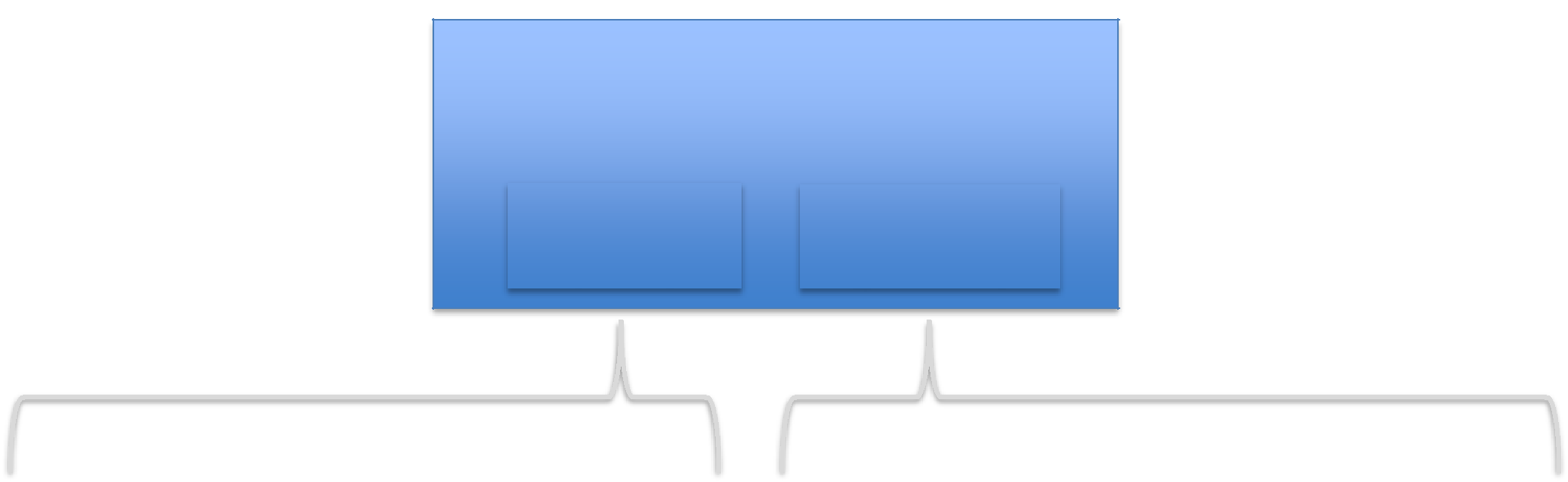
**gasPrice** = User specified in the transaction

Miners decides the minimal acceptable price



**Transaction Fee = gasUsed \* gasPrice**

Transaction Fee : Parameters



Transaction

……

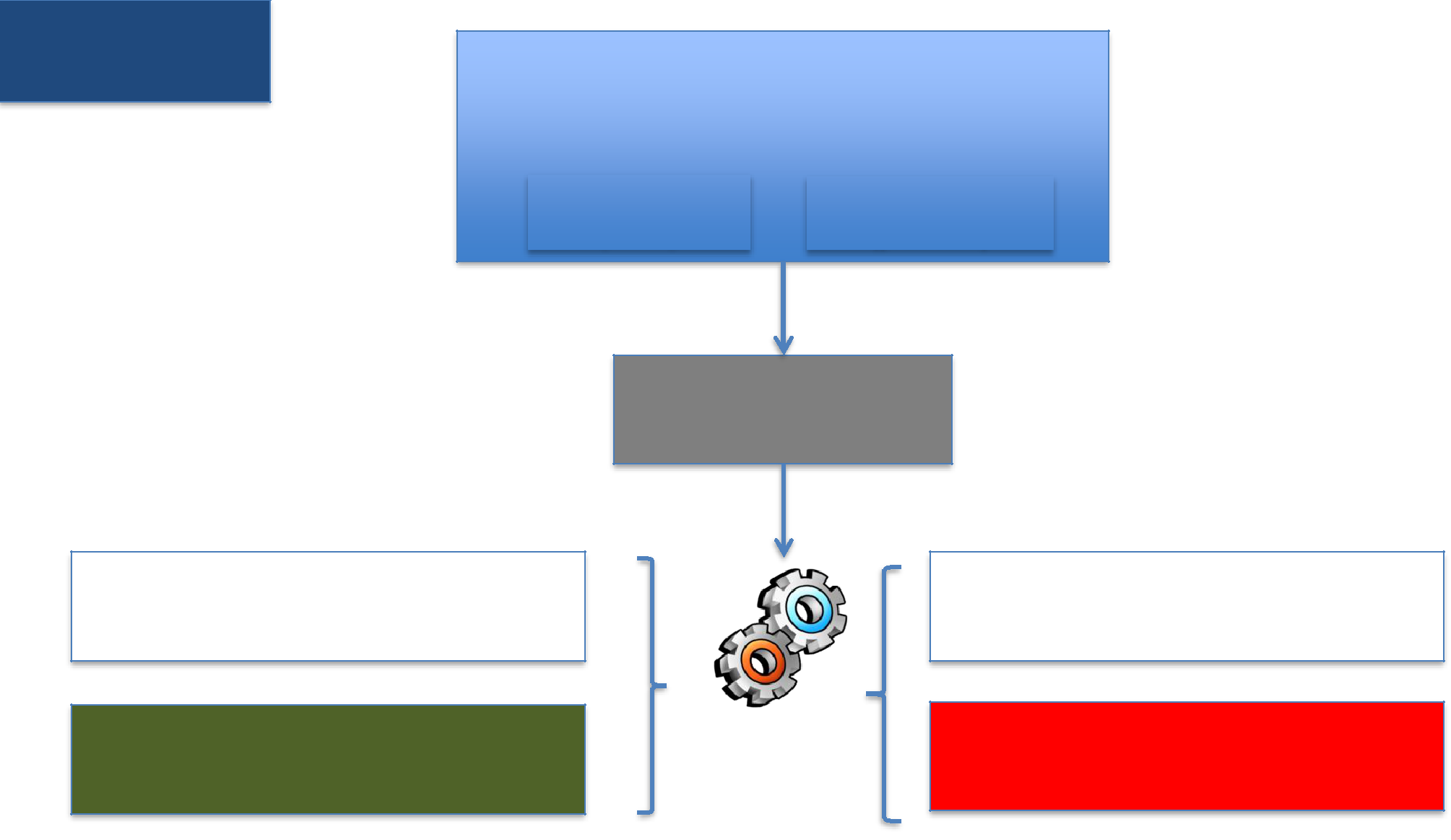
……

|  |  |  |
| --- | --- | --- |
| Start Gas |  | Gas Price |
| (units) |  | (µ Ethers) |
|  |  |  |

• Max units of gas originator willing to spend • Per unit gas price that originator willing to pay

Processing

Transaction



……

……

|  |  |  |
| --- | --- | --- |
| Start Gas |  | Gas Price |
| (units) |  | (µ Ethers) |

**Escrow:**

startGas\*gasPrice

**Fee Paid:**

**gasUsed**\*gasPrice

**Refund:**

(startGas - gasUsed)\*gasPrice

**Fee Paid:**

**startGas**\*gasPrice

**Out of gas exception No changes made**

Consensus

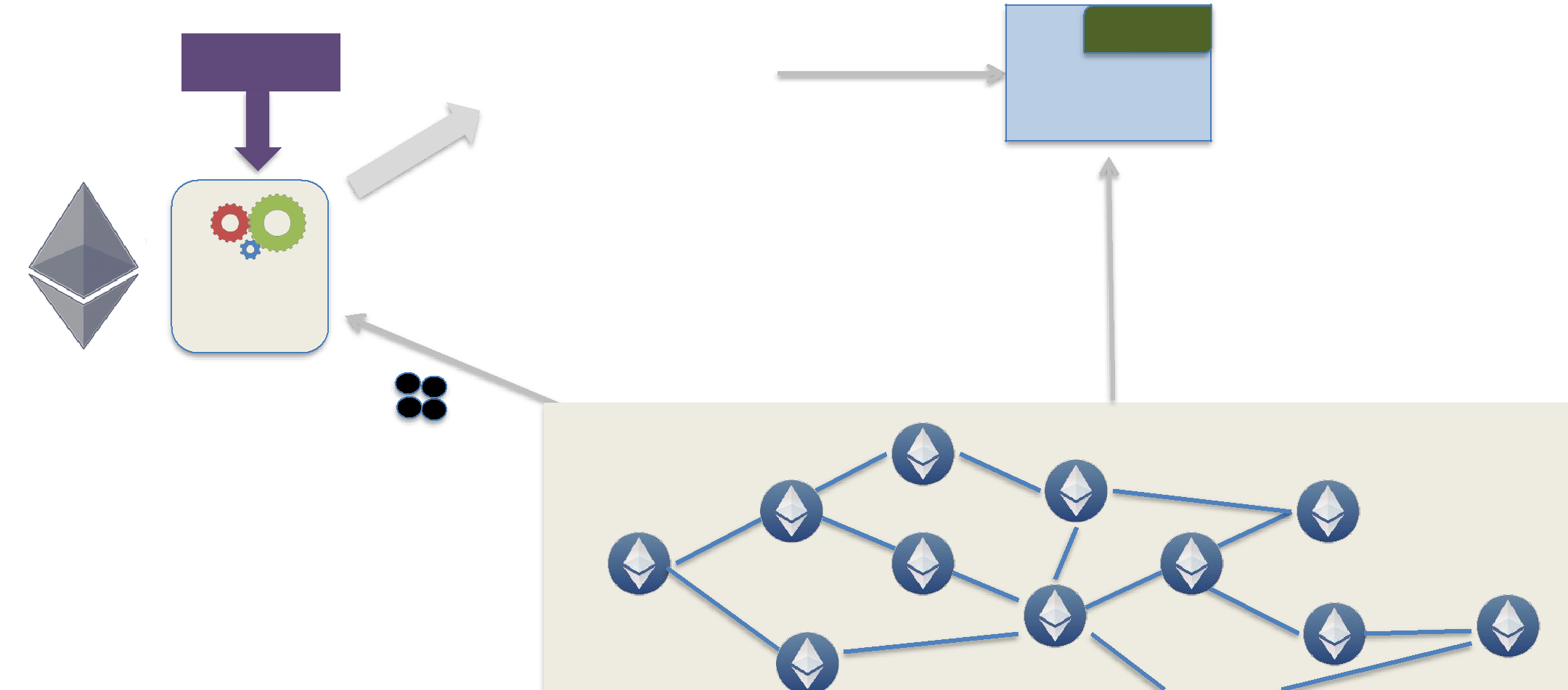
* Process by which blocks get created
  + Validate transactions
    - Secures the network

|  |  |  |
| --- | --- | --- |
| Proof of Work |  | Proof of Stake |
|  |  |  |

* Incentive driven model

• Fixed reward in tokens • Transaction fee

|  |  |
| --- | --- |
| Proof of Work | • Computationally (CPU | Memory | Bandwidth) intensive |
|  |  |



Proof of

Work

|  |  |  |  |
| --- | --- | --- | --- |
| Difficulty | **{Puzzle Solution}** |  |  |
|  | Block |  |
|  |  |  |

Solve Puzzle

Validate Block

Unconfirmed

Transactions



Ethereum: Proof of Work

• Protocol: GHOST • Algorithm: ETHash

• Difficulty: Network adjusted; block created ~14 seconds

* Incentive: 3 Ether

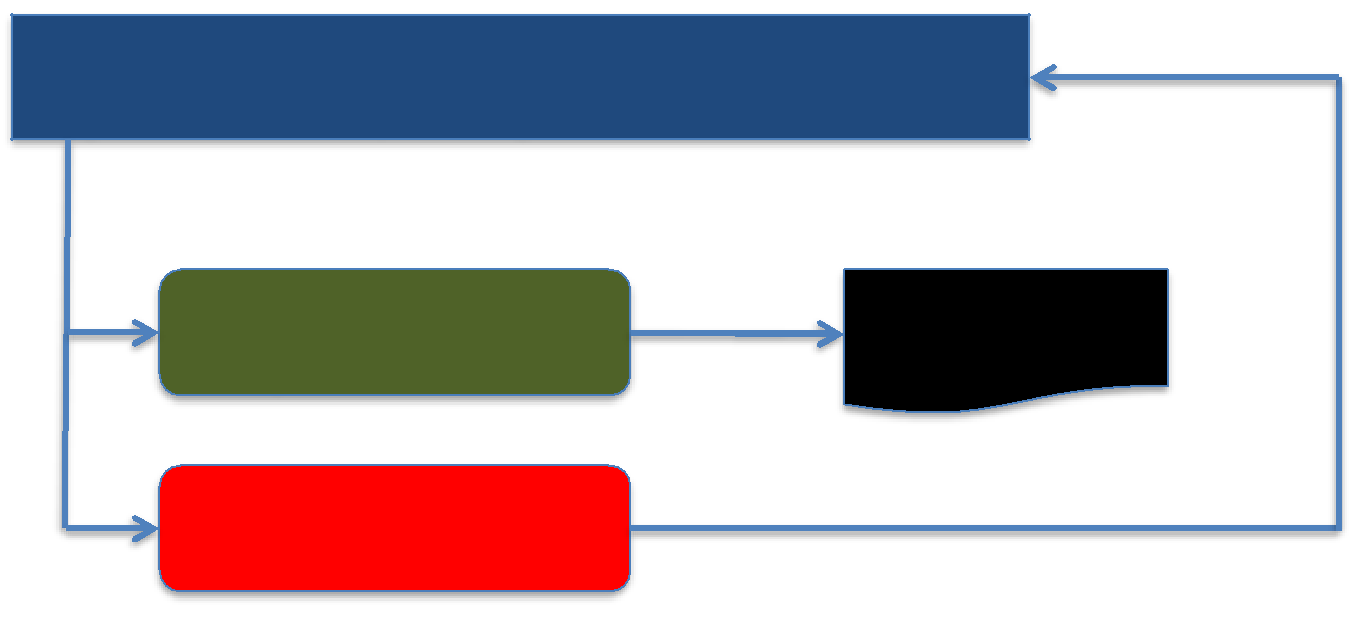
Gas fee for transactions



Proof of work is environmentally Un-Friendly

Proof of Stake

* Node to validate selected by the network | No competition
  + **Stake** –refers to the wealth that users holds on the network
  + Node that validates referred to as *Validator* not a miner



Network Selects a validator

Higher chance depending on stake of he validator

|  |  |
| --- | --- |
| Validator Node | Blocks gets |
| completed validation | added |

Validator Node

Did not validate

Ethereum: Proof of Stake

* Ethereum future version will switch to ***Proof of Stake***
  + Protocol: CASPER
* Why switch to Proof Of Stake?
  + - Reduced energy consumption
    - A lower incentive needed for motivation
    - Stake in the network will promote good behavior
    - Punishment as part of the protocol will act as deterrent

Ethereum Network

|  |  |
| --- | --- |
| Live Network | • Network ID = 1 |
|  |  |

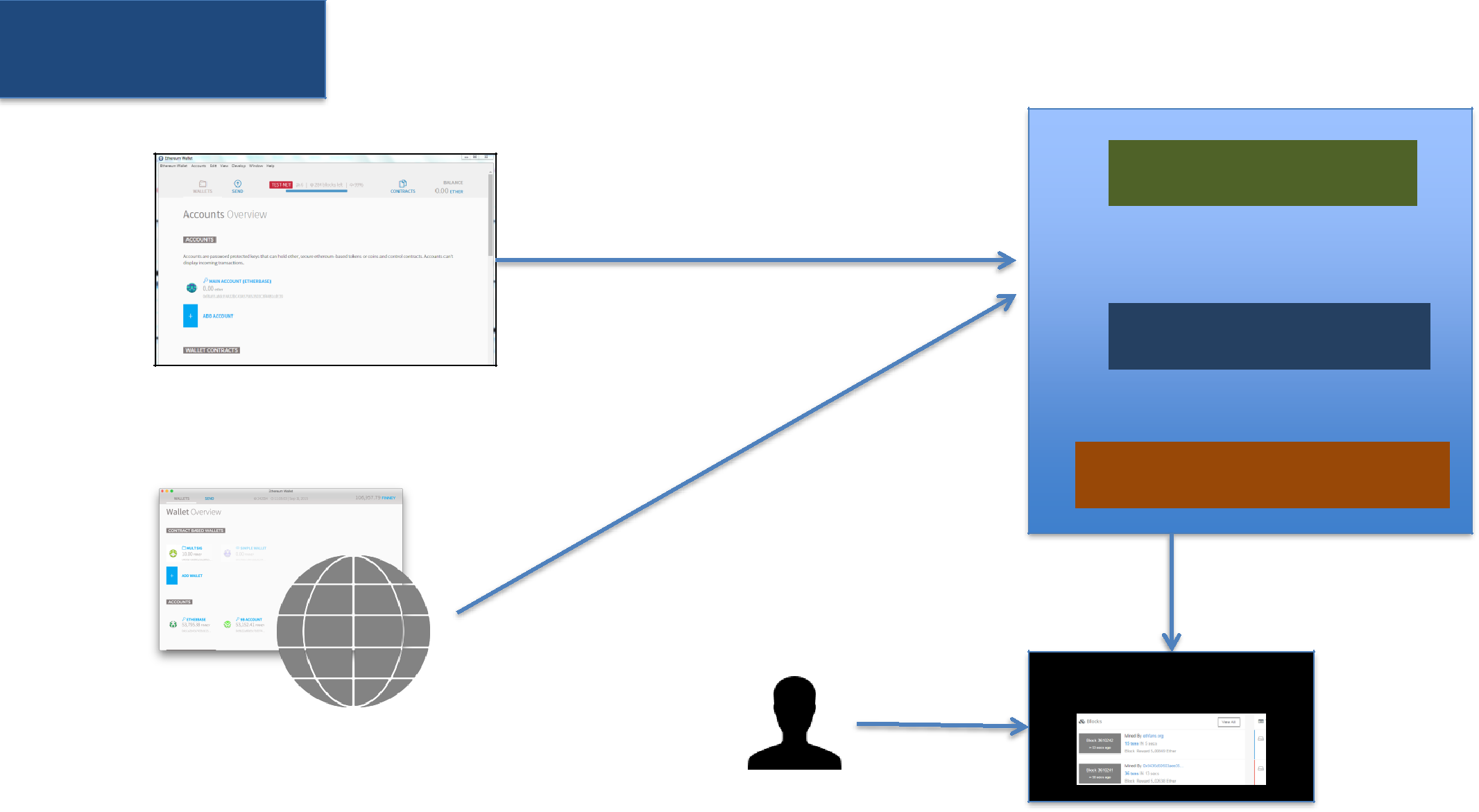
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test-Net | • | Network ID = 2 Morden | *retired* |  |
|  | • | **Network ID = 3 Ropsten** | *current* |  |
|  |  |
|  |  | *KOVAN RINKEBY (ID=4)* | *current* |  |

Private Network • Network ID = Assigned

Private Network

* Data privacy
* As a distributed database
* Consortium
* Industry verticals
* Permissioned
* Internal transactions & contracts





Interaction

Live Network

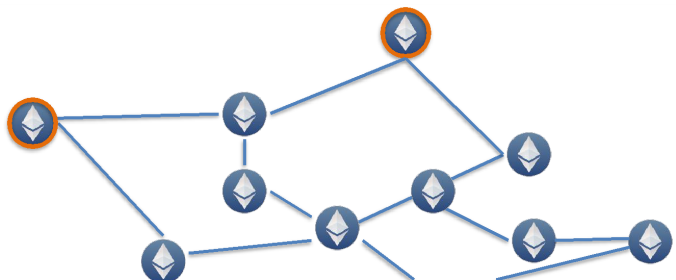
|  |  |  |
| --- | --- | --- |
| **Wallet (Mist)** | Test-Net |  |
|  |  |

Private Network

|  |  |
| --- | --- |
| Dapp | Block Explorer |

Developer

|  |  |
| --- | --- |
| Proof of Authority | Consensus Model |
|  |  |

* No concept of mining or miners
* *Pre-approved Authority nodes* validate the transaction
  + Blocks are said to be “Sealed by the nodes”

Ethereum: Proof of Authority

* Ethereum network can be configured to use PoA
* Protocol: CLIQUE



*Refer to lectures on Private Network to see*

* *How to setup a Clique private network*
* *How Puppeth and Bootnode tools are used*

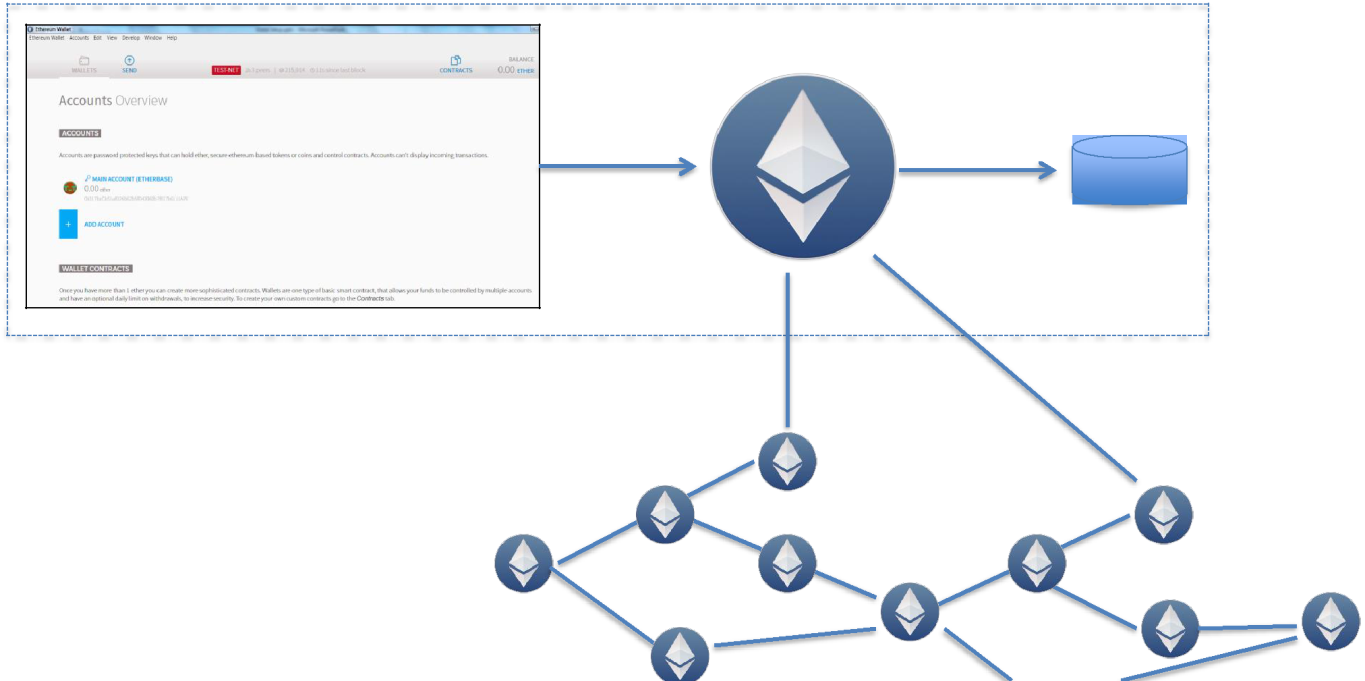
|  |  |
| --- | --- |
| Proof of Authority | Where is it used? |

**Network ID = 4 Rinkeby**

Private Network

* More Secure: Nodes with validation authority are trusted
* Configurability of Block times
* Computationally less intensive

Wallet Architecture



Eth Node

Testnet

Blockchain explorer

* Websites (or webapps) that show information on

• Transactions • Blocks • Accounts



Etherchain.org

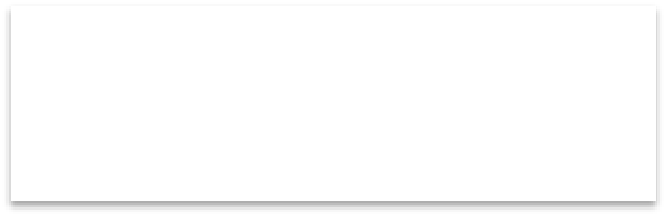
https://etherscan.io/ https://live.ether.camp/ <https://etherchain.org/>

<https://testnet.etherscan.io/>

Type of Accounts



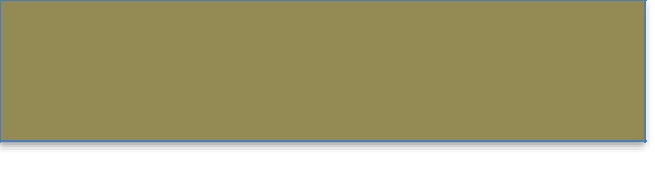
|  |  |  |
| --- | --- | --- |
| Externally Owned | • | Has an address |
| Account | • | Private key protected by password |
|  |  |  |

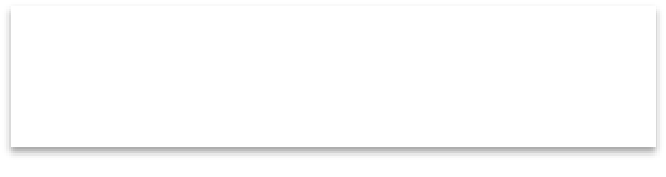


|  |  |  |
| --- | --- | --- |
| Contract Account | • Has an address but NO private key |  |
| • Holds/Run code |  |
|  |  |
|  |  |  |

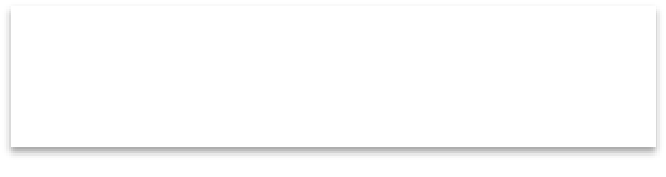
• Associated with Account(s)

• NOT free to use

Contract Account



|  |  |
| --- | --- |
| Single Owner | • One *Account* creates & owns |
|  |  |

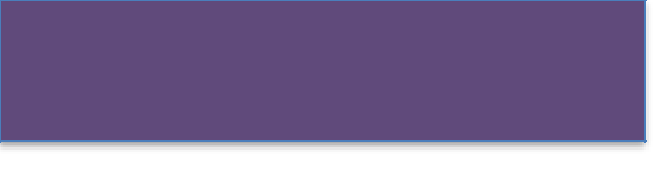


|  |  |  |  |
| --- | --- | --- | --- |
| MultiSig | • | One *Account* creates |  |
|  |  |  |
|  | • | Multiple owners |  |
|  |  |

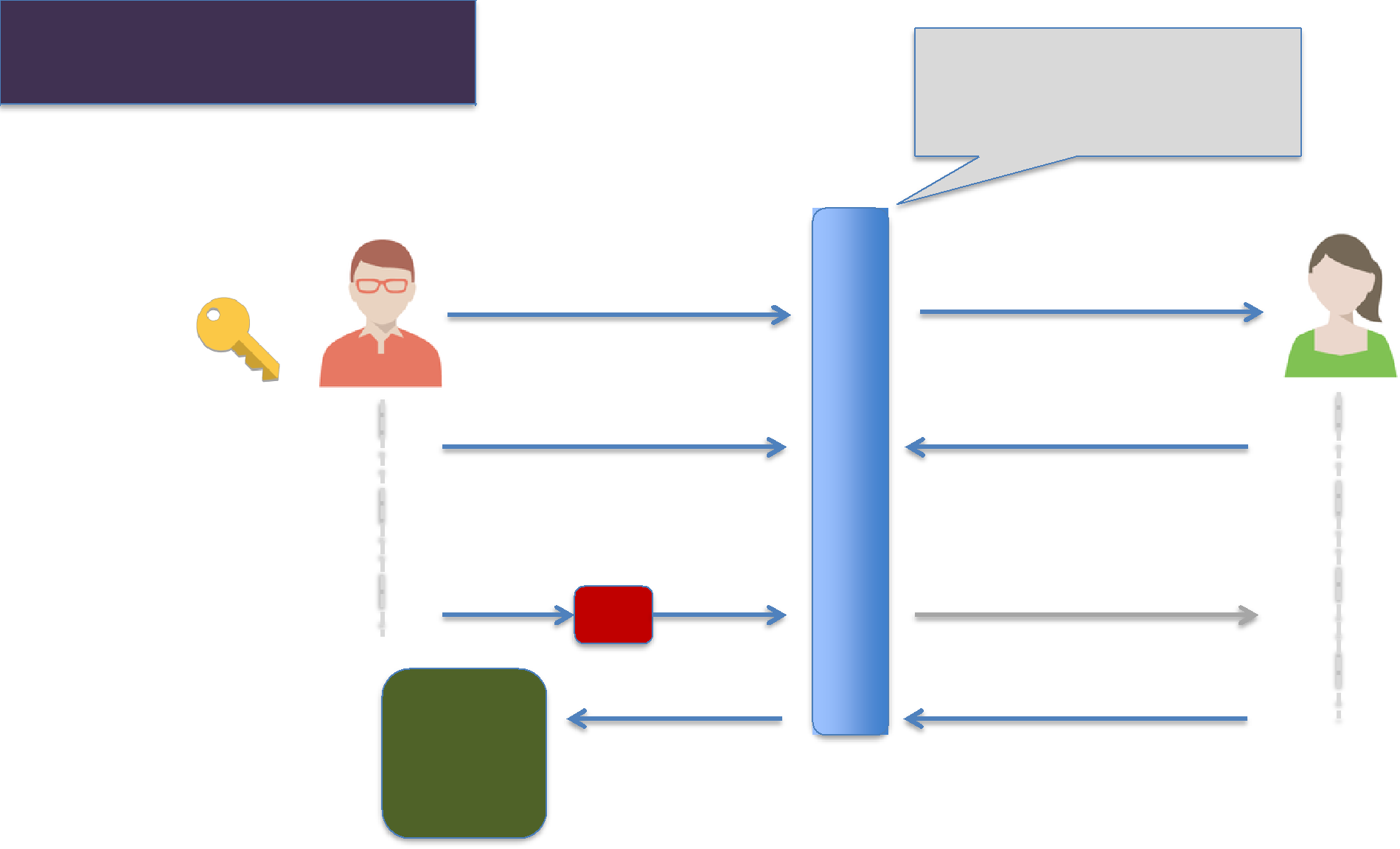
• M-of-N type wallets

N = Number of owners

M = Required to confirm transaction

Single Owner

* Accounts can't display incoming transactions
* Create simple contract to see incoming transactions

MultiSig Contract

|  |  |  |  |
| --- | --- | --- | --- |
|  | Creates | Contract |  |
|  | Send 2 Ethers |  |
|  | MultiSig |  |
| Send 3 Ether to Bob | |  |
|  |  |
|  | TX |  |  |
| Bob’sA/c | Transfer 3 Ether | |  |

Daily limit: 2

Sig required: 2

Import Contract



Send 2 Ethers

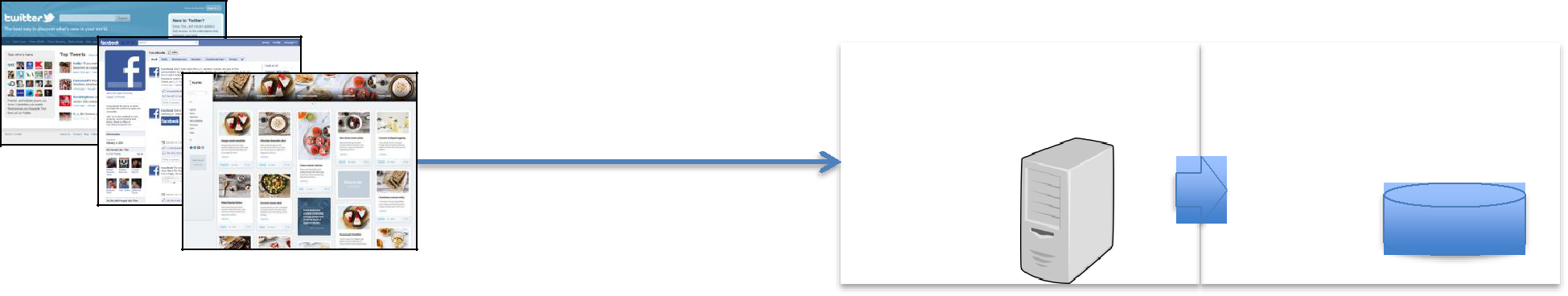
Prompts

Approves

Web App  DAPP

Centralized Resources

Owned by the organization

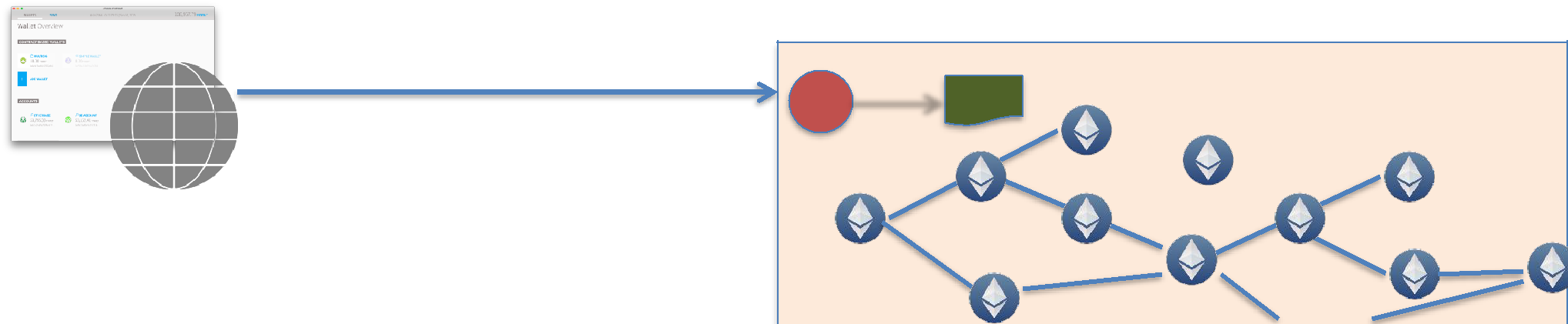


|  |  |  |
| --- | --- | --- |
| Mid Tier | Data |  |
|  |
|  |  |  |

Front end apps

Decentralized Resources

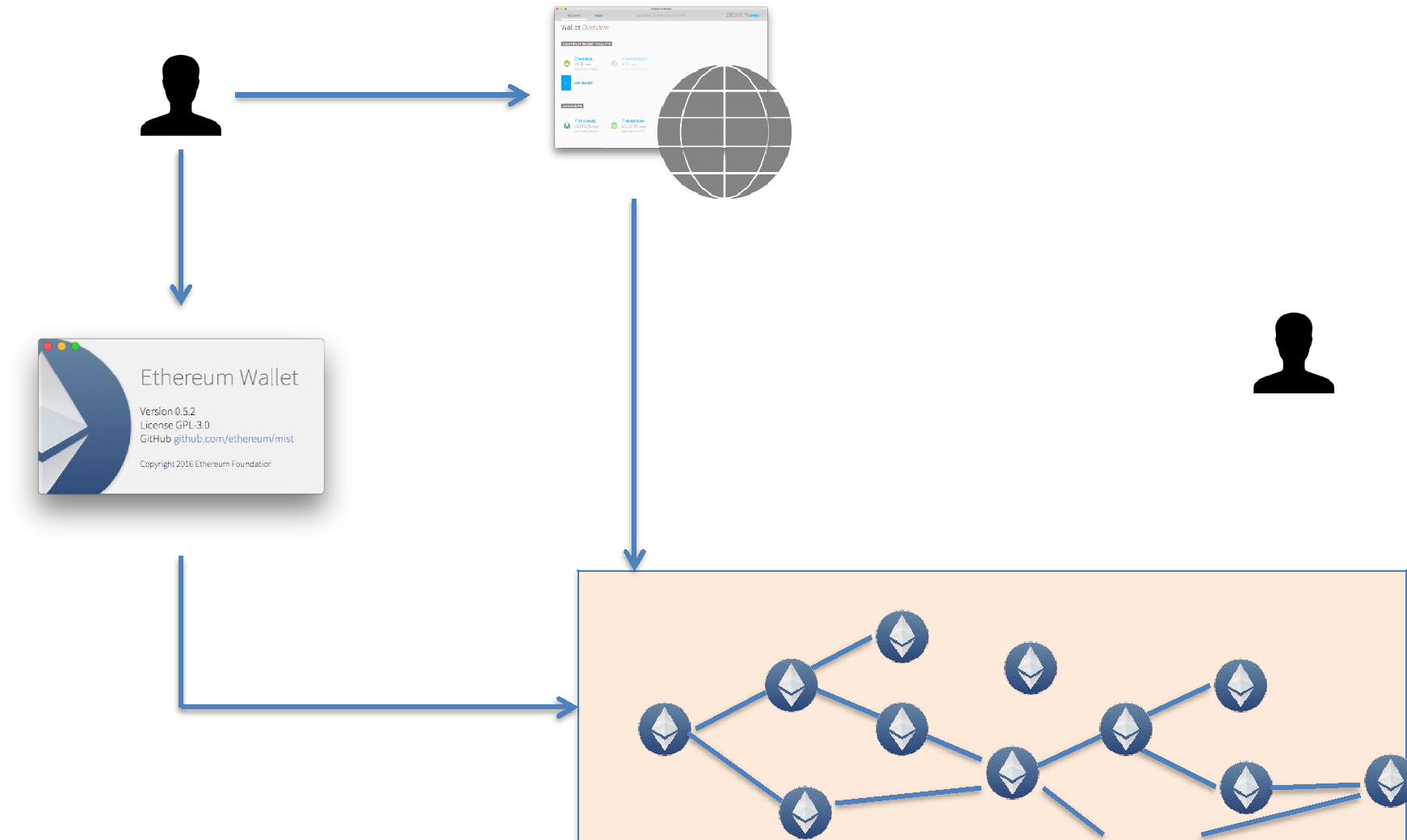
Public domain



C  Data

Working of Dapp

Invoke Contract



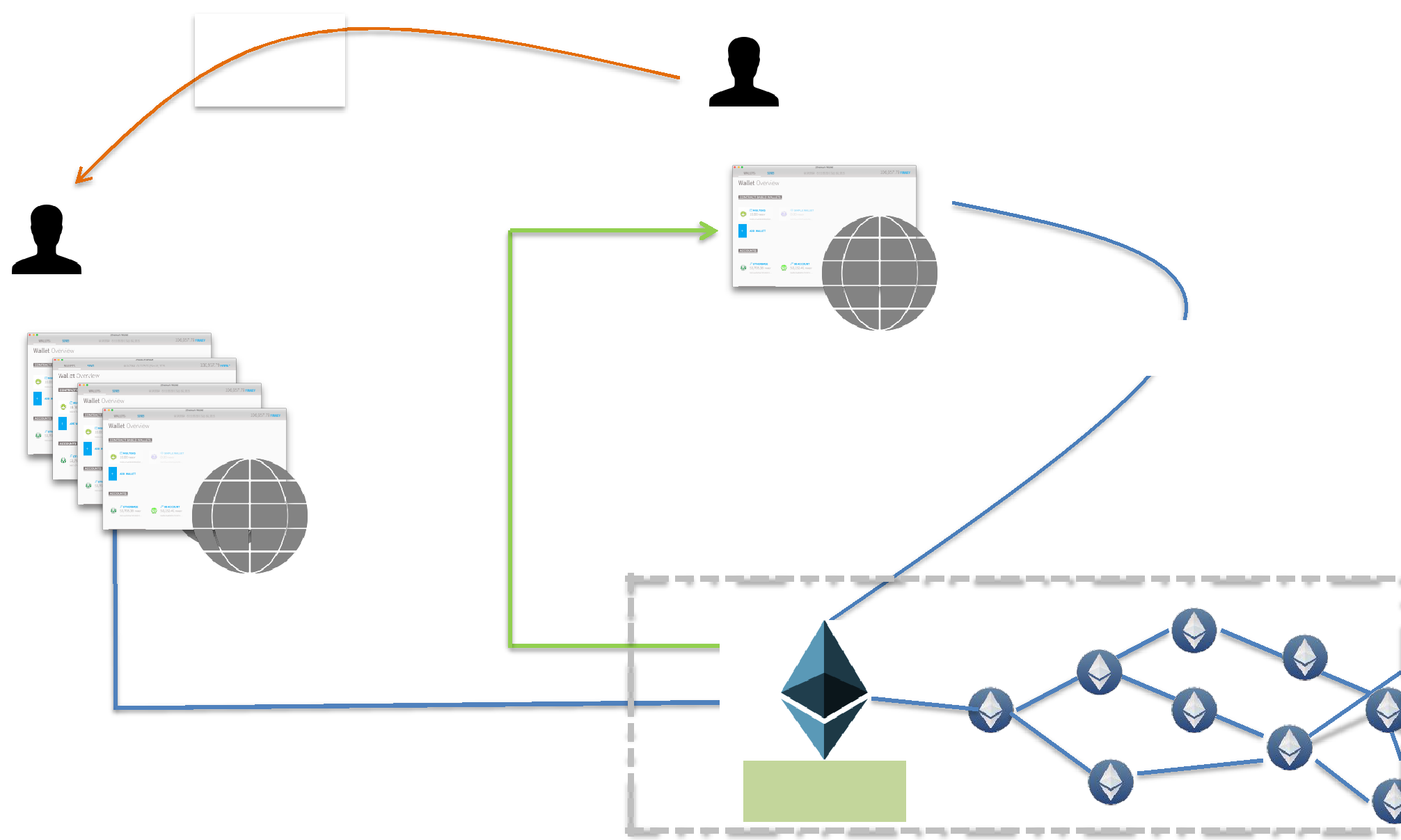
• App user pays

*gas/fee*

• Miner collects

* Manage funds
* Invoke Contracts
* Transaction validated/mined
* Recorded in ledger

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Example DAPP |  |
| Ship |  |
|  | Bidding |  |
| Goods |  |  |
|  |  |  |
|  | Seller Application | |  |



Event: BidReceived

Buyer Application

withdrawFunds()

Bid()

Contract

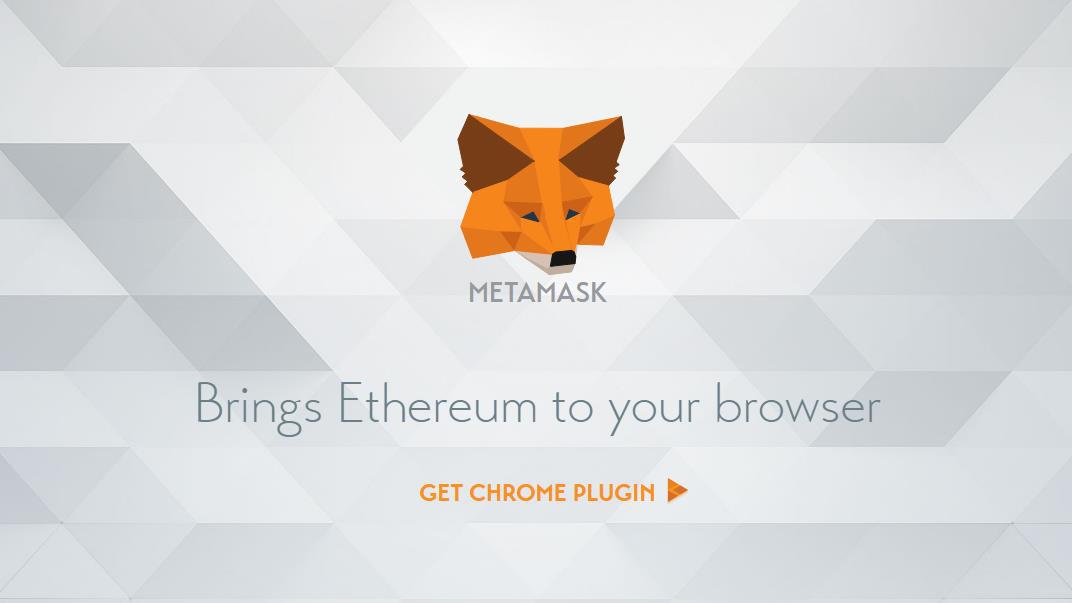
DAPP Technology Stack



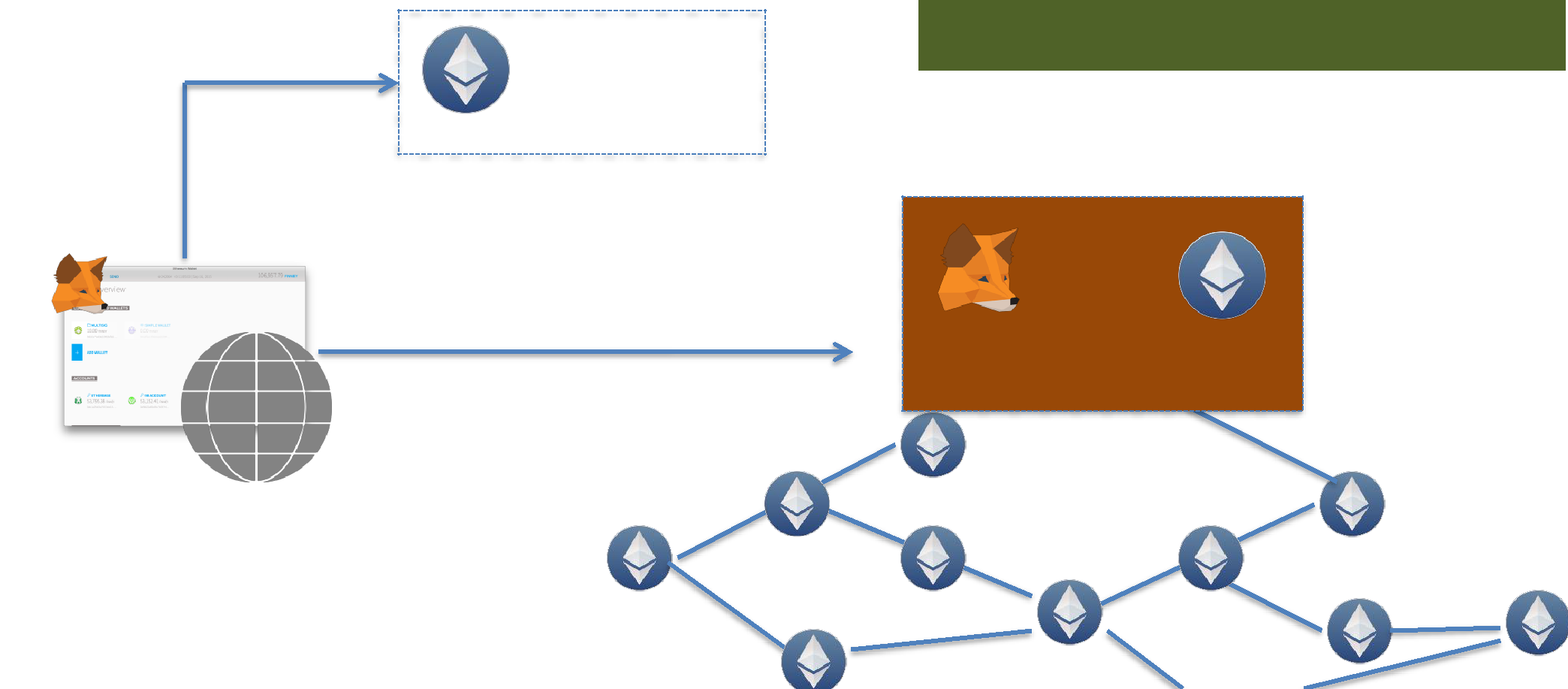
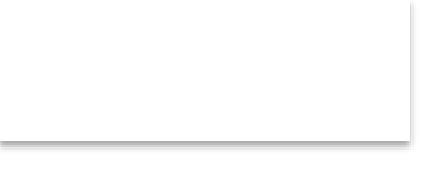
OLIDITY

Serpent

Lisp Like Language

**https://metamask.io**

|  |  |
| --- | --- |
| Meta Mask | Chrome plugin turns browser into DAPP container |
|  |  |

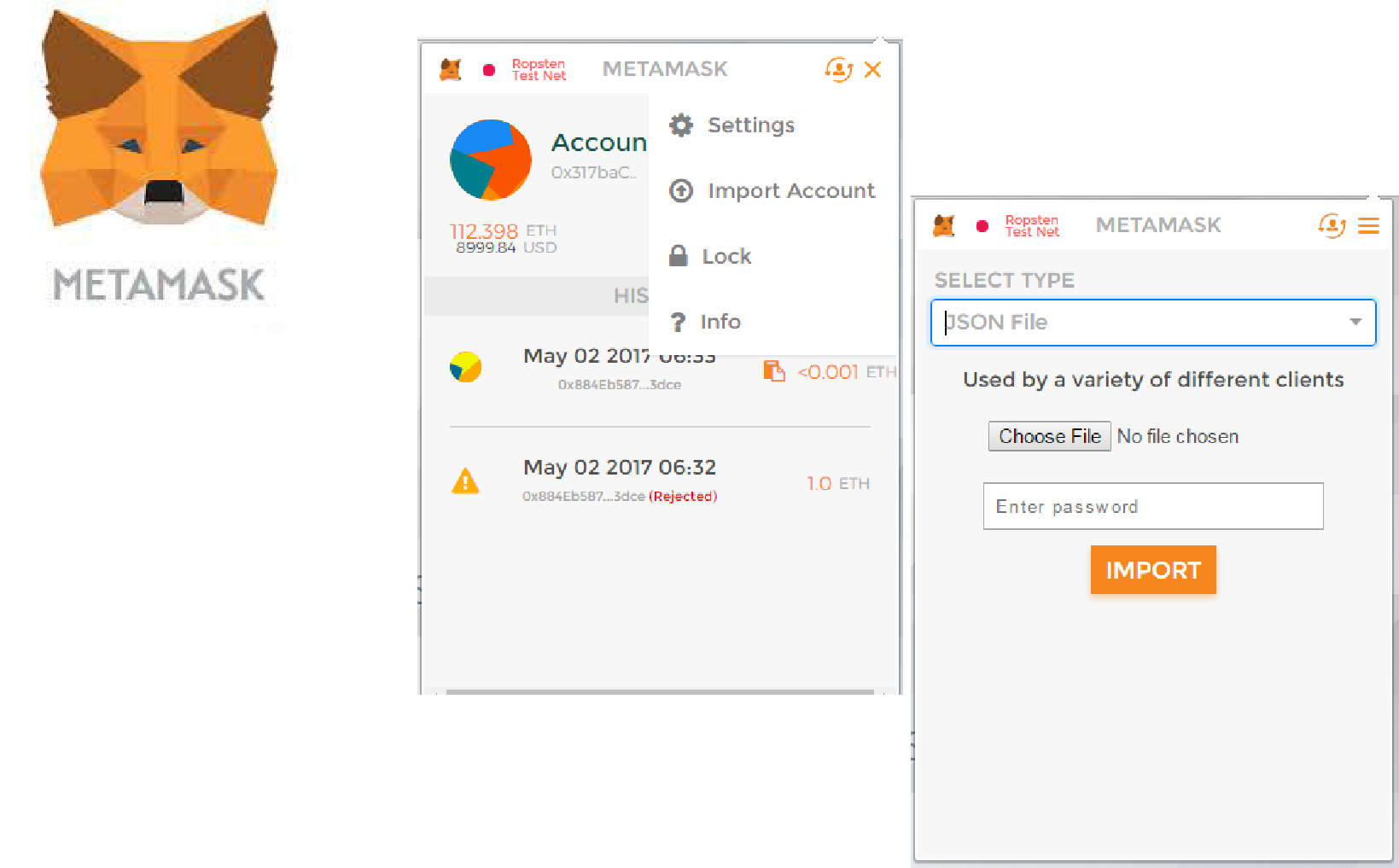


Local Node

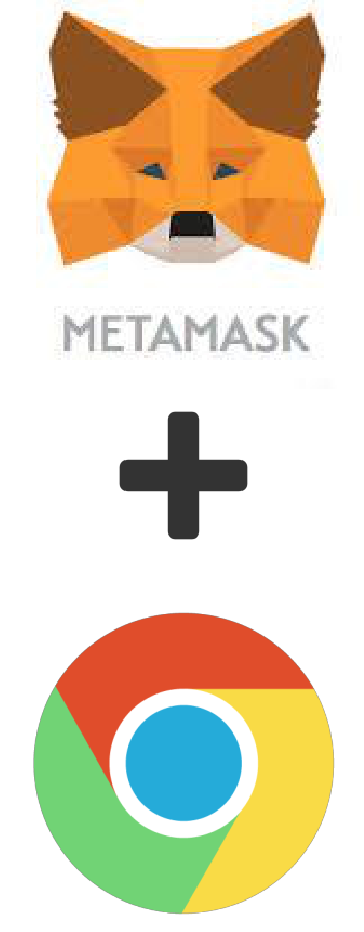
Or Simulator

Local Ethereum node not needed

Node hosted @ MetaMask



* Manage accounts in a browser vault
  + Export/Import accounts
  + Send Funds
* Exposes web3 object to browser app
  + Single Page Applications
* Supports multiple endpoints
* Does not support contract deployment
* Does not support mining



Remix https://ethereum.github.io/browser-solidity

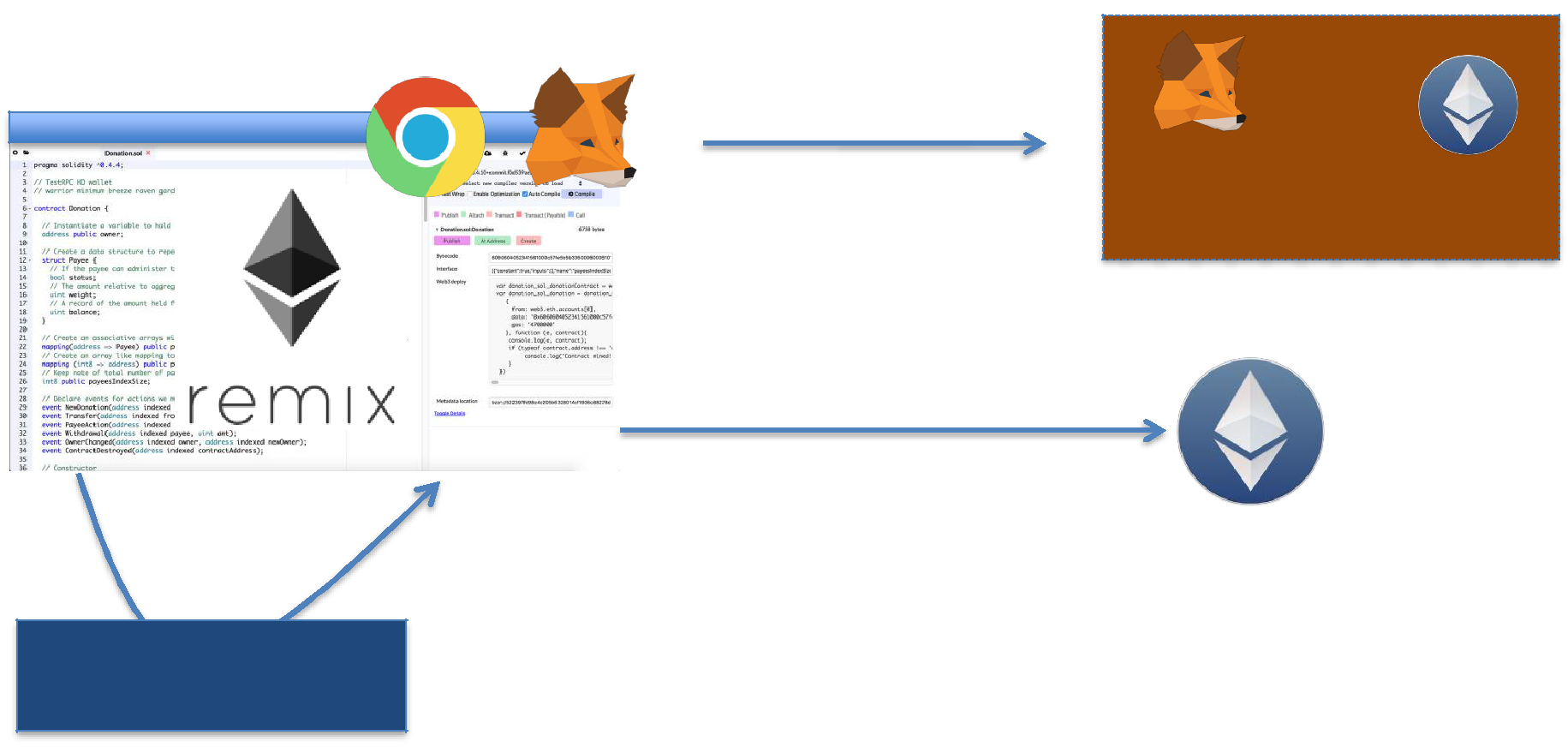
* Code smart contracts in a browser
* Test the contracts in simulator
* Deploy the contracts to live network



* Does not have account management

Browser Solidity

https://ethereum.github.io/browser-solidity



Node hosted @ MetaMask

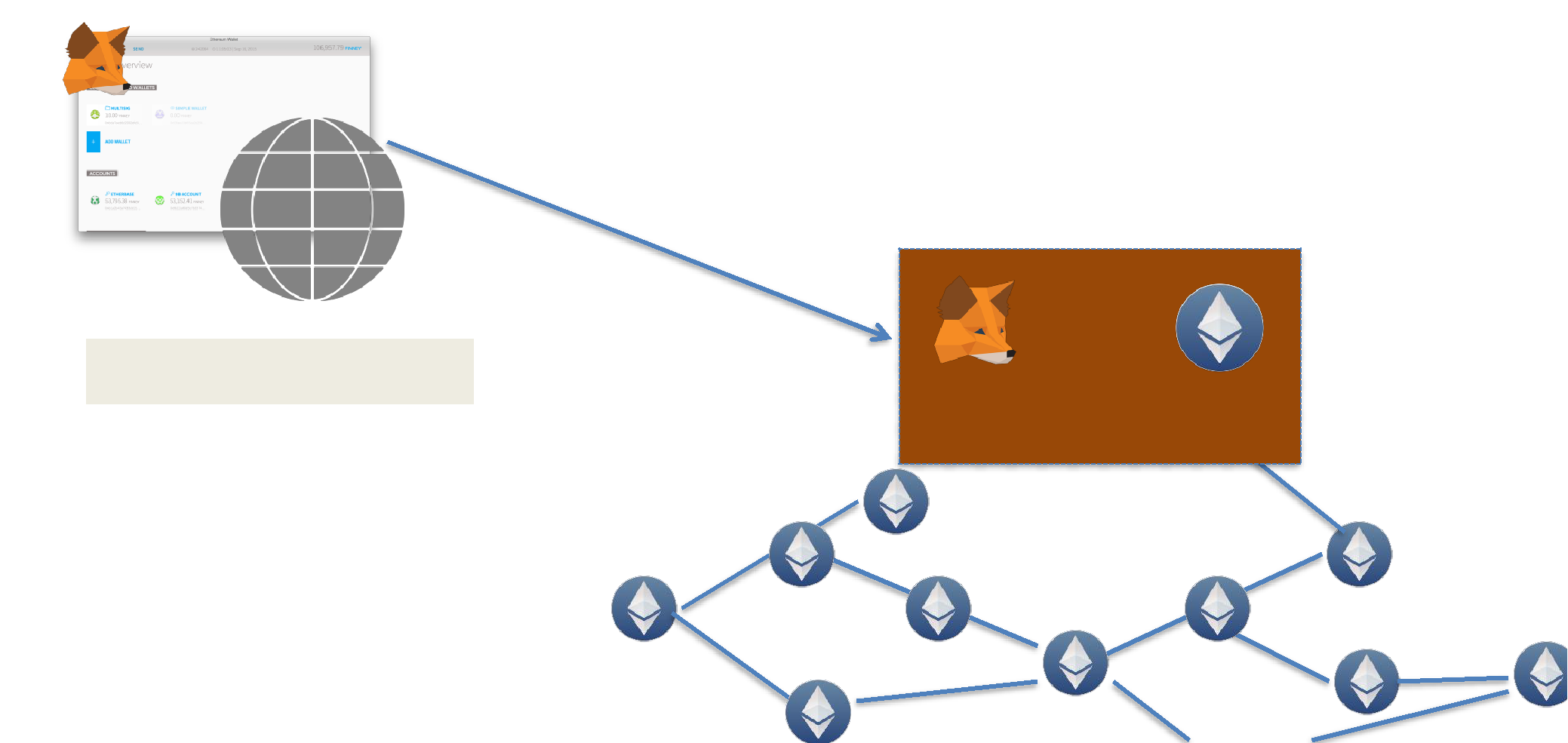
Local Geth

Javascript VM

Memory

Online Wallet • Available at [http://wallet.ethereum.org](http://wallet.ethereum.org/)

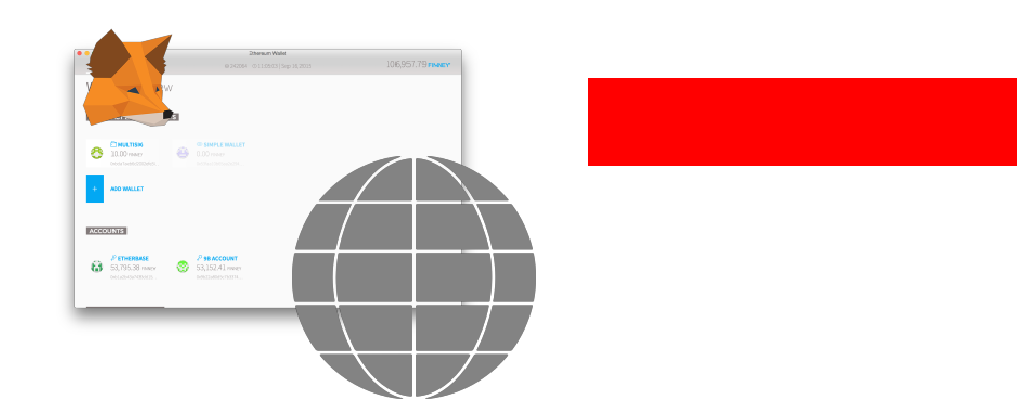
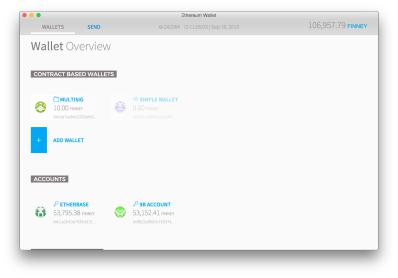
* Accounts managed in *MetaMask*



**Wallet.Ethereum.org**

Node hosted @ MetaMask

Local versus Online Wallet



No mining option

|  |  |  |  |
| --- | --- | --- | --- |
| • Use local node (e.g., geth) | | • Use external hosted node | |
| • | Unavailable till fully synched | • | Available right away |
| • | Keystore managed by app | • | Keystore managed by *MetaMask* |
| • Number of n/w limited | | • Supports many n/w including private | |