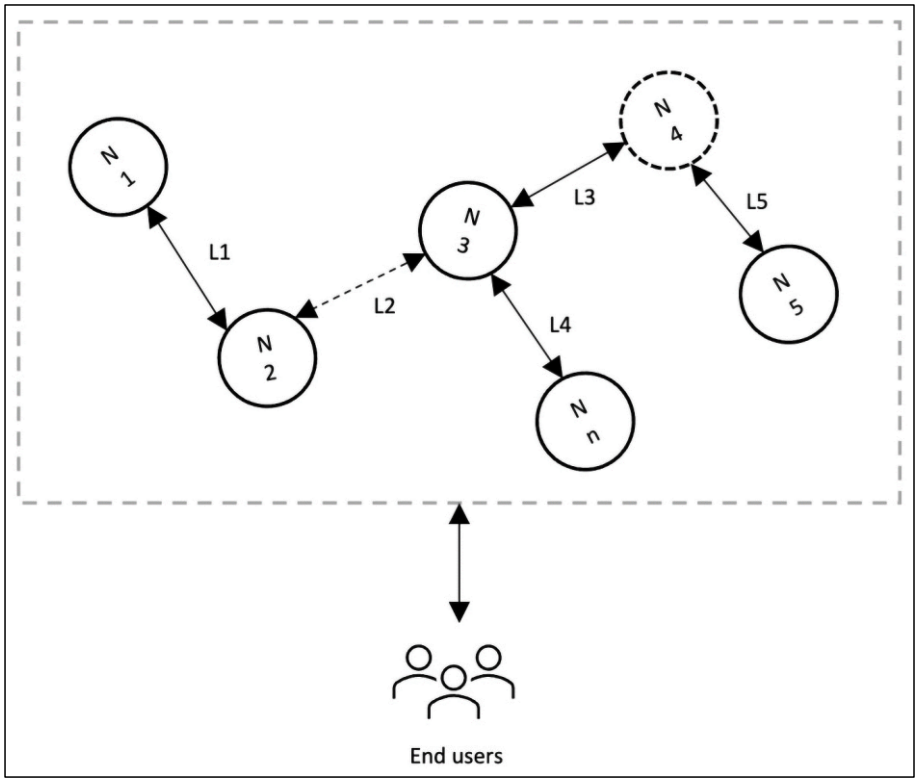
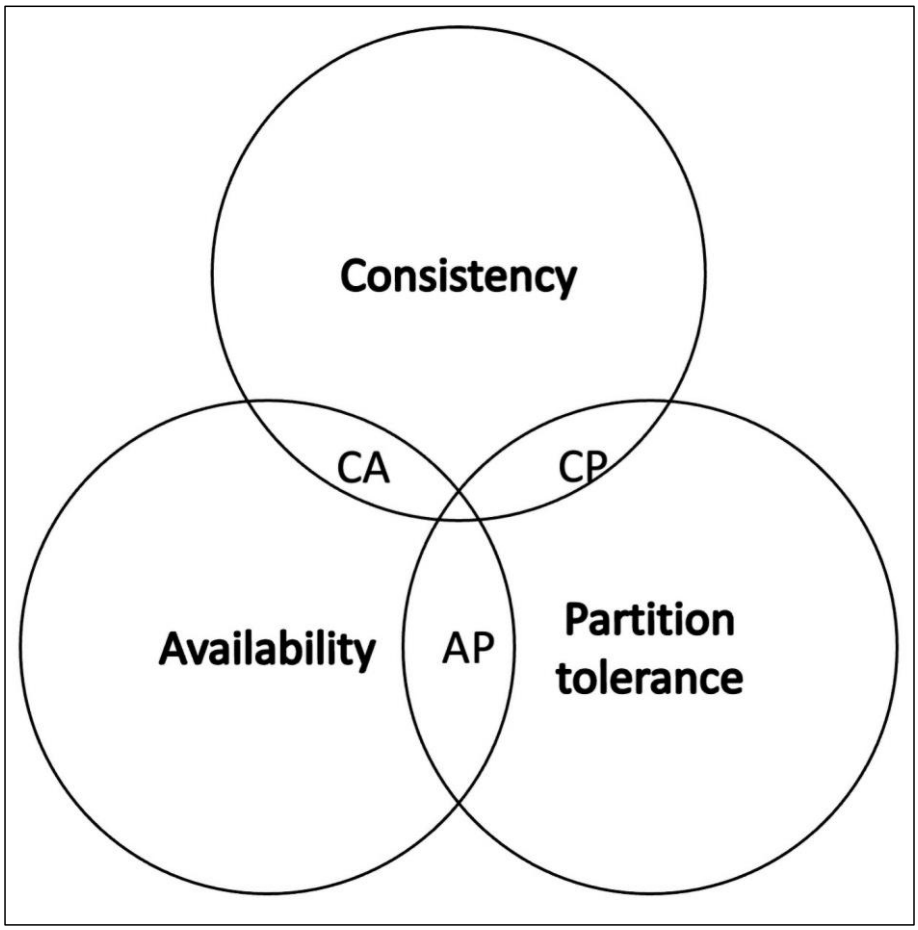
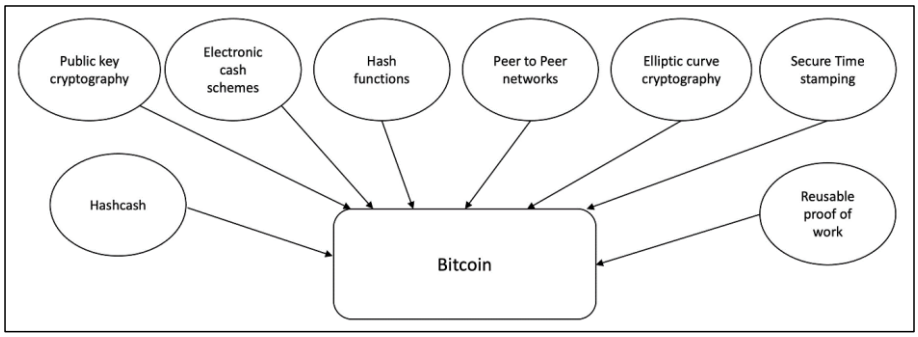
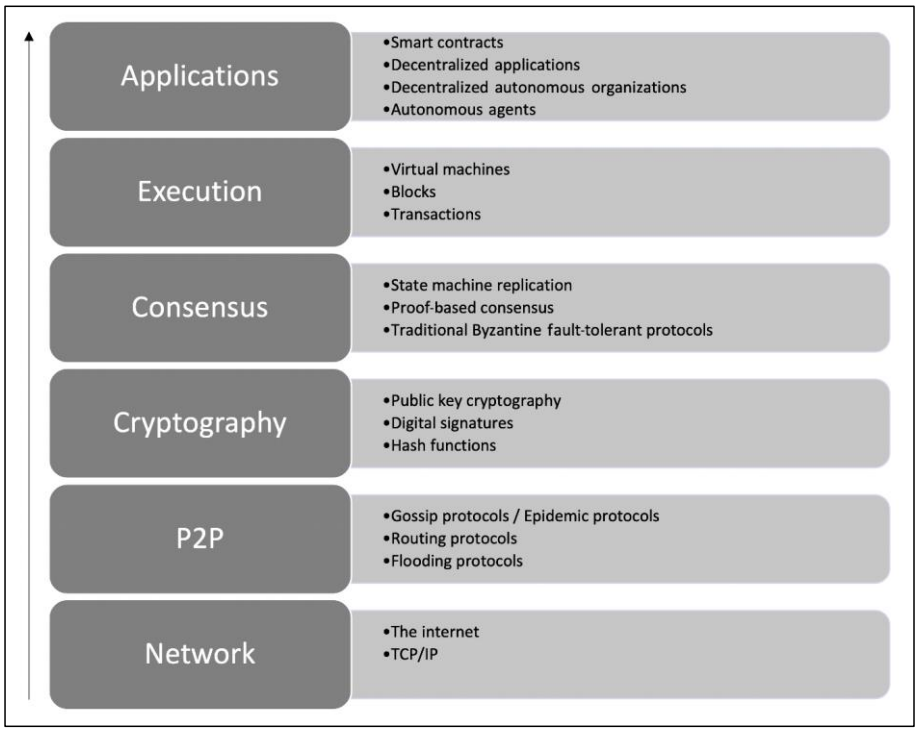
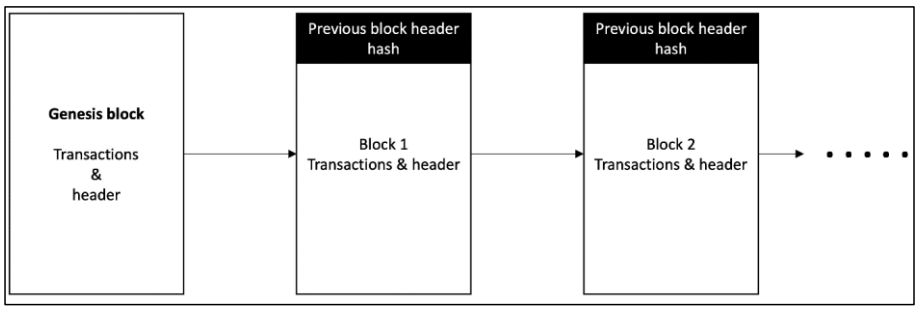
Chapter 1: Blockchain 101

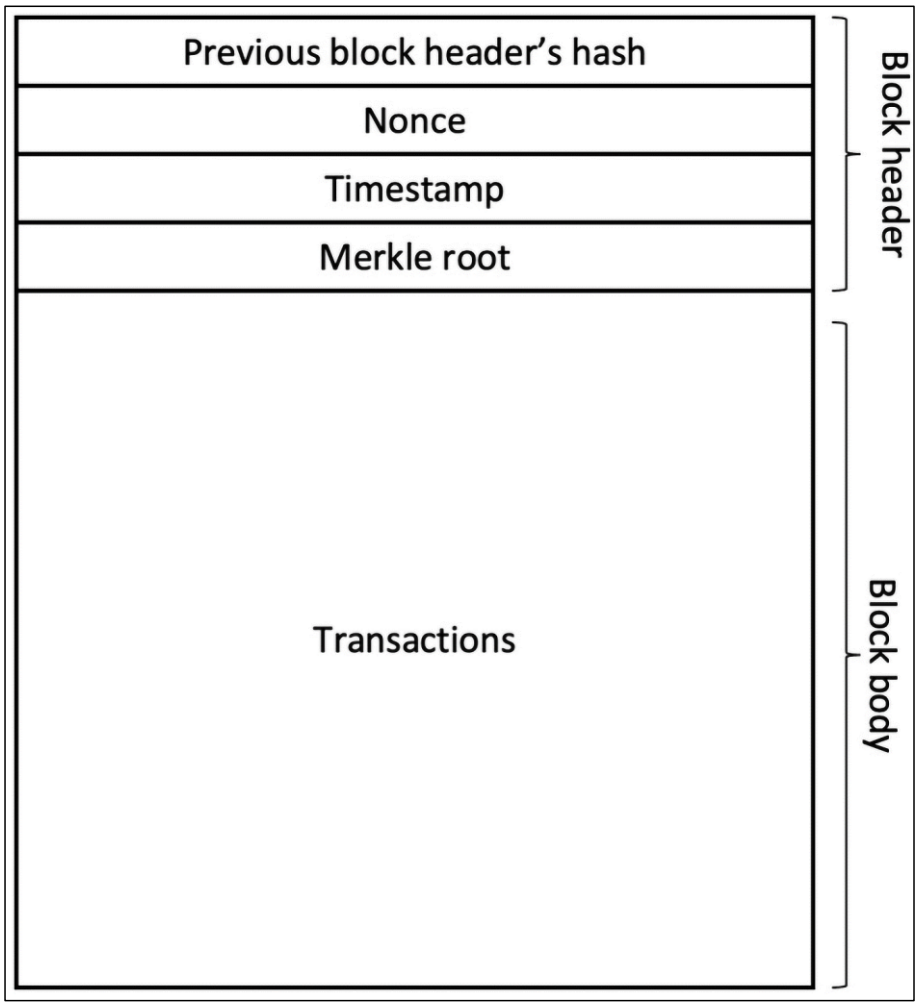


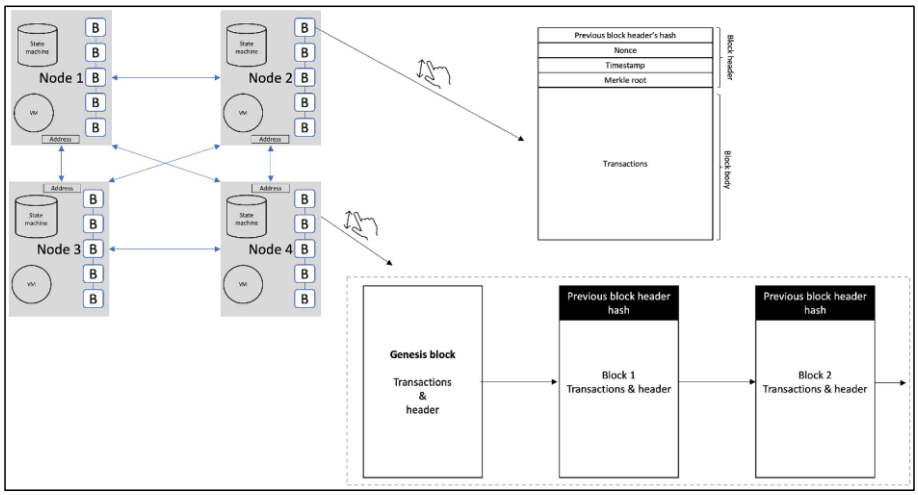

Consistency
CA
CP
Availability
AP
Partition tolerance

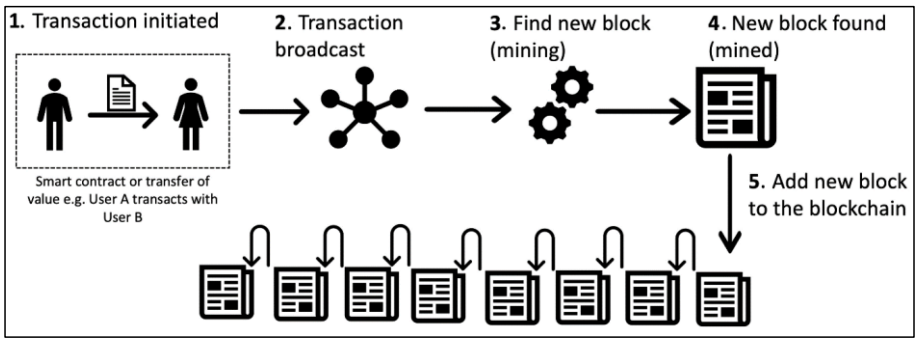

Public key cryptography
Electronic cash schemes
Hash functions
Peer to Peer networks
Elliptic curve cryptography
Secure Time stamping
Hashcash
Reusable proof of work
Bitcoin

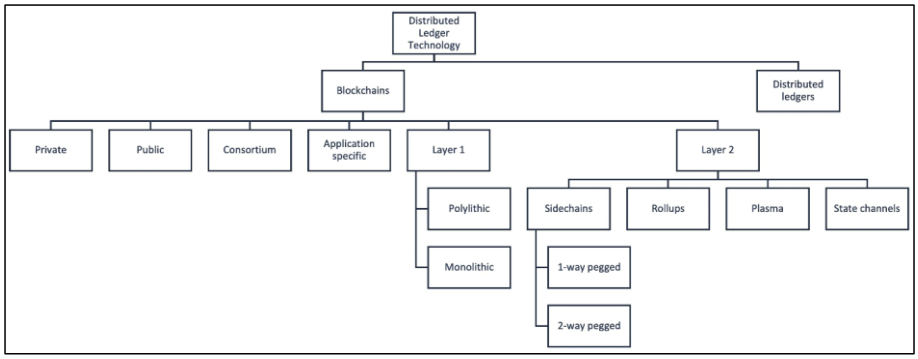

Applications
•Smart contracts
•Decentralized applications
•Decentralized autonomous organizations
•Autonomous agents
Execution
•Virtual machines
•Blocks
•Transactions
Consensus
•State machine replication
•Proof-based consensus
•Traditional Byzantine fault-tolerant protocols
Cryptography
•Public key cryptography
•Digital signatures
•Hash functions
P2P
•Gossip protocols / Epidemic protocols
•Routing protocols
•Flooding protocols
Network
•The internet
•TCP/IP


Genesis block
Transactions
&
header
Previous block header hash
Block 1
Transactions & header
Previous block header hash
Block 2
Transactions & header

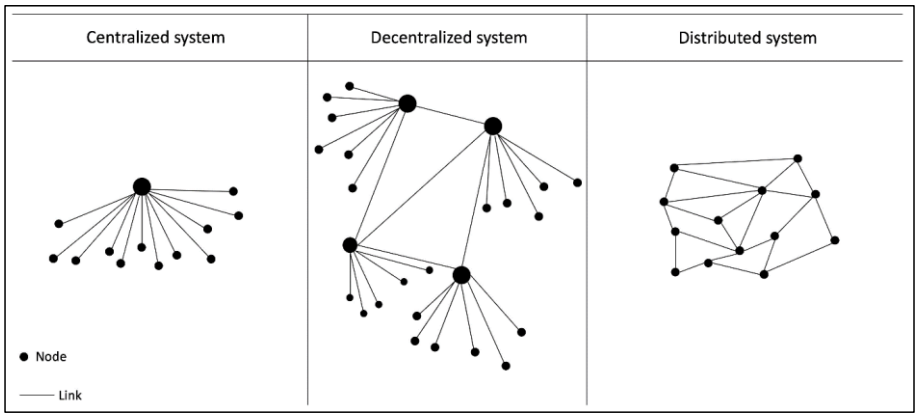

Previous block header's hash
Nonce
Timestamp
Merkle root
Transactions
Block header
Block body

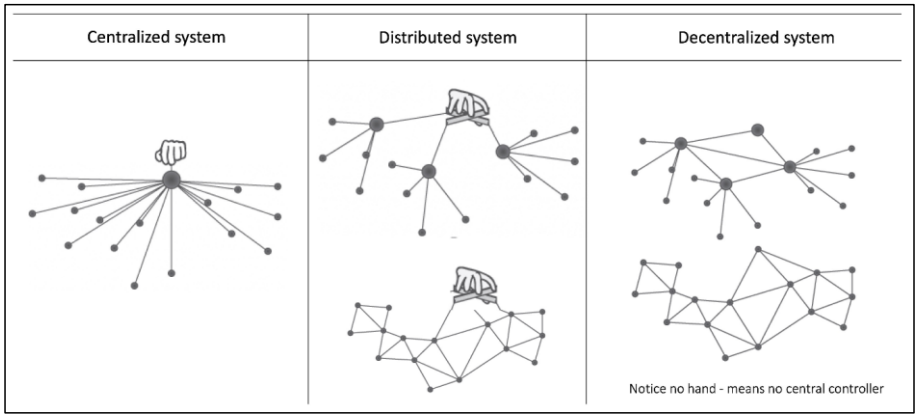

Node 1
Node 2
Node 3
Node 4
State machine
VM
Address
Previous block header's hash
Nonce
Timestamp
Merkle root
Block header
Transactions
Block body
Genesis block
Transactions & header
Previous block header hash
Block 1 Transactions & header
Block 2 Transactions & header

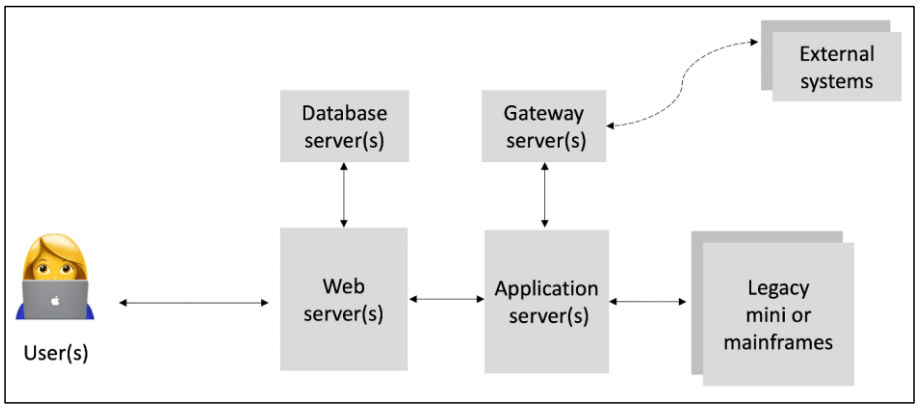

1. Transaction initiated
2. Transaction broadcast
3. Find new block (mining)
4. New block found (mined)
5. Add new block to the blockchain
Smart contract or transfer of value e.g. User A transacts with User B

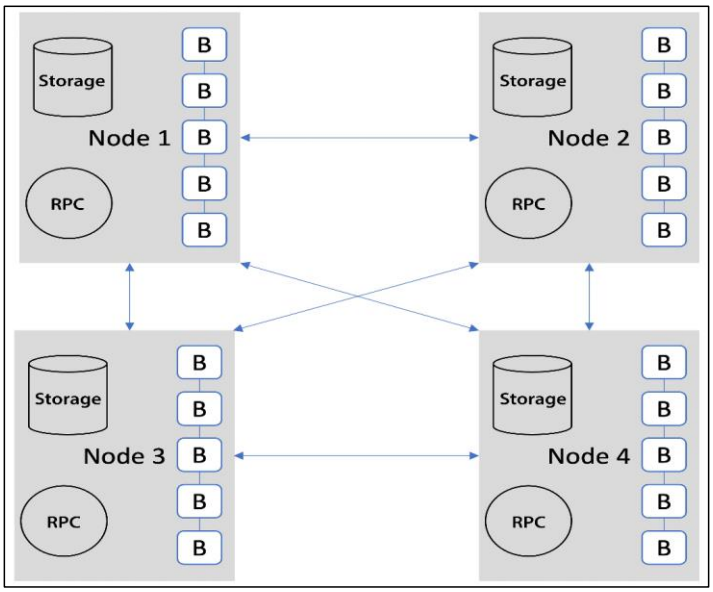

Distributed Ledger Technology
Blockchains
Distributed ledgers
Private
Public
Consortium
Application specific
Layer 1
Layer 2
Polylithic
Monolithic
Sidechains
Rollups
Plasma
State channels
1-way pegged
2-way pegged

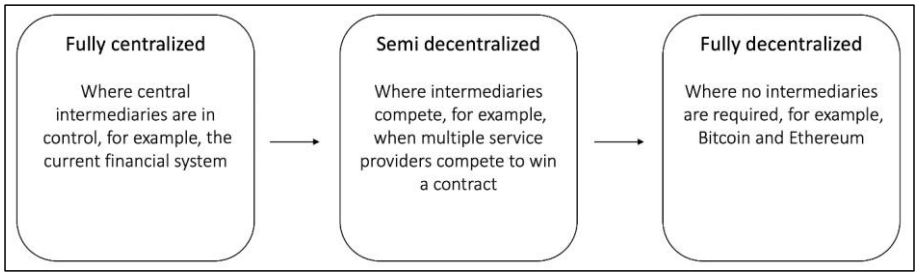
Chapter 2: Decentralization

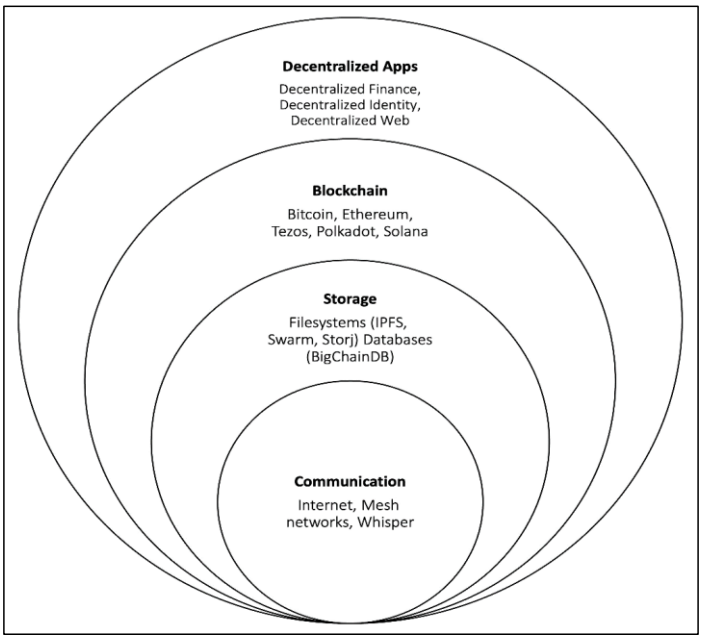


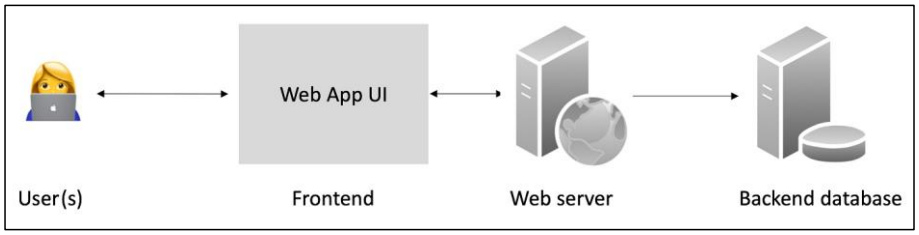


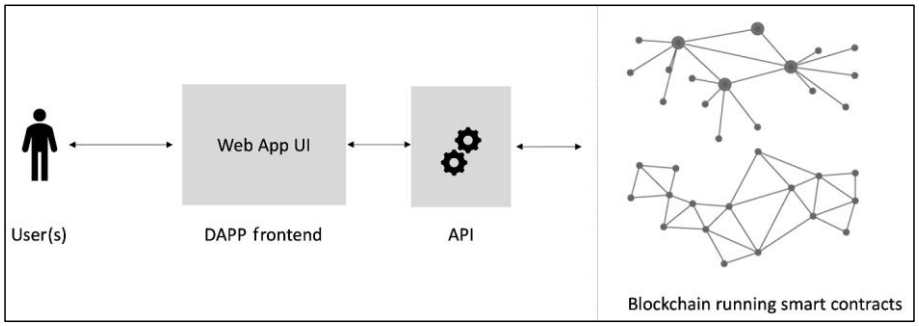



Storage
B
Node 1
RPC
Storage
Node 2
RPC
Storage
Node 3
RPC
Storage
Node 4
RPC

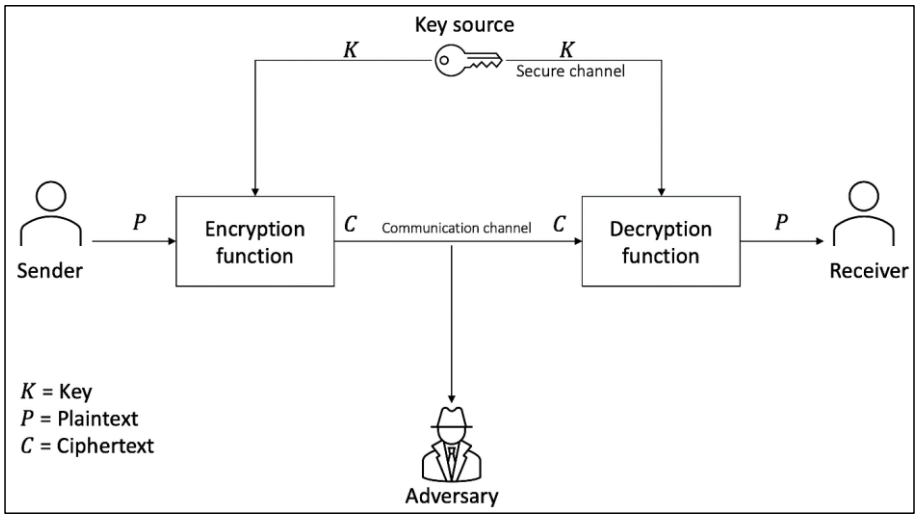

Fully centralized
Where central intermediaries are in control, for example, the current financial system
Semi decentralized
Where intermediaries compete, for example, when multiple service providers compete to win a contract
Fully decentralized
Where no intermediaries are required, for example, Bitcoin and Ethereum

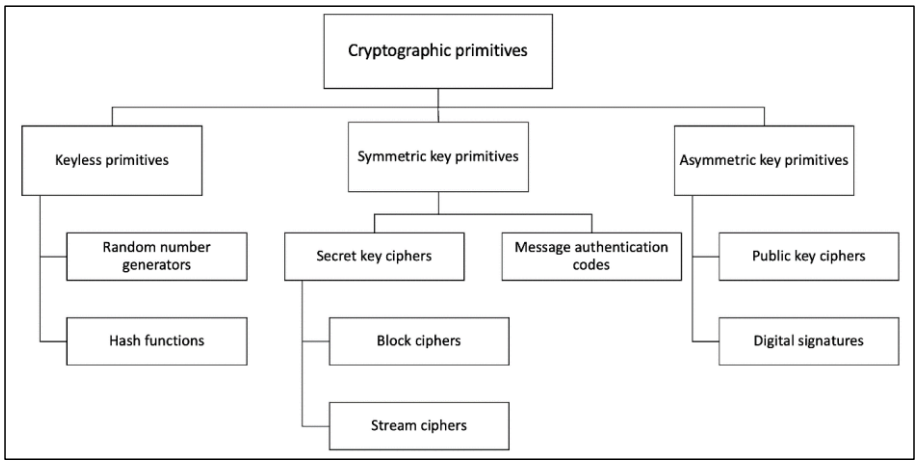

Decentralized Apps
Decentralized Finance,
Decentralized Identity,
Decentralized Web
Blockchain
Bitcoin, Ethereum,
Tezos, Polkadot, Solana
Storage
Filesystems (IPFS,
Swarm, Storj) Databases
(BigChainDB)
Communication
Internet, Mesh
networks, Whisper

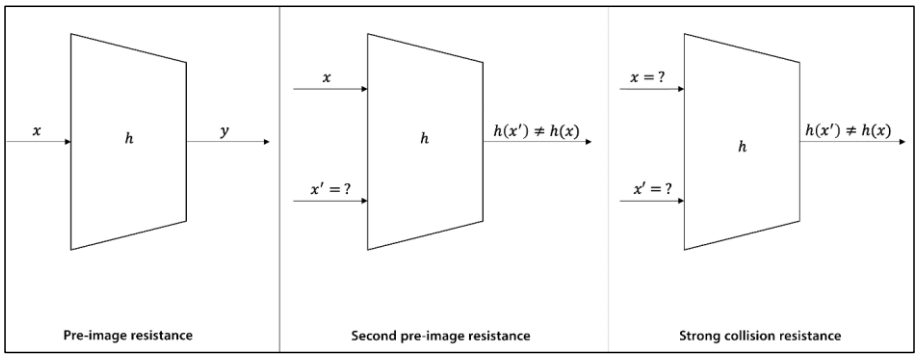

Web App UI
User(s)
Frontend
Web server
Backend database

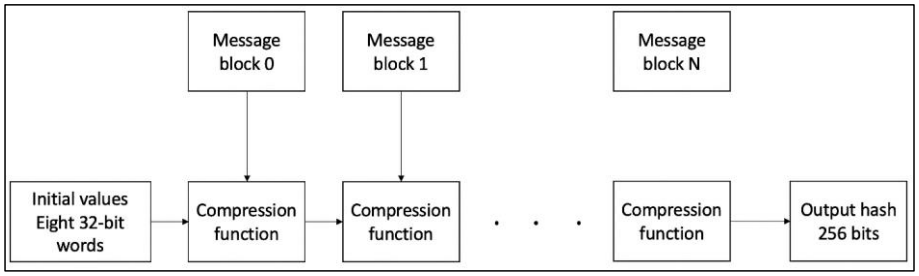

Web App UI
User(s)
DAPP frontend
API
Blockchain running smart contracts

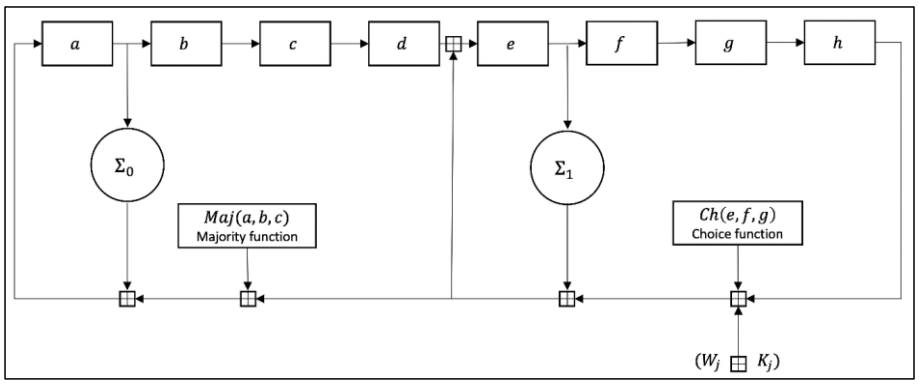
Chapter 3: Symmetric Cryptography

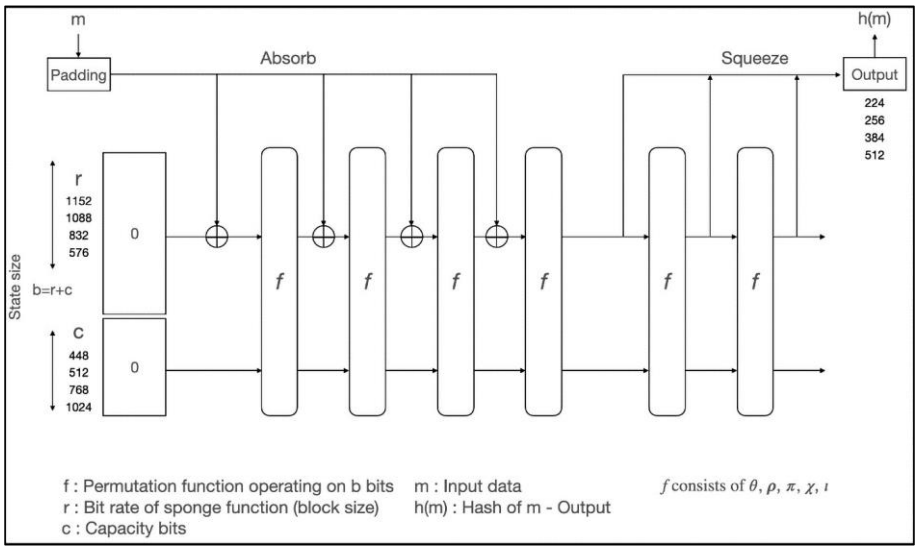


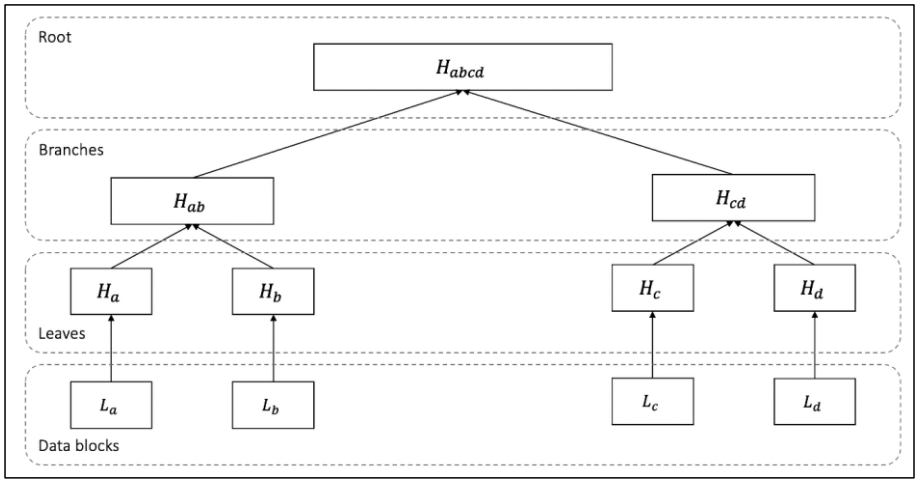


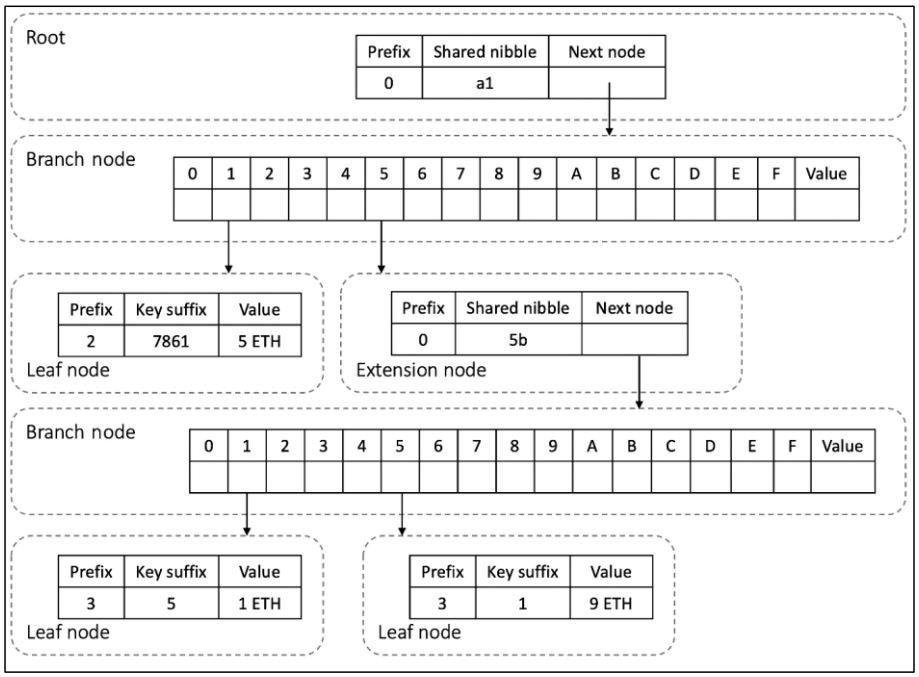

x
h
y
Pre-image resistance
x
x′ = ?
h
h(x′) ≠ h(x)
Second pre-image resistance
x = ?
x′ = ?
h
h(x′) ≠ h(x)
Strong collision resistance

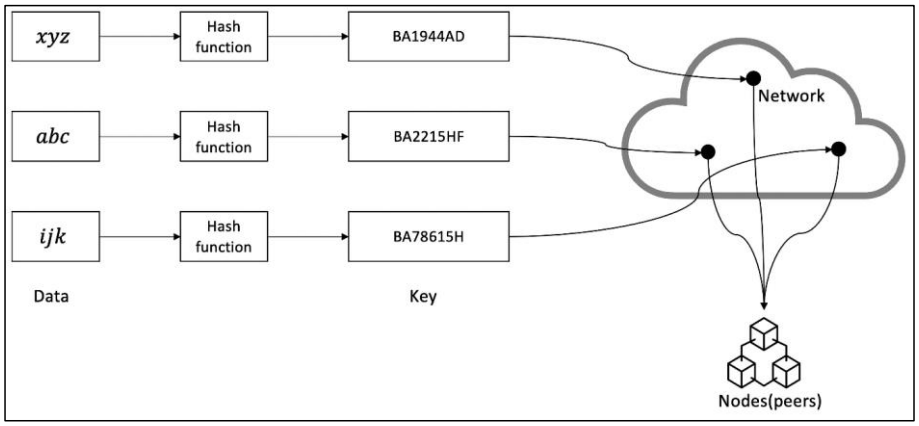

Message block 0
Message block 1
Message block N
Initial values Eight 32-bit words
Compression function
Compression function
Compression function
Output hash 256 bits

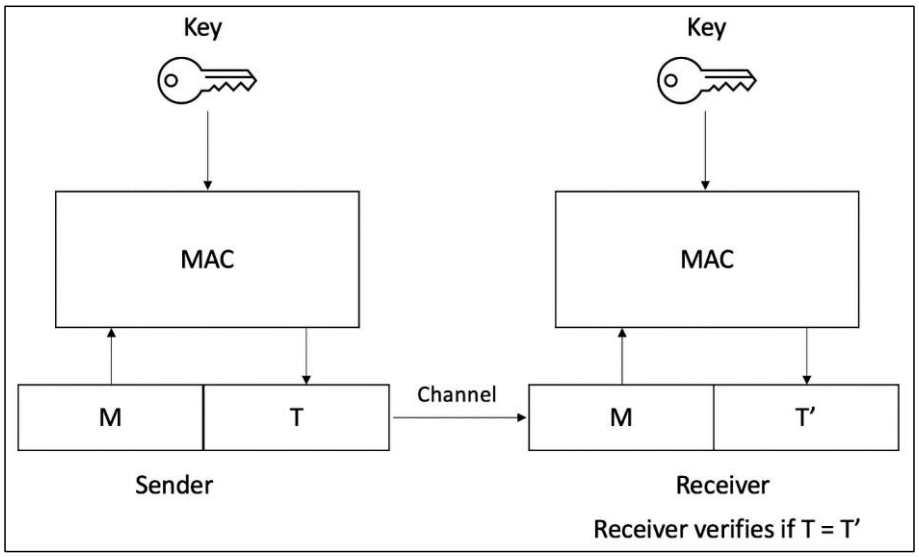

a
b
c
d
e
f
g
h
Σ0
Σ1
Maj(a, b, c)
Majority function
Ch(e, f, g)
Choice function
(Wj ⊞ Kj)

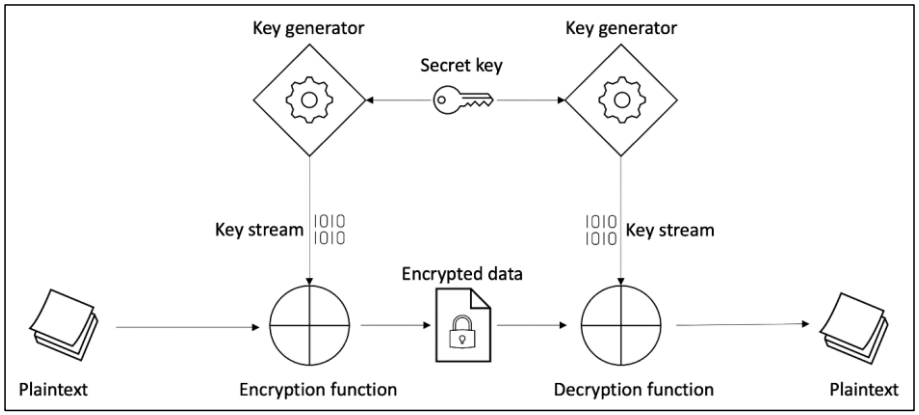

m
Padding
Absorb
Squeeze
h(m)
Output
224
256
384
512
r
1152
1088
832
576
b=r+c
c
448
512
768
1024
State size
0
0
f
f
f
f
f
f
f : Permutation function operating on b bits
r : Bit rate of sponge function (block size)
c : Capacity bits
m : Input data
h(m) : Hash of m - Output
f consists of θ, ρ, π, χ, ι

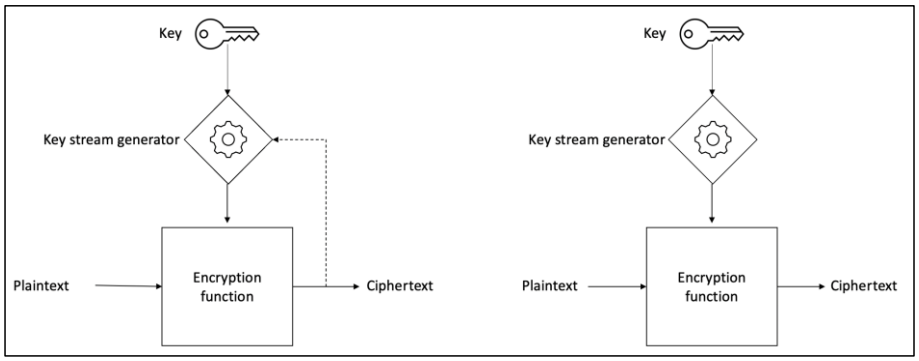

Root
H_{abcd}
Branches
H_{ab}
H_{cd}
H_a
H_b
H_c
H_d
Leaves
L_a
L_b
L_c
L_d
Data blocks

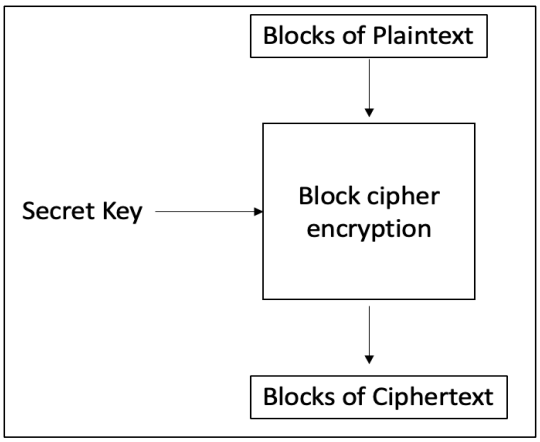

Root
Prefix
Shared nibble
Next node
0
a1
Branch node
0 1 2 3 4 5 6 7 8 9 A B C D E F Value
Prefix
Key suffix
Value
2
7861
5 ETH
Leaf node
Prefix
Shared nibble
Next node
0
5b
Extension node
Branch node
0 1 2 3 4 5 6 7 8 9 A B C D E F Value
Prefix
Key suffix
Value
3
5
1 ETH
Leaf node
Prefix
Key suffix
Value
3
1
9 ETH
Leaf node

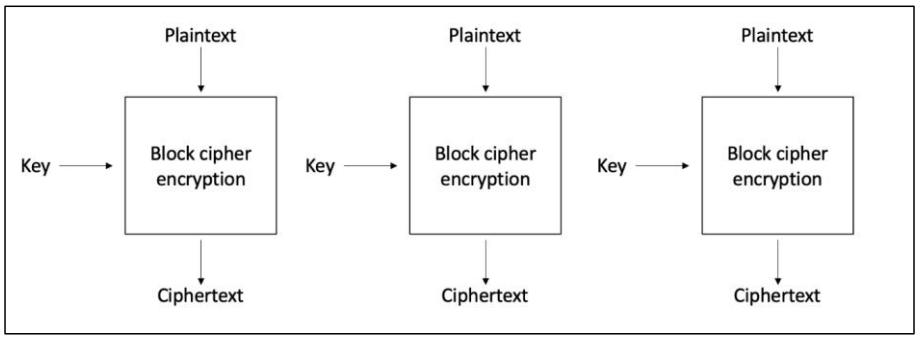

xyz
Hash function
BA1944AD
abc
Hash function
BA2215HF
ijk
Hash function
BA78615H
Data
Key
Network
Nodes(peers)

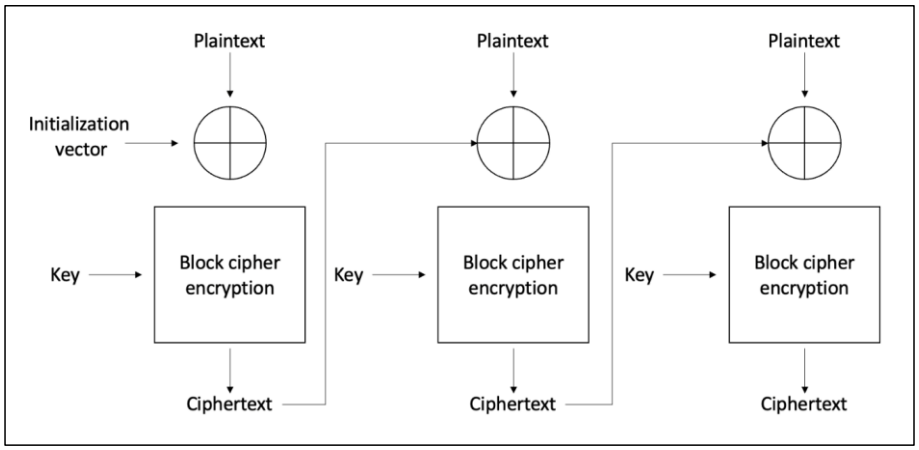

Key
Key
MAC
MAC
M
T
Channel
M
T'
Sender
Receiver
Receiver verifies if T = T'

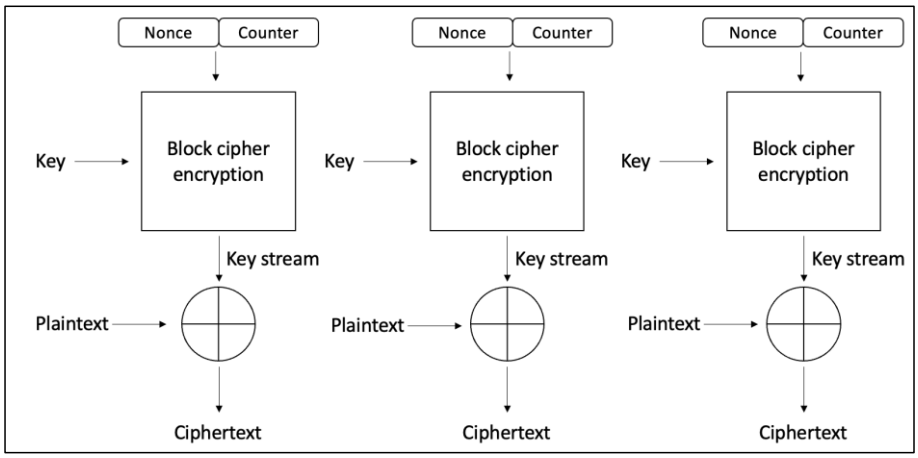

Key generator
Key generator
Secret key
Key stream
Key stream
Encrypted data
Plaintext
Encryption function
Decryption function
Plaintext

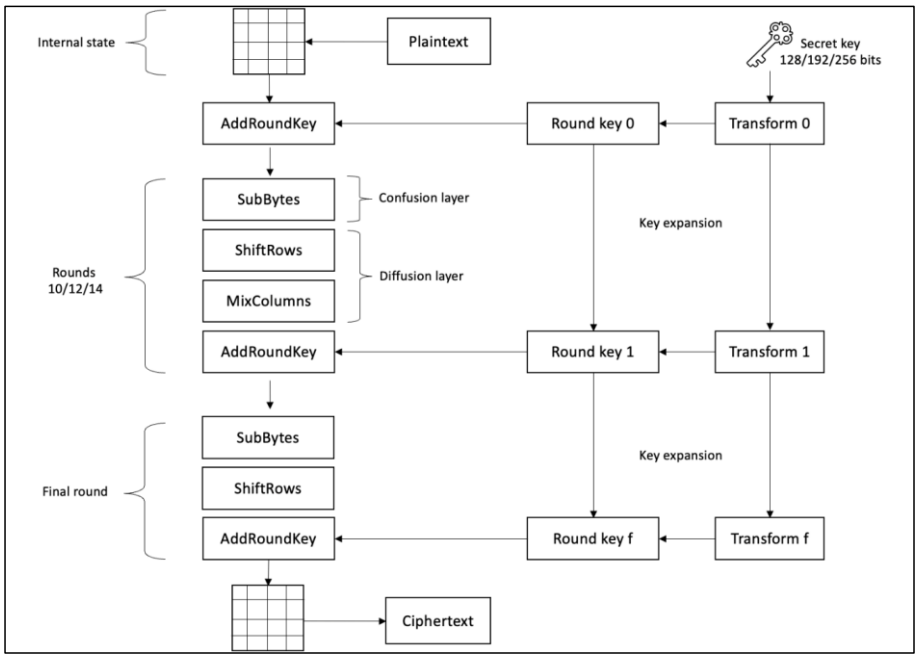

Key
Key
Key stream generator
Key stream generator
Plaintext
Encryption function
Ciphertext
Plaintext
Encryption function
Ciphertext


Blocks of Plaintext
Secret Key
Block cipher encryption
Blocks of Ciphertext

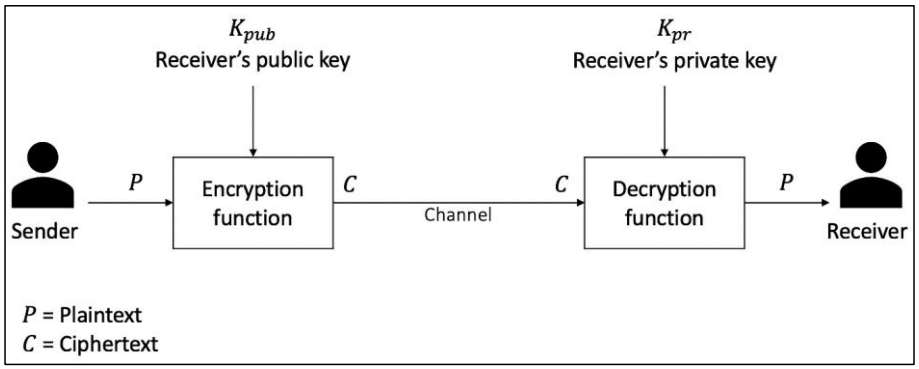

Plaintext
Plaintext
Plaintext
Key
Block cipher encryption
Key
Block cipher encryption
Key
Block cipher encryption
Ciphertext
Ciphertext
Ciphertext

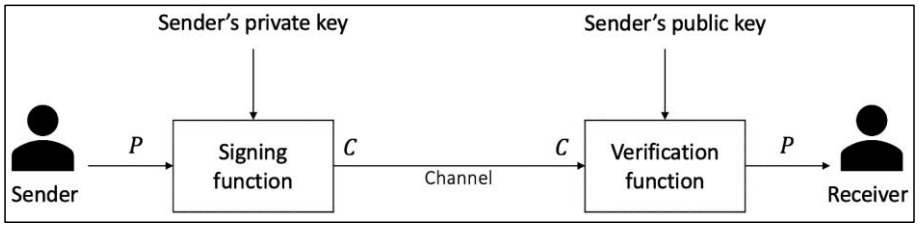

Plaintext
Plaintext
Plaintext
Initialization vector
Key
Block cipher encryption
Key
Block cipher encryption
Key
Block cipher encryption
Ciphertext
Ciphertext
Ciphertext

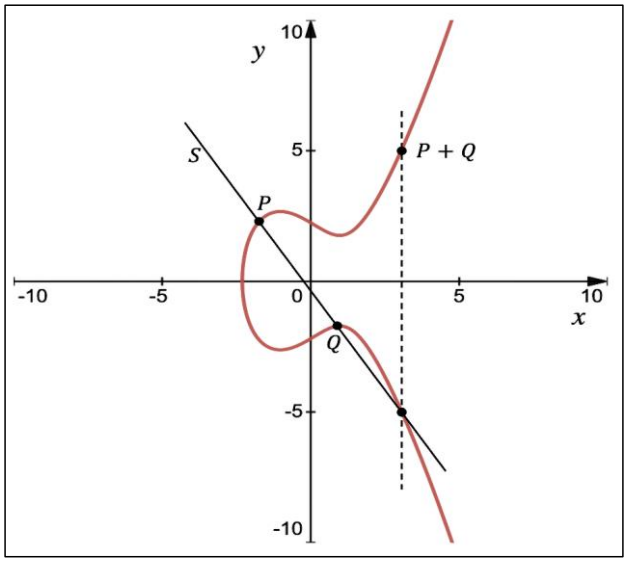

Nonce
Counter
Nonce
Counter
Nonce
Counter
Key
Block cipher encryption
Key
Block cipher encryption
Key
Block cipher encryption
Key stream
Key stream
Key stream
Plaintext
Plaintext
Plaintext
Ciphertext
Ciphertext
Ciphertext

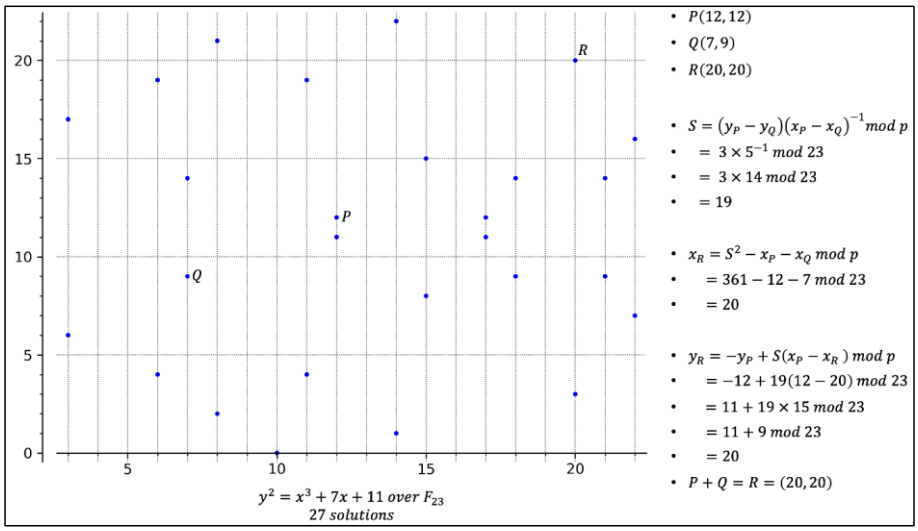

Internal state
Plaintext
Secret key
128/192/256 bits
AddRoundKey
Round key 0
Transform 0
SubBytes
Confusion layer
Key expansion
ShiftRows
Diffusion layer
Rounds
10/12/14
MixColumns
AddRoundKey
Round key 1
Transform 1
SubBytes
Key expansion
Final round
ShiftRows
AddRoundKey
Round key f
Transform f
Ciphertext

Chapter 4: Asymmetric Cryptography







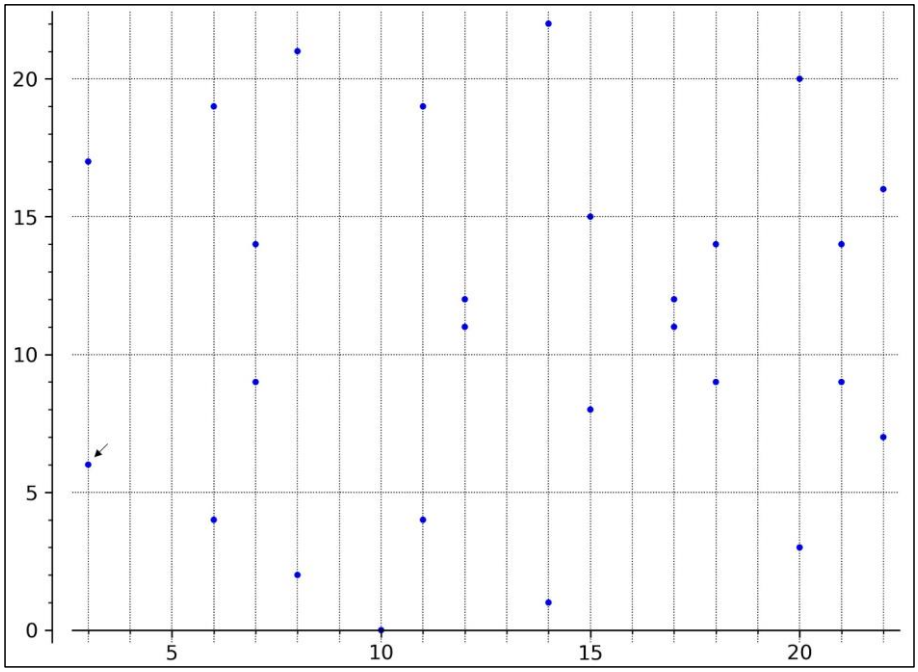


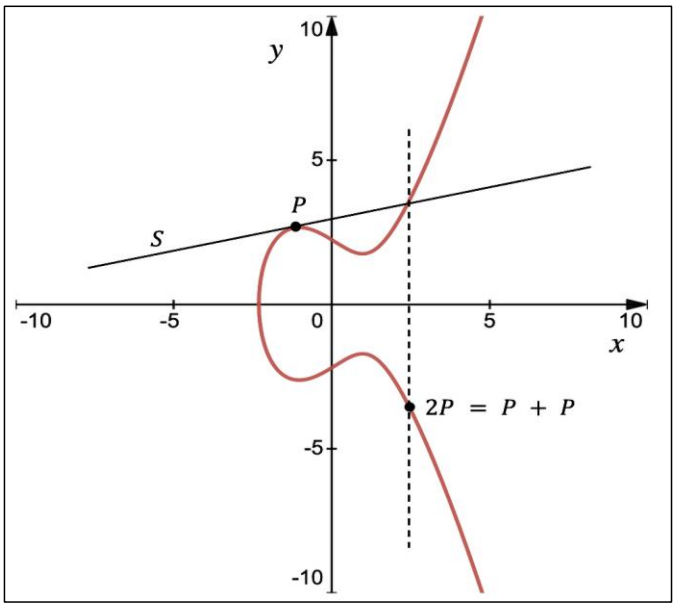
- $P(12, 12)$
- $Q(7, 9)$
- $R(20, 20)$

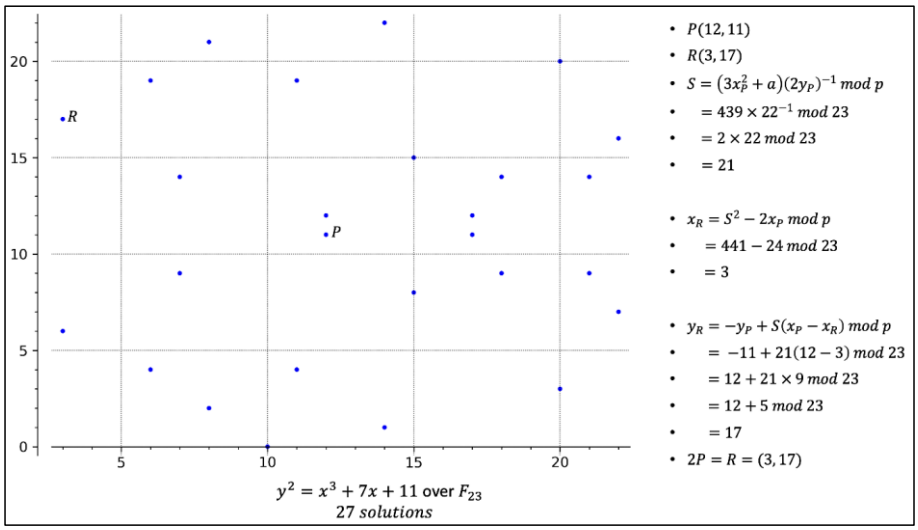
- $S = (y_P - y_Q)(x_P - x_Q)^{-1} mod\ p$
- $= 3 \times 5^{-1}\ mod\ 23$
- $= 3 \times 14\ mod\ 23$
- $= 19$

- $x_R = S^2 - x_P - x_Q\ mod\ p$
- $= 361 - 12 - 7\ mod\ 23$
- $= 20$

- $y_R = -y_P + S(x_P - x_R)\ mod\ p$
- $= -12 + 19(12 - 20)\ mod\ 23$
- $= 11 + 19 \times 15\ mod\ 23$
- $= 11 + 9\ mod\ 23$
- $= 20$
- $P + Q = R = (20, 20)$



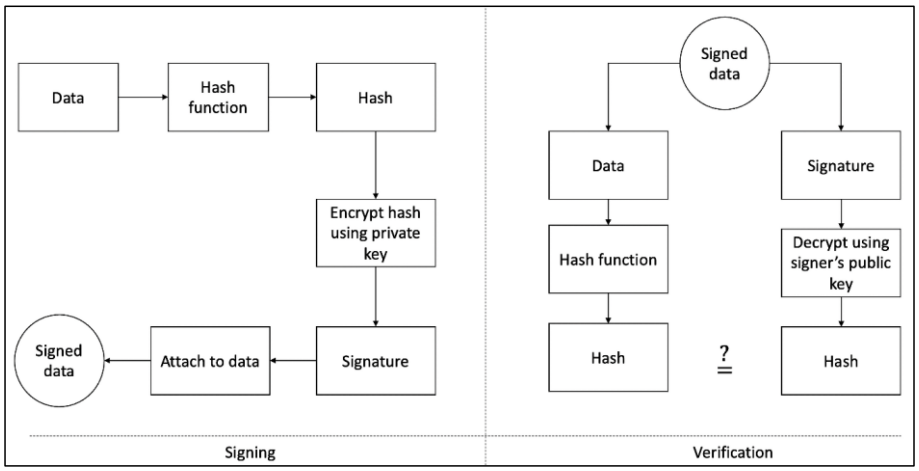


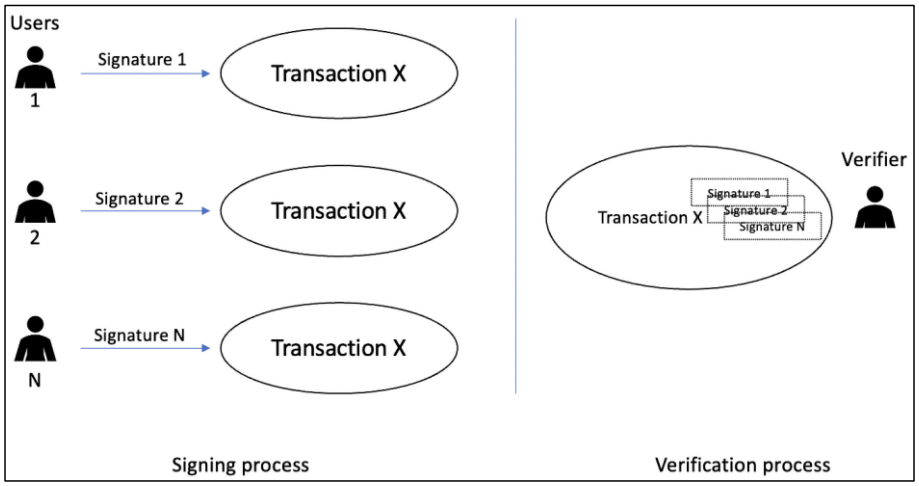


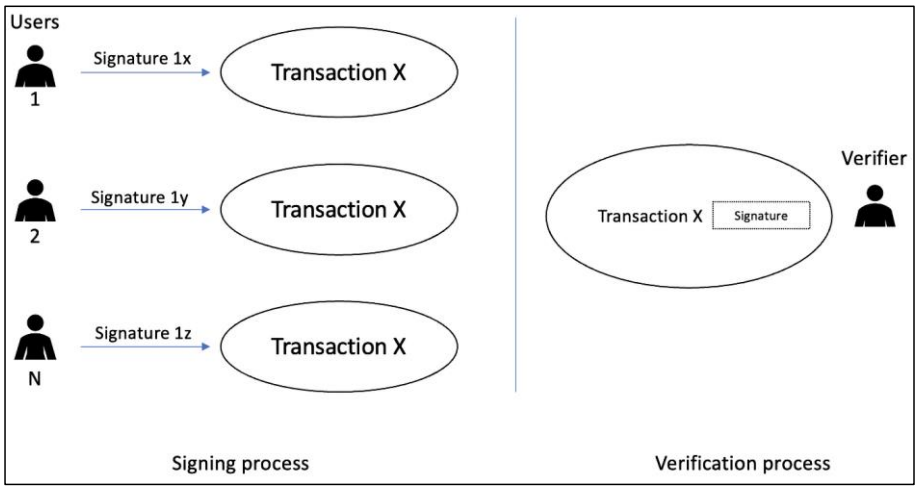
- $P(12, 11)$
- $R(3, 17)$
- $S = (3x_P^2 + a)(2y_P)^{-1} \bmod p$
- $= 439 \times 22^{-1} \bmod 23$
- $= 2 \times 22 \bmod 23$
- $= 21$

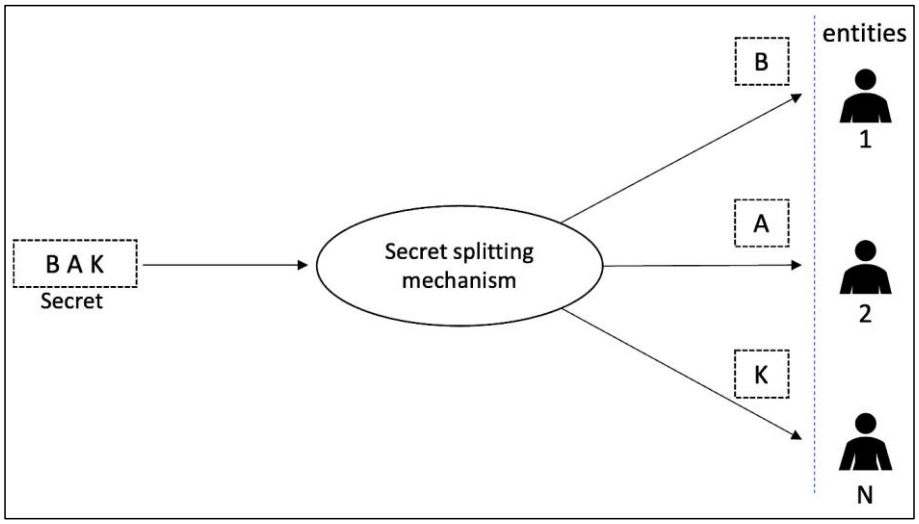
- $x_R = S^2 - 2x_P \bmod p$
- $= 441 - 24 \bmod 23$
- $= 3$

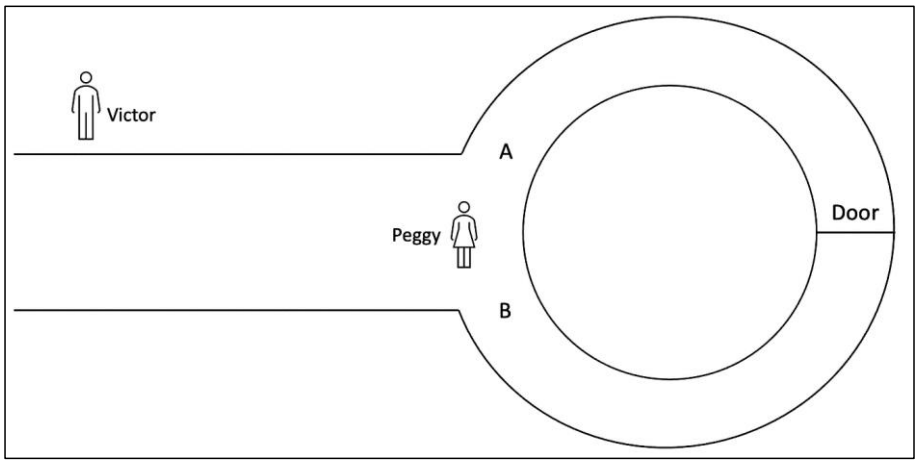
- $y_R = -y_P + S(x_P - x_R) \bmod p$
- $= -11 + 21(12 - 3) \bmod 23$
- $= 12 + 21 \times 9 \bmod 23$
- $= 12 + 5 \bmod 23$
- $= 17$
- $2P = R = (3, 17)$

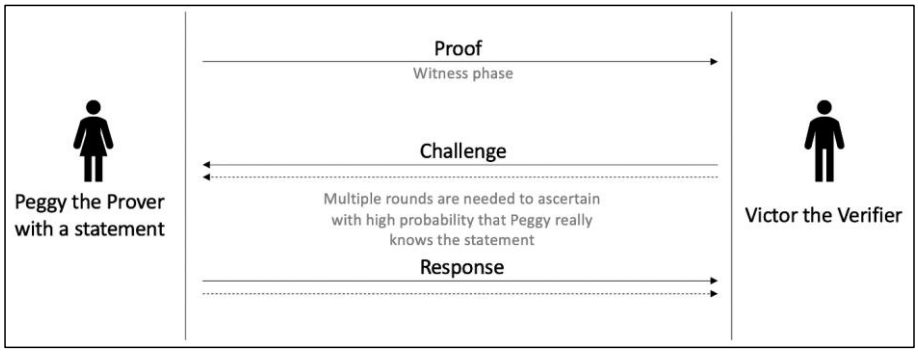

Data
Hash function
Hash
Encrypt hash using private key
Signature
Attach to data
Signed data
Signing
Signed data
Data
Hash function
Hash
?
=
Signature
Decrypt using signer's public key
Hash
Verification


Users
1
Signature 1
Transaction X
2
Signature 2
Transaction X
N
Signature N
Transaction X
Signing process
Transaction X
Signature 1
Signature 2
Signature N
Verifier
Verification process


Users
Signature 1x
Transaction X
1
Signature 1y
Transaction X
2
Signature 1z
Transaction X
N
Verifier
Transaction X
Signature
Signing process
Verification process

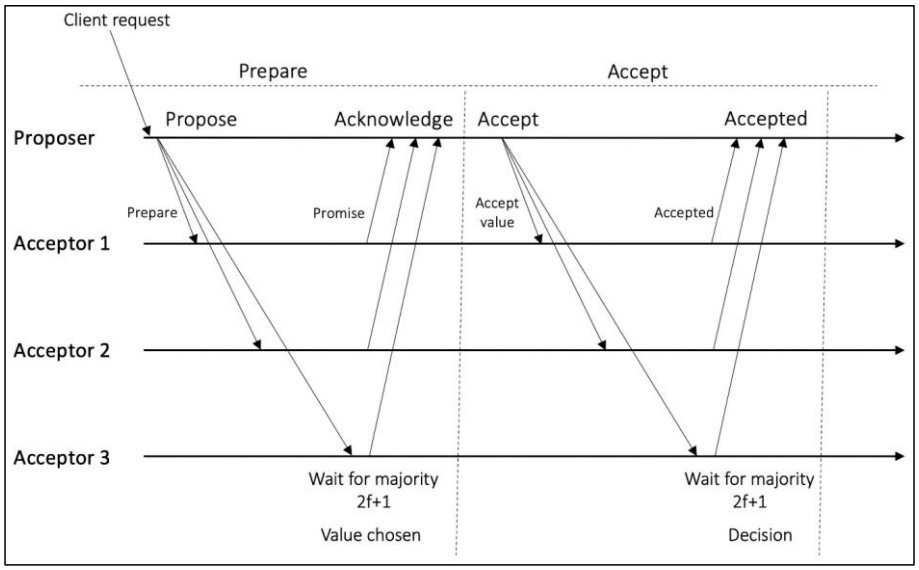

entities
B
1
A
2
K
N
B A K
Secret
Secret splitting mechanism

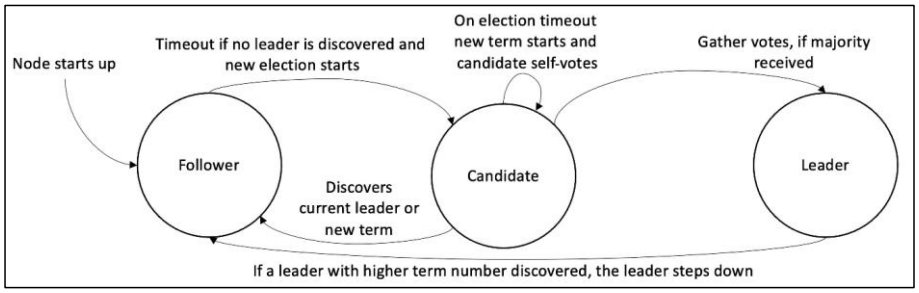

Victor
A
Peggy
Door
B

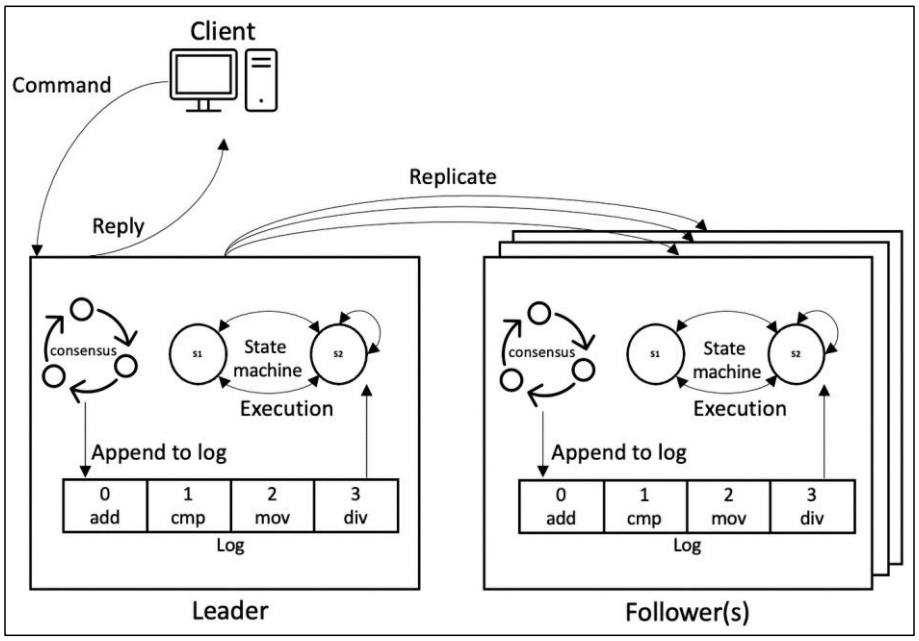

Proof
Witness phase
Challenge
Multiple rounds are needed to ascertain
with high probability that Peggy really
knows the statement
Response
Peggy the Prover
with a statement
Victor the Verifier

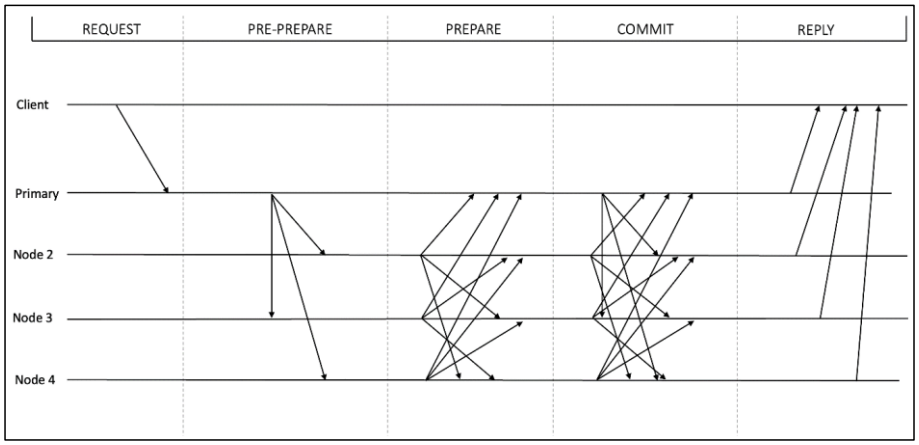

Computation
Arithmetic circuit
R1CS
QAP
ZK-SNARK

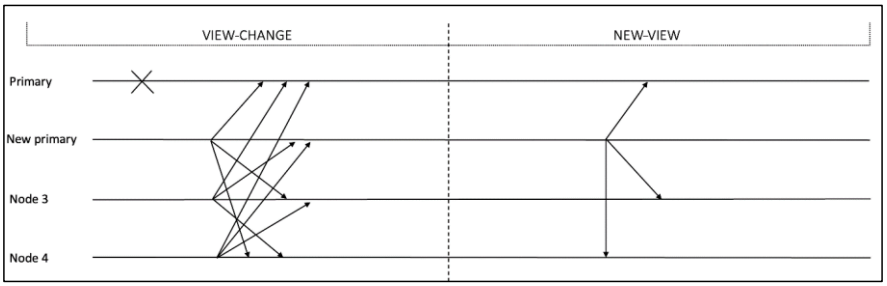
Chapter 5: Consensus Algorithms

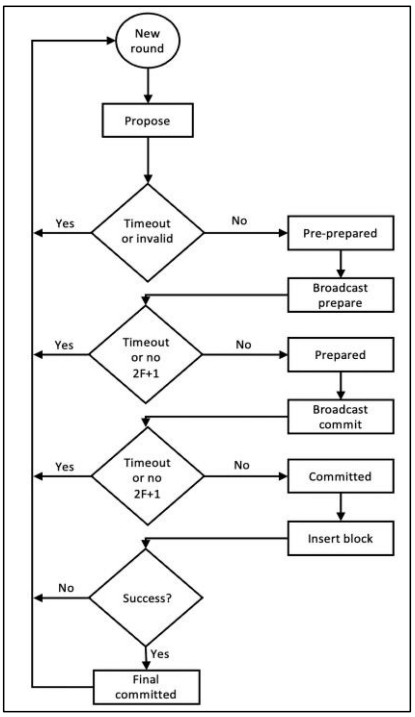


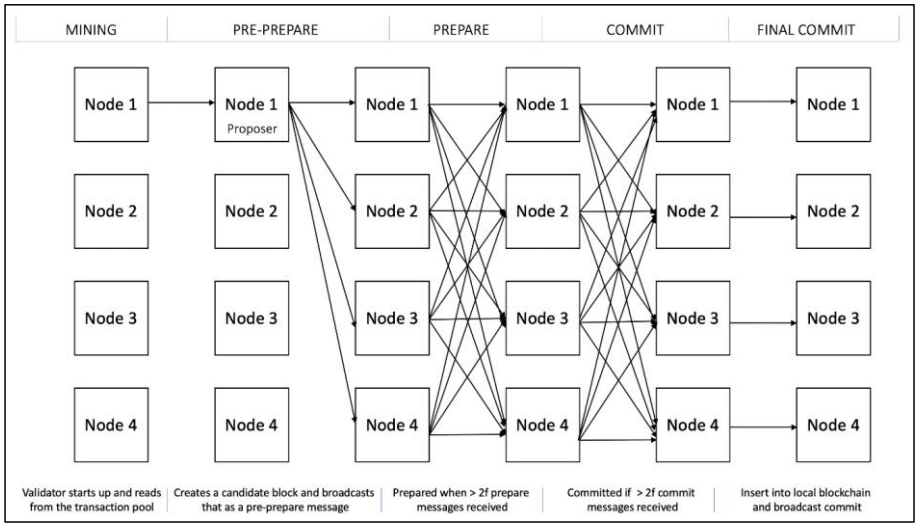


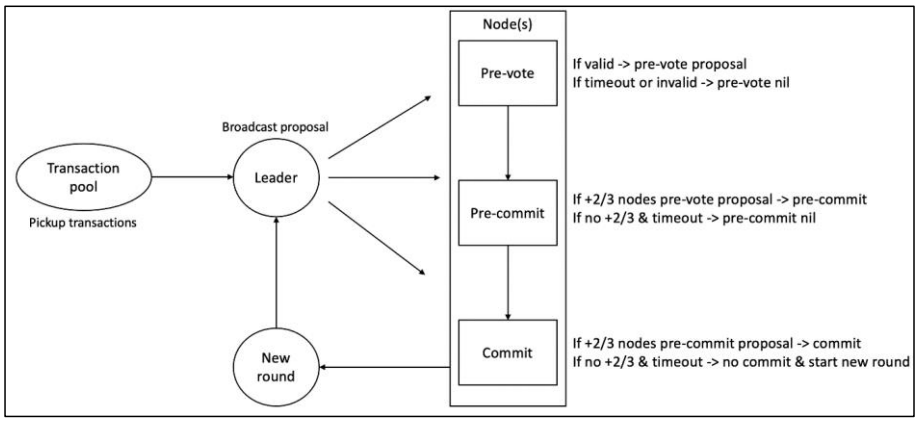

Client
Command
Reply
Replicate
consensus
S1
State machine
S2
Execution
Append to log
0 add
1 cmp
2 mov
3 div
Log
Leader
Follower(s)

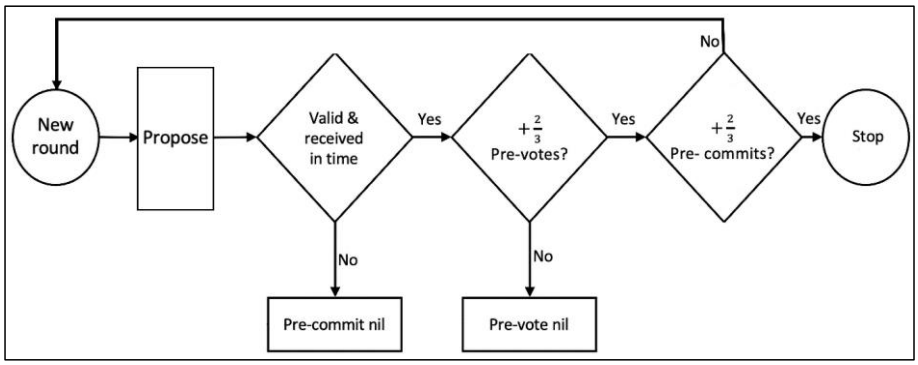

REQUEST
PRE-PREPARE
PREPARE
COMMIT
REPLY
Client
Primary
Node 2
Node 3
Node 4

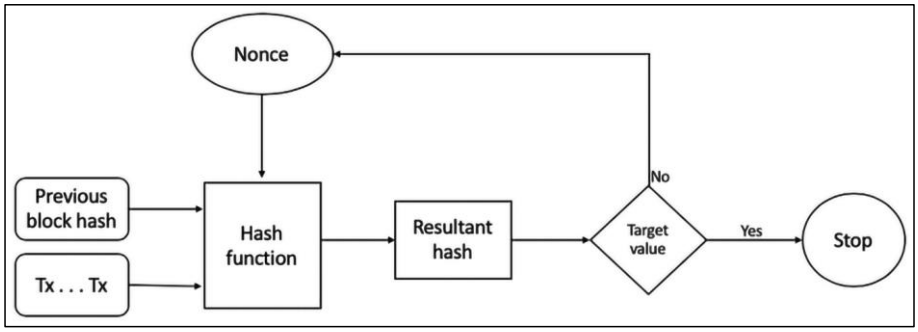

VIEW-CHANGE
NEW-VIEW
Primary
New primary
Node 3
Node 4

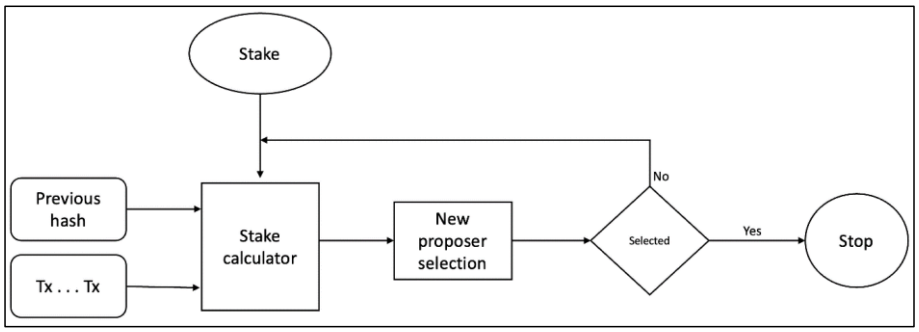

New round
Propose
Timeout or invalid
Yes
No
Pre-prepared
Broadcast prepare
Timeout or no 2F+1
Yes
No
Prepared
Broadcast commit
Timeout or no 2F+1
Yes
No
Committed
Insert block
Success?
No
Yes
Final committed

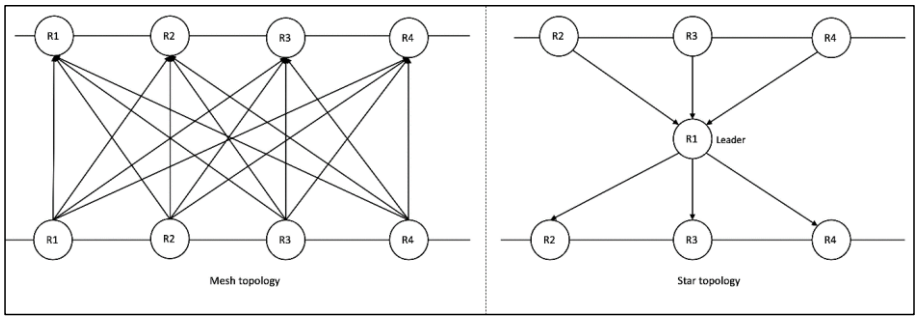

MINING
PRE-PREPARE
PREPARE
COMMIT
FINAL COMMIT
Node 1
Node 2
Node 3
Node 4
Node 1
Proposer
Validator starts up and reads from the transaction pool
Creates a candidate block and broadcasts that as a pre-prepare message
Prepared when > 2f prepare messages received
Committed if > 2f commit messages received
Insert into local blockchain and broadcast commit

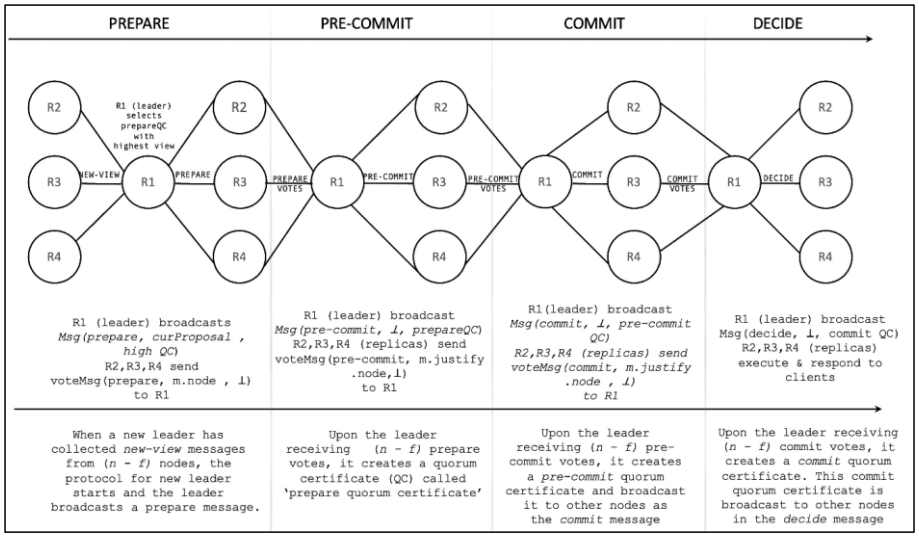

Node(s)
Pre-vote
If valid -> pre-vote proposal
If timeout or invalid -> pre-vote nil
Broadcast proposal
Transaction pool
Leader
Pickup transactions
Pre-commit
If +2/3 nodes pre-vote proposal -> pre-commit
If no +2/3 & timeout -> pre-commit nil
New round
Commit
If +2/3 nodes pre-commit proposal -> commit
If no +2/3 & timeout -> no commit & start new round


New round
Propose
Valid & received in time
Yes
+2/3 Pre-votes?
Yes
+2/3 Pre- commits?
Yes
Stop
No
No
No
Pre-commit nil
Pre-vote nil

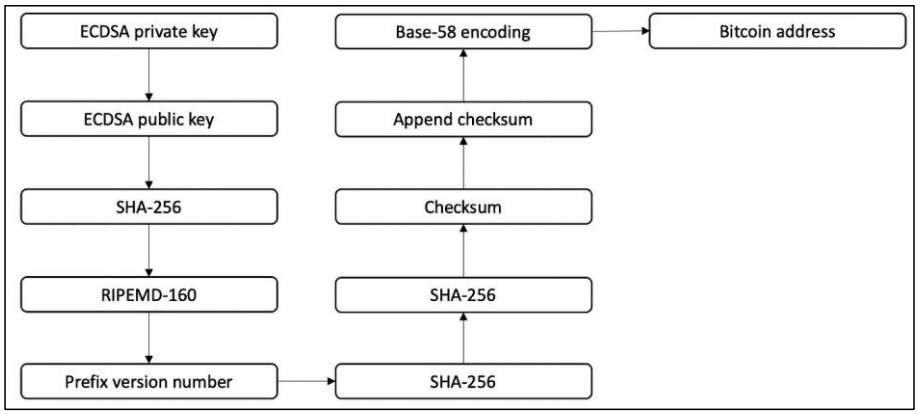

Nonce
Previous block hash
Tx . . . Tx
Hash function
Resultant hash
Target value
No
Yes
Stop

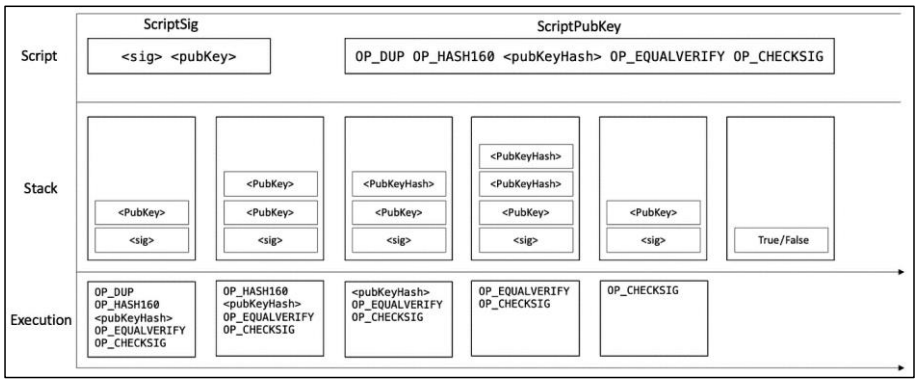

Stake
Previous hash
Tx . . . Tx
Stake calculator
New proposer selection
Selected
No
Yes
Stop

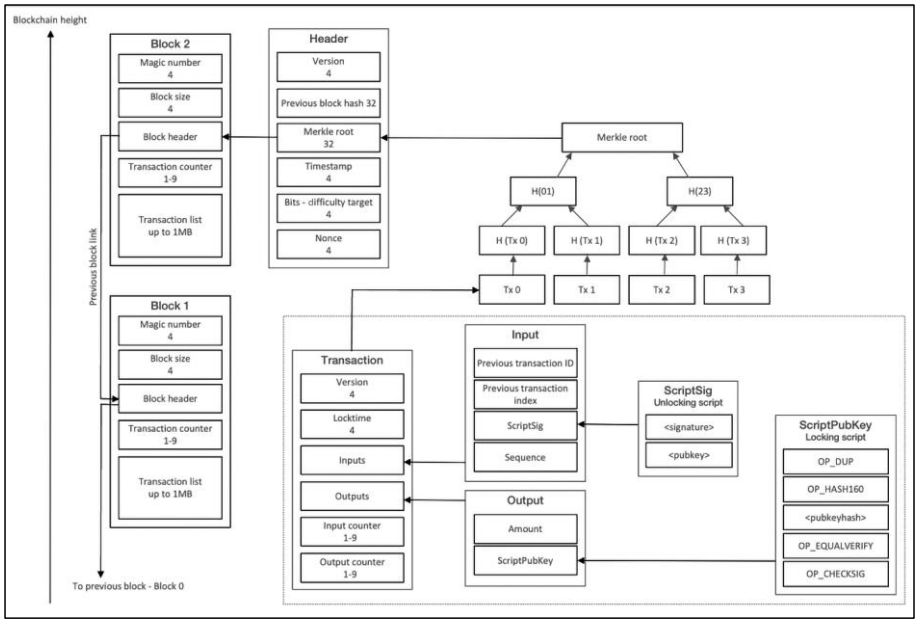

R1
R2
R3
R4
Mesh topology
Leader
Star topology

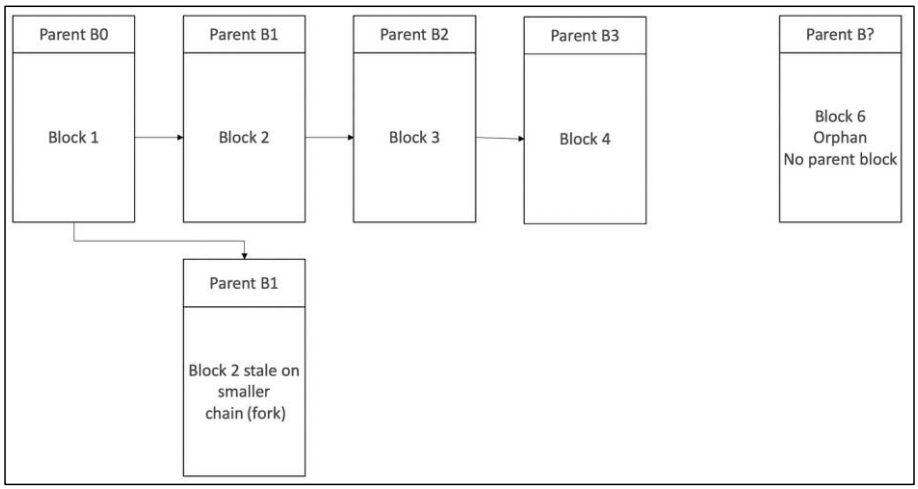

PREPARE
PRE-COMMIT
COMMIT
DECIDE
R1 (leader) selects prepareQC with highest view
NEW-VIEW
PREPARE
PREPARE VOTES
PRE-COMMIT
PRE-COMMIT VOTES
COMMIT
COMMIT VOTES
DECIDE
R1 (leader) broadcasts Msg(prepare, curProposal , high QC) R2,R3,R4 send voteMsg(prepare, m.node , ⊥) to R1
R1 (leader) broadcast Msg(pre-commit, ⊥, prepareQC) R2,R3,R4 (replicas) send voteMsg(pre-commit, m.justify .node,⊥) to R1
R1(leader) broadcast Msg(commit, ⊥, pre-commit QC) R2,R3,R4 (replicas) send voteMsg(commit, m.justify .node , ⊥) to R1
R1 (leader) broadcast Msg(decide, ⊥, commit QC) R2,R3,R4 (replicas) execute & respond to clients
When a new leader has collected new-view messages from (n - f) nodes, the protocol for new leader starts and the leader broadcasts a prepare message.
Upon the leader receiving (n - f) prepare votes, it creates a quorum certificate (QC) called 'prepare quorum certificate'
Upon the leader receiving (n - f) pre-commit votes, it creates a pre-commit quorum certificate and broadcast it to other nodes as the commit message
Upon the leader receiving (n - f) commit votes, it creates a commit quorum certificate. This commit quorum certificate is broadcast to other nodes in the decide message

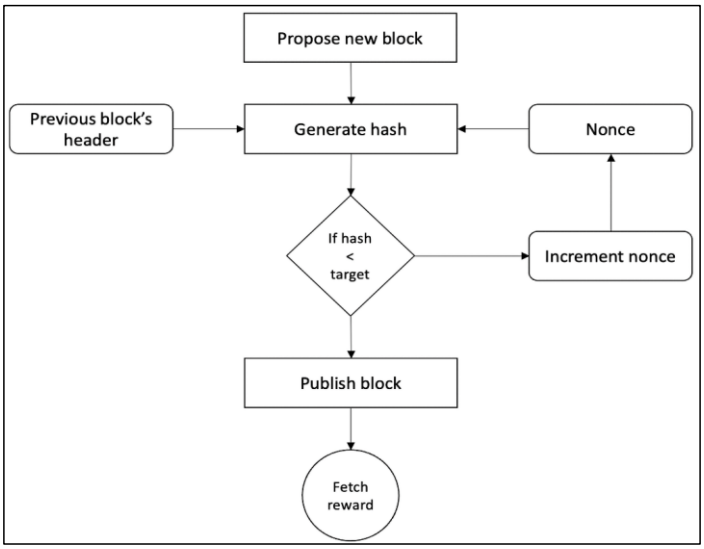
Chapter 6: Bitcoin Architecture

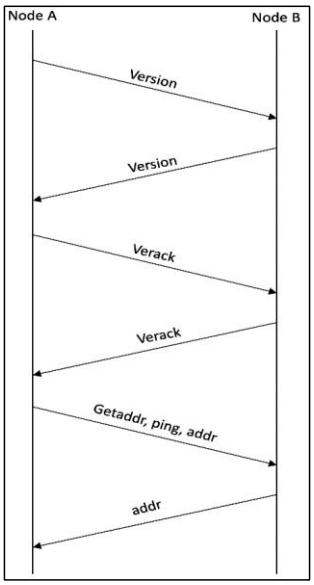


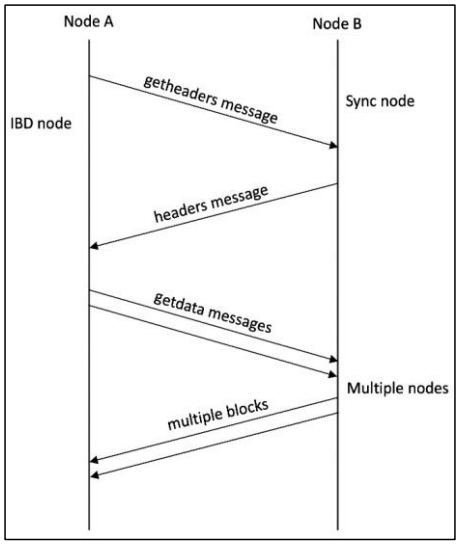

ScriptSig
ScriptPubKey
Script
<sig> <pubKey>
OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG
Stack
<PubKey>
<sig>
<PubKey>
<PubKey>
<sig>
<PubKeyHash>
<PubKey>
<sig>
<PubKeyHash>
<PubKeyHash>
<PubKey>
<sig>
<PubKey>
<sig>
True/False
Execution
OP_DUP
OP_HASH160
<pubKeyHash>
OP_EQUALVERIFY
OP_CHECKSIG
OP_HASH160
<pubKeyHash>
OP_EQUALVERIFY
OP_CHECKSIG
<pubKeyHash>
OP_EQUALVERIFY
OP_CHECKSIG
OP_EQUALVERIFY
OP_CHECKSIG
OP_CHECKSIG

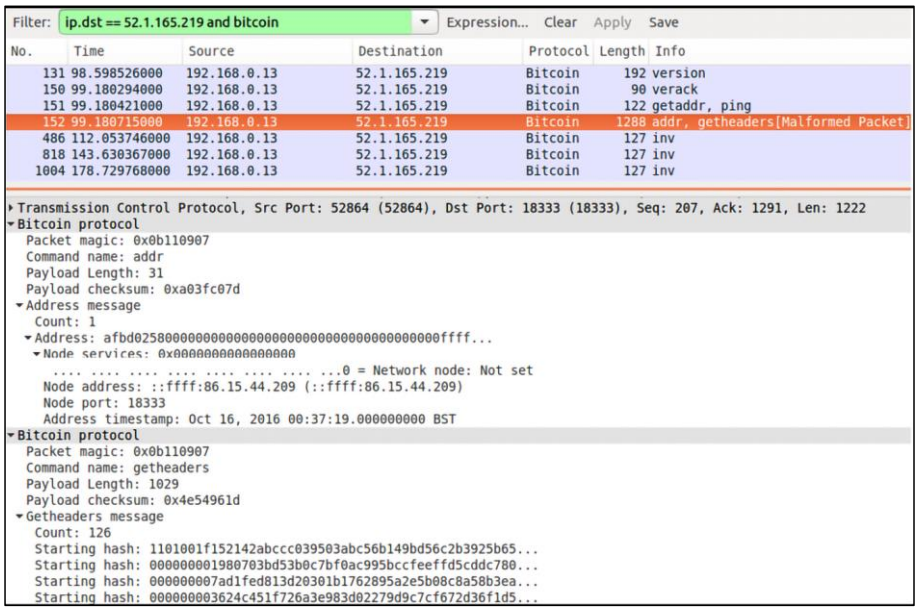

Blockchain height
Block 2
Magic number 4
Block size 4
Block header
Transaction counter 1-9
Transaction list up to 1MB
Header
Version 4
Previous block hash 32
Merkle root 32
Timestamp 4
Bits - difficulty target 4
Nonce 4
Merkle root
H(01)
H(23)
H (Tx 0)
H (Tx 1)
H (Tx 2)
H (Tx 3)
Tx 0
Tx 1
Tx 2
Tx 3
Previous block link
Block 1
Magic number 4
Block size 4
Block header
Transaction counter 1-9
Transaction list up to 1MB
To previous block - Block 0
Transaction
Version 4
Locktime 4
Inputs
Outputs
Input counter 1-9
Output counter 1-9
Input
Previous transaction ID
Previous transaction index
ScriptSig
Sequence
Output
Amount
ScriptPubKey
ScriptSig
Unlocking script
<signature>
<pubkey>
ScriptPubKey
Locking script
OP_DUP
OP_HASH160
<pubkeyhash>
OP_EQUALVERIFY
OP_CHECKSIG


Parent B0
Block 1
Parent B1
Block 2
Parent B2
Block 3
Parent B3
Block 4
Parent B?
Block 6
Orphan
No parent block
Parent B1
Block 2 stale on
smaller
chain (fork)


Propose new block
Previous block's
header
Generate hash
Nonce
If hash
<
target
Increment nonce
Publish block
Fetch
reward


Node A
Node B
Version
Version
Verack
Verack
Getaddr, ping, addr
addr

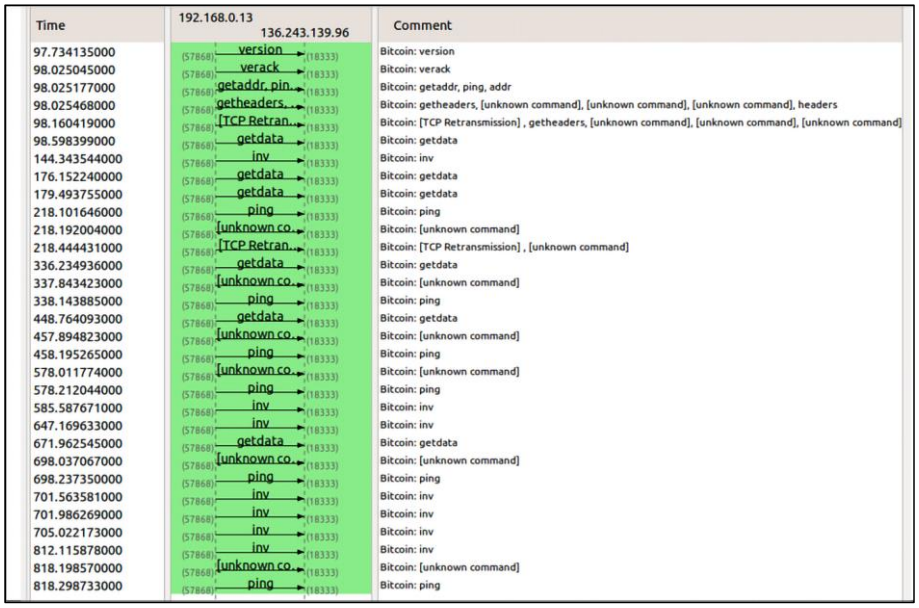

Node A
Node B
getheaders message
Sync node
IBD node
headers message
getdata messages
Multiple nodes
multiple blocks



Filter: ip.dst == 52.1.165.219 and bitcoin — Expression... Clear Apply Save

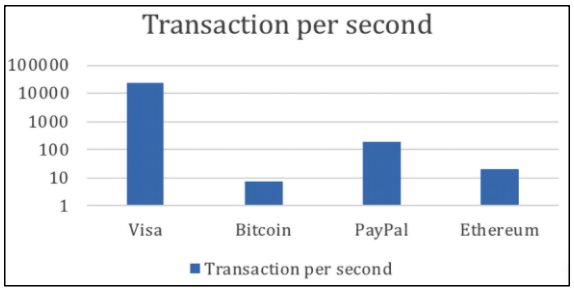
No.	Time	Source	Destination	Protocol	Length	Info
131	98.598526000	192.168.0.13	52.1.165.219	Bitcoin	192	version
150	99.180294000	192.168.0.13	52.1.165.219	Bitcoin	90	verack
151	99.180421000	192.168.0.13	52.1.165.219	Bitcoin	122	getaddr, ping
152	99.180715000	192.168.0.13	52.1.165.219	Bitcoin	1288	addr, getheaders[Malformed Packet]
486	112.053746000	192.168.0.13	52.1.165.219	Bitcoin	127	inv
818	143.630367000	192.168.0.13	52.1.165.219	Bitcoin	127	inv
1004	178.729768000	192.168.0.13	52.1.165.219	Bitcoin	127	inv

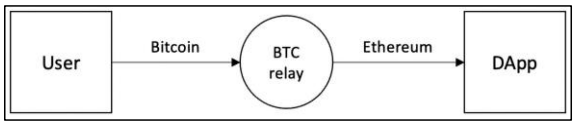
```
▸ Transmission Control Protocol, Src Port: 52864 (52864), Dst Port: 18333 (18333), Seq: 207, Ack: 1291, Len: 1222
▾ Bitcoin protocol
    Packet magic: 0x0b110907
    Command name: addr
    Payload Length: 31
    Payload checksum: 0xa03fc07d
  ▾ Address message
      Count: 1
    ▾ Address: afbd025800000000000000000000000000000000000000000000ffff...
      ▾ Node services: 0x0000000000000000
          .... .... .... .... .... .... .... .... ...0 = Network node: Not set
        Node address: ::ffff:86.15.44.209 (::ffff:86.15.44.209)
        Node port: 18333
        Address timestamp: Oct 16, 2016 00:37:19.000000000 BST
▾ Bitcoin protocol
    Packet magic: 0x0b110907
    Command name: getheaders
    Payload Length: 1029
    Payload checksum: 0x4e54961d
  ▾ Getheaders message
      Count: 126
      Starting hash: 1101001f152142abccc039503abc56b149bd56c2b3925b65...
      Starting hash: 000000001980703bd53b0c7bf0ac995bccfeeffd5cddc780...
      Starting hash: 000000007ad1fed813d20301b1762895a2e5b08c8a58b3ea...
      Starting hash: 000000003624c451f726a3e983d02279d9c7cf672d36f1d5...
```



Time	192.168.0.13 (57868) → 136.243.139.96 (18333)	Comment
97.734135000	version	Bitcoin: version
98.025045000	verack	Bitcoin: verack
98.025177000	getaddr, pin…	Bitcoin: getaddr, ping, addr
98.025468000	getheaders, …	Bitcoin: getheaders, [unknown command], [unknown command], [unknown command], headers
98.160419000	[TCP Retran…	Bitcoin: [TCP Retransmission] , getheaders, [unknown command], [unknown command], [unknown command]
98.598399000	getdata	Bitcoin: getdata
144.343544000	inv	Bitcoin: inv
176.152240000	getdata	Bitcoin: getdata
179.493755000	getdata	Bitcoin: getdata
218.101646000	ping	Bitcoin: ping
218.192004000	[unknown co…	Bitcoin: [unknown command]
218.444431000	[TCP Retran…	Bitcoin: [TCP Retransmission] , [unknown command]
336.234936000	getdata	Bitcoin: getdata
337.843423000	[unknown co…	Bitcoin: [unknown command]
338.143885000	ping	Bitcoin: ping
448.764093000	getdata	Bitcoin: getdata
457.894823000	[unknown co…	Bitcoin: [unknown command]
458.195265000	ping	Bitcoin: ping
578.011774000	[unknown co…	Bitcoin: [unknown command]
578.212044000	ping	Bitcoin: ping
585.587671000	inv	Bitcoin: inv
647.169633000	inv	Bitcoin: inv
671.962545000	getdata	Bitcoin: getdata
698.037067000	[unknown co…	Bitcoin: [unknown command]
698.237350000	ping	Bitcoin: ping
701.563581000	inv	Bitcoin: inv
701.986269000	inv	Bitcoin: inv
705.022173000	inv	Bitcoin: inv
812.115878000	inv	Bitcoin: inv
818.198570000	[unknown co…	Bitcoin: [unknown command]
818.298733000	ping	Bitcoin: ping

Chapter 7: Bitcoin in Practice

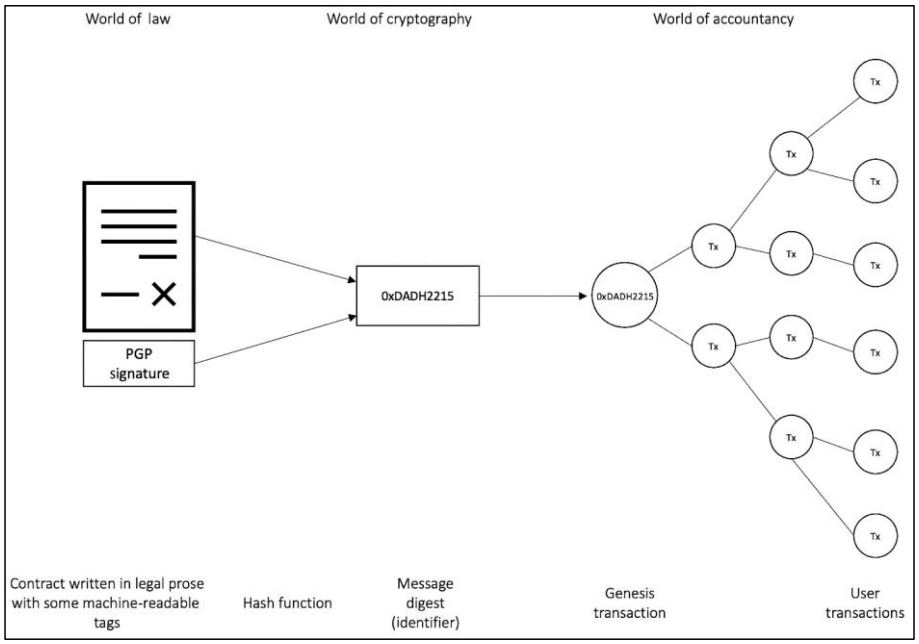


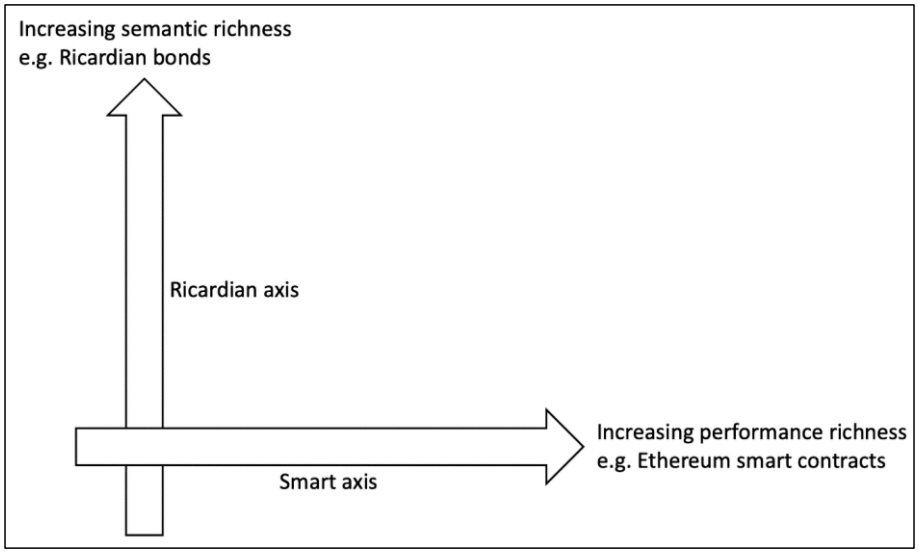


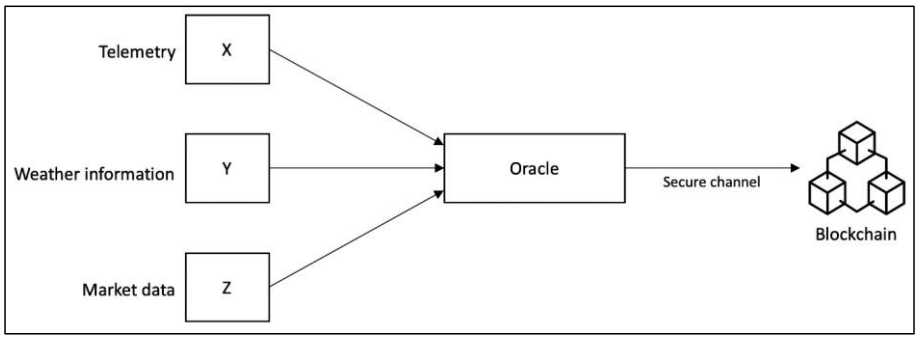

```
{
  "amount": 0.00000000,
  "fee": -0.00003320,
  "confirmations": 7,
  "blockhash": "7c50e79b54dcda17e32cc7b7b53fc095584befe4e952422bdf096de3b93fe539",
  "blockindex": 1,
  "blocktime": 1577228072,
  "txid": "a83ff460a32f29387d531f19e7092a5dcf6ce52d20931227447c0b9b7a5f2980",
  "walletconflicts": [
  ],
  "time": 1577227733,
  "timereceived": 1577227733,
  "bip125-replaceable": "no",
  "details": [
    {
      "address": "2NC31WFFRwRkwd3S4TpyjN5GGDY7E63GSVd",
      "category": "send",
      "amount": -20.00000000,
      "label": "",
      "vout": 0,
      "fee": -0.00003320,
      "abandoned": false
    },
    {
      "address": "2NC31WFFRwRkwd3S4TpyjN5GGDY7E63GSVd",
      "category": "receive",
      "amount": 20.00000000,
      "label": "",
      "vout": 0
    }
  ],
  "hex": "020000000001018712030 19a3456fc50b2044e79a4f078bc0ceb278734d44faf82ce4f24
35770000000017a914ce1afa4c71513d952194695adedfad119faf8f87870851d0b20000000017a914
c730f514cc654f222cecce09faf6a8e87d07c9a44a0220273478b3f452bec625d3d87002cb00831476
6ef37cf310609bce20e575c8000000"
}
```

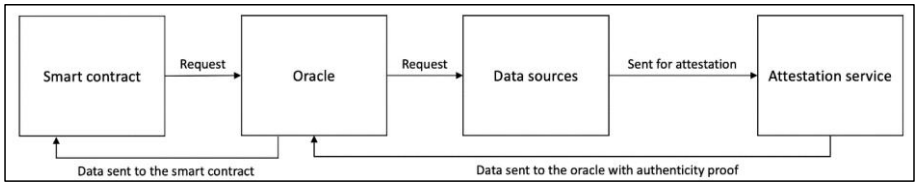

Commented [KS1]: Left it here, apply borders

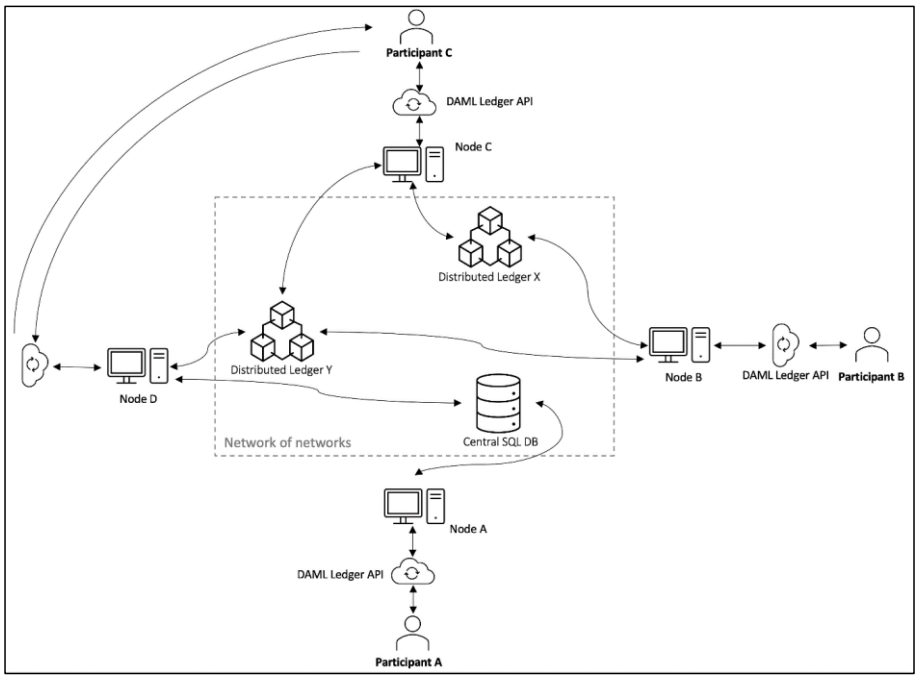
Chapter 8: Smart Contracts



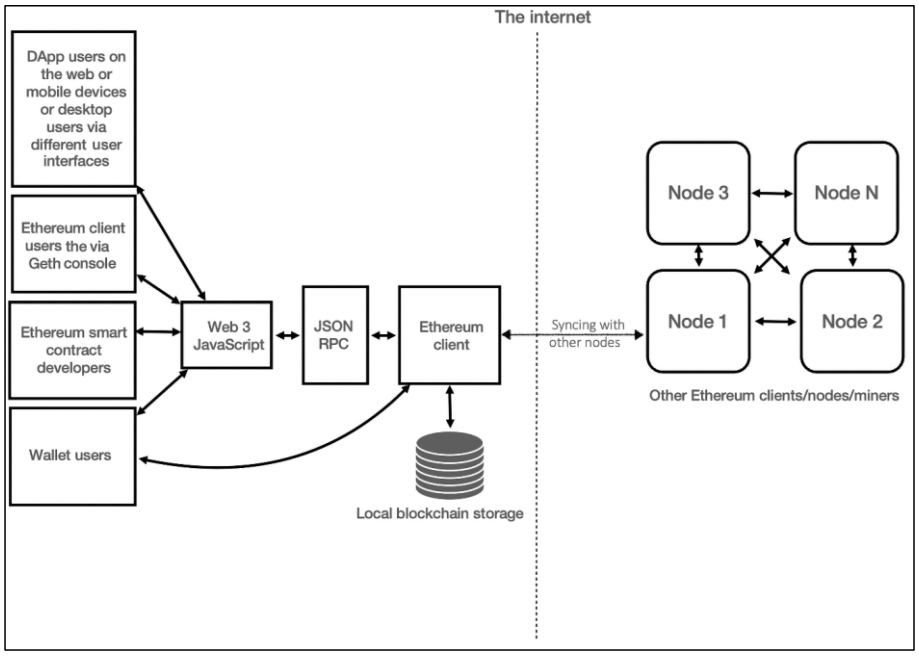


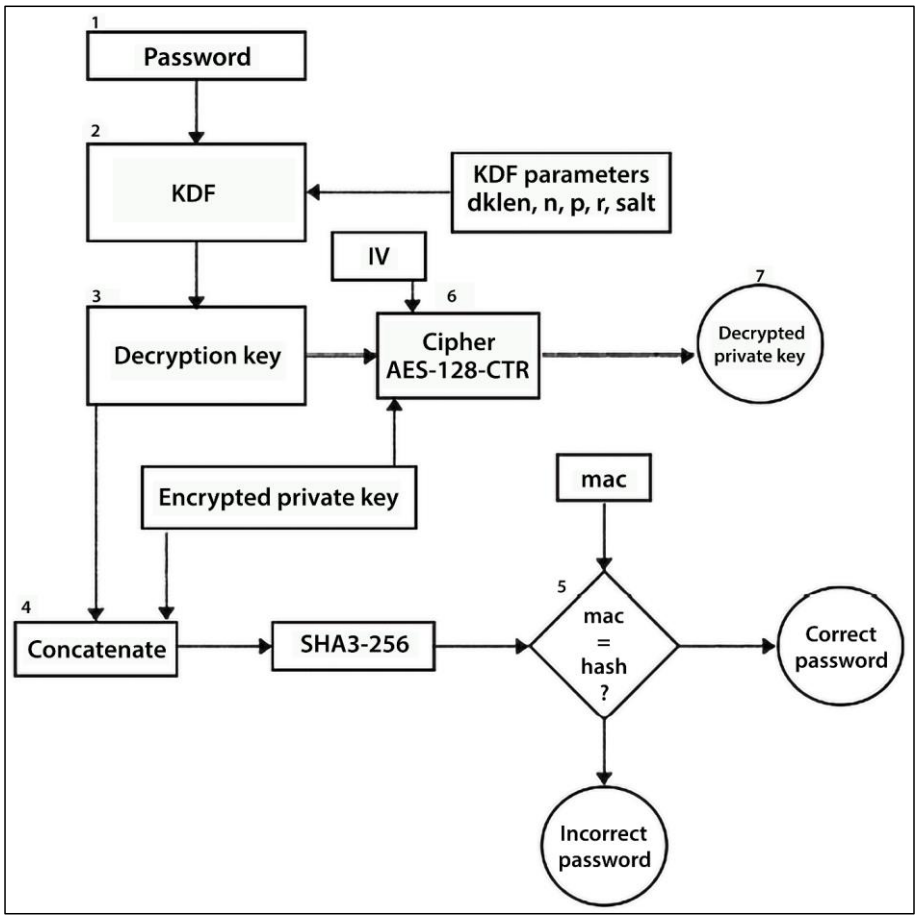

Telemetry
X
Weather information
Y
Market data
Z
Oracle
Secure channel
Blockchain

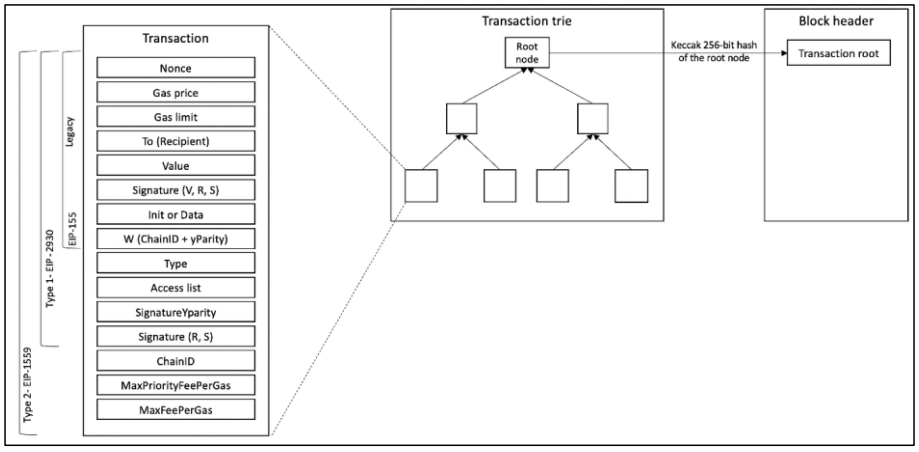

Smart contract
Request
Oracle
Request
Data sources
Sent for attestation
Attestation service
Data sent to the smart contract
Data sent to the oracle with authenticity proof

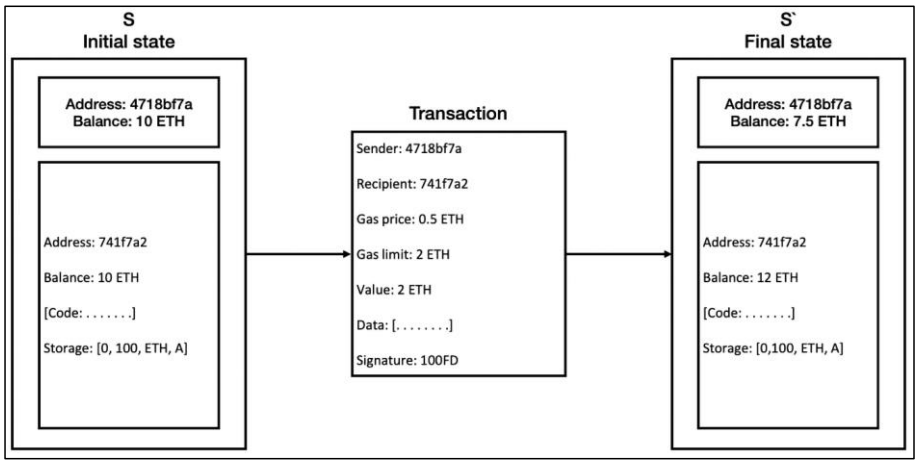

Participant C
DAML Ledger API
Node C
Distributed Ledger X
Distributed Ledger Y
Node D
Node B
DAML Ledger API
Participant B
Network of networks
Central SQL DB
Node A
DAML Ledger API
Participant A

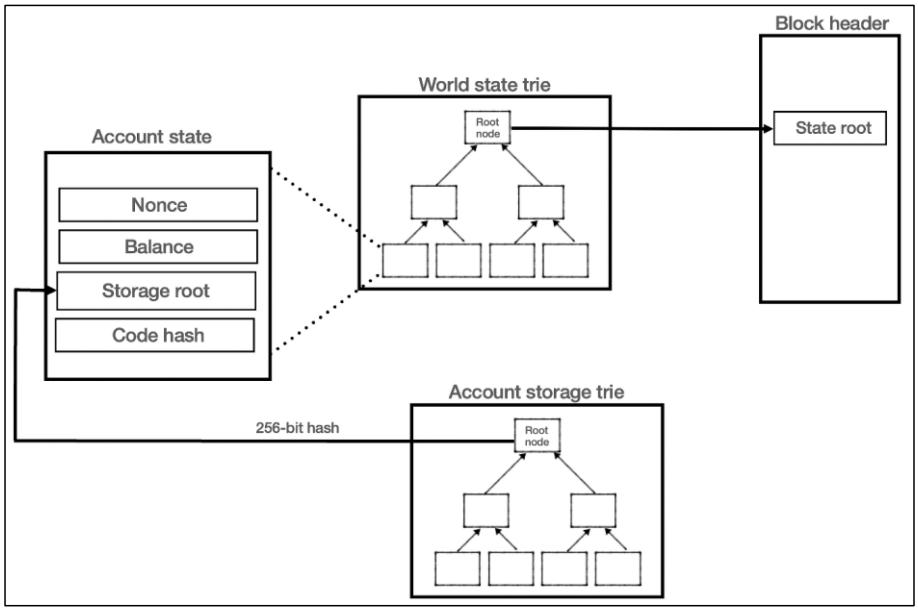
Chapter 9: Ethereum Architecture

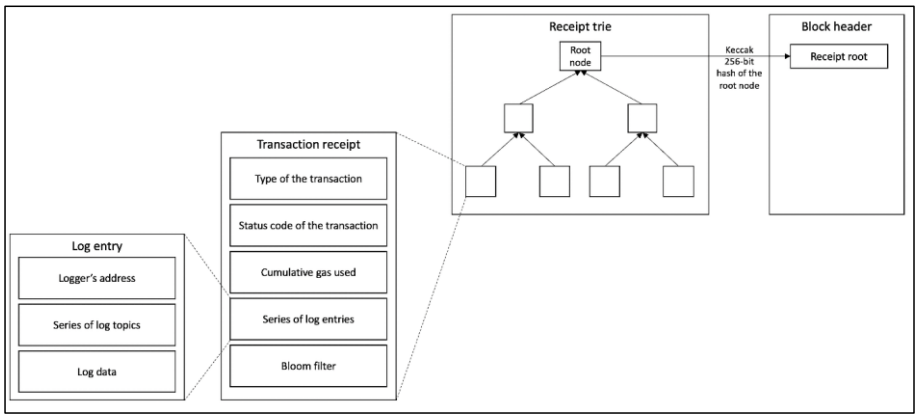


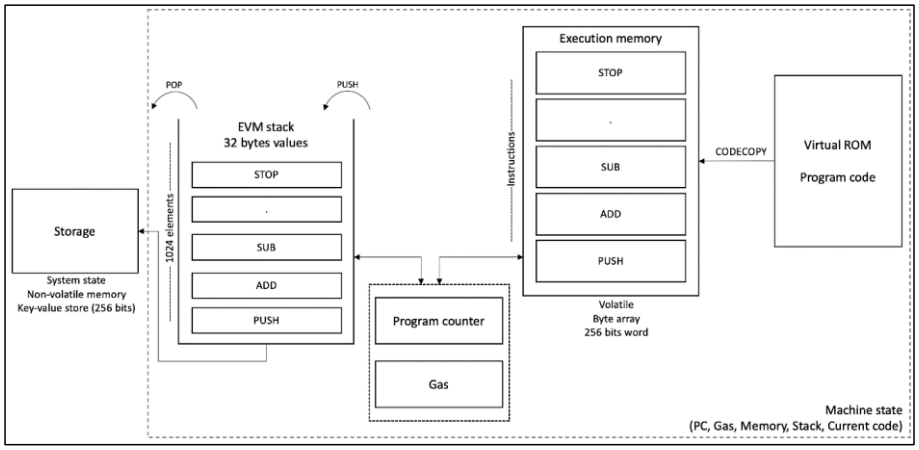

1
Password
2
KDF
KDF parameters
dklen, n, p, r, salt
IV
3
Decryption key
6
Cipher
AES-128-CTR
7
Decrypted
private key
Encrypted private key
mac
4
Concatenate
SHA3-256
5
mac
=
hash
?
Correct
password
Incorrect
password

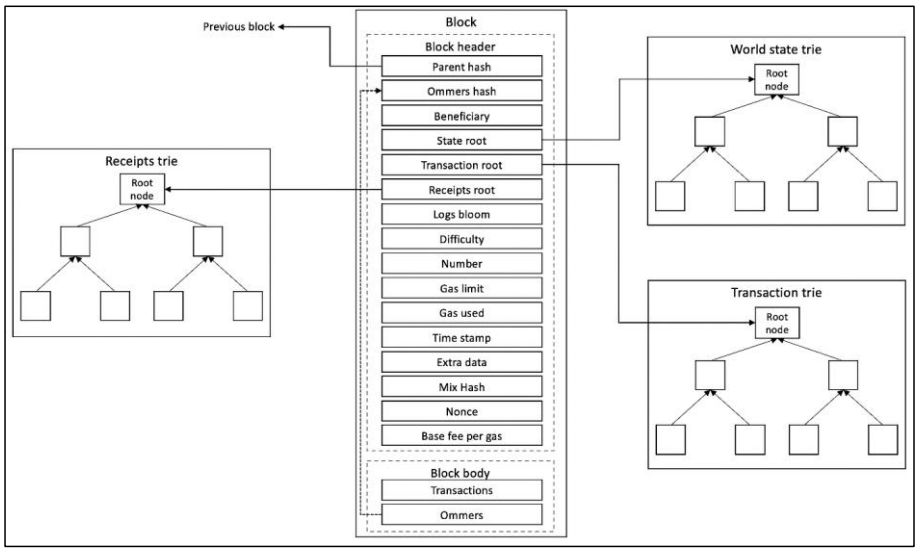

Transaction
Nonce
Gas price
Gas limit
To (Recipient)
Value
Signature (V, R, S)
Init or Data
W (ChainID + yParity)
Type
Access list
SignatureYparity
Signature (R, S)
ChainID
MaxPriorityFeePerGas
MaxFeePerGas
Legacy
EIP-155
Type 1 - EIP-2930
Type 2 - EIP-1559
Transaction trie
Root
node
Keccak 256-bit hash
of the root node
Block header
Transaction root

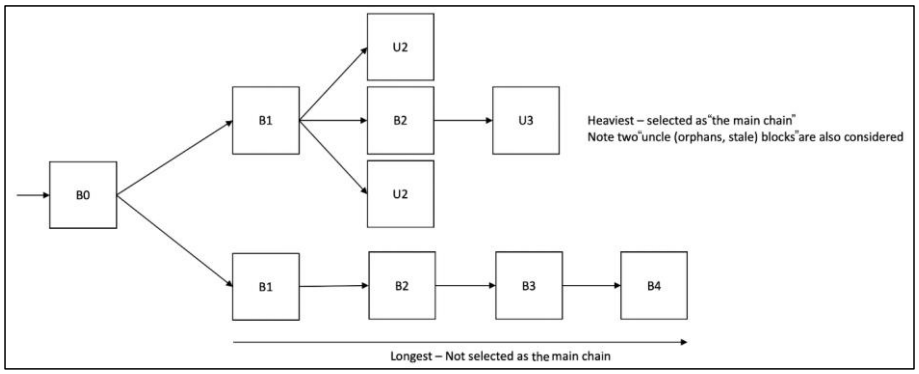

S
Initial state
Address: 4718bf7a
Balance: 10 ETH
Address: 741f7a2
Balance: 10 ETH
[Code: . . . . . .]
Storage: [0, 100, ETH, A]
Transaction
Sender: 4718bf7a
Recipient: 741f7a2
Gas price: 0.5 ETH
Gas limit: 2 ETH
Value: 2 ETH
Data: [. . . . . .]
Signature: 100FD
S`
Final state
Address: 4718bf7a
Balance: 7.5 ETH
Address: 741f7a2
Balance: 12 ETH
[Code: . . . . . .]
Storage: [0,100, ETH, A]

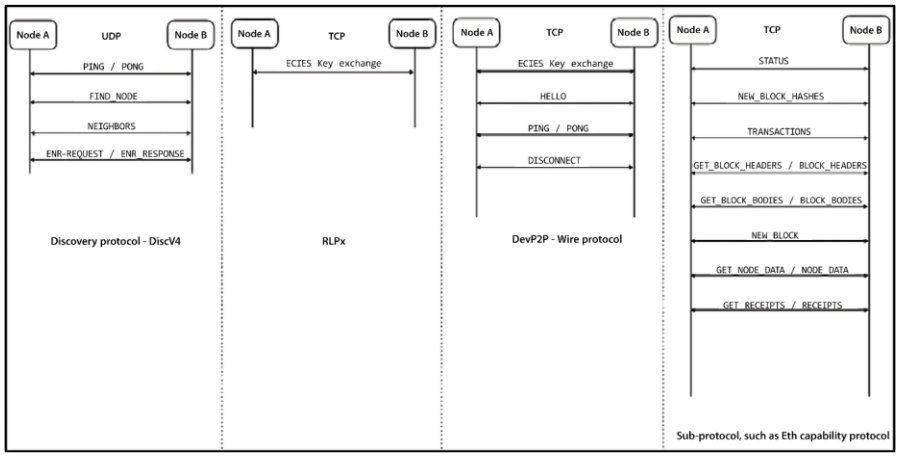

Block header
World state trie
Root node
State root
Account state
Nonce
Balance
Storage root
Code hash
Account storage trie
256-bit hash
Root node

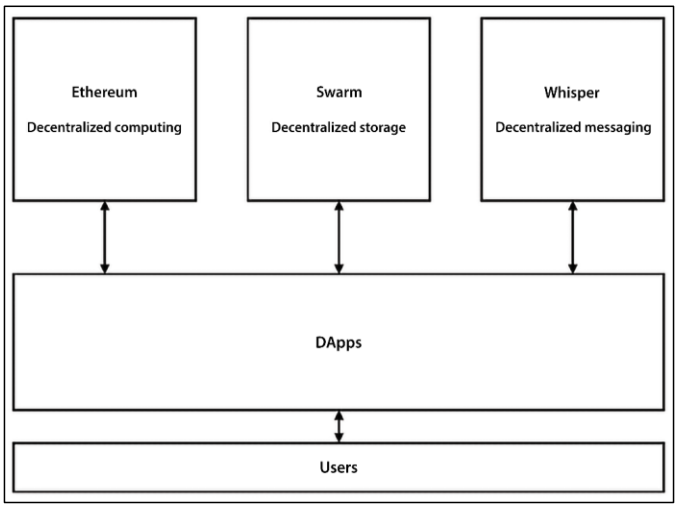

Receipt trie
Root node
Keccak 256-bit hash of the root node
Block header
Receipt root
Transaction receipt
Type of the transaction
Status code of the transaction
Cumulative gas used
Series of log entries
Bloom filter
Log entry
Logger's address
Series of log topics
Log data


Execution memory
STOP
.
SUB
ADD
PUSH
Instructions
CODECOPY
Virtual ROM
Program code
POP
PUSH
EVM stack
32 bytes values
STOP
.
SUB
ADD
PUSH
1024 elements
Storage
System state
Non-volatile memory
Key-value store (256 bits)
Program counter
Gas
Volatile
Byte array
256 bits word
Machine state
(PC, Gas, Memory, Stack, Current code)

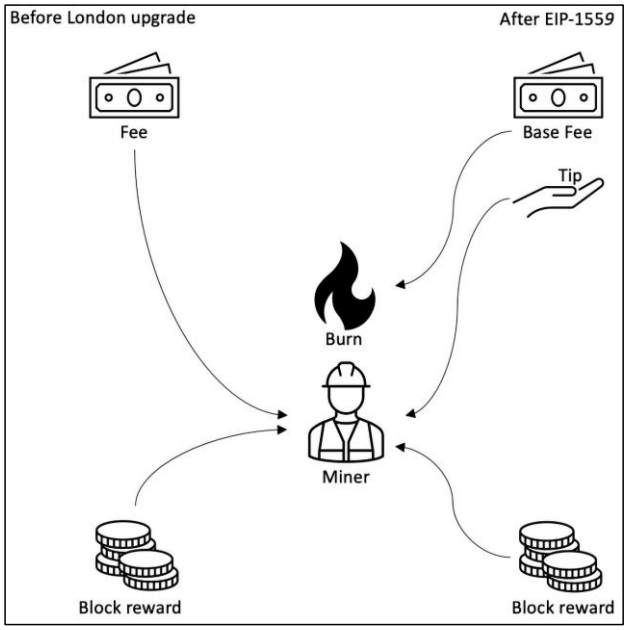

Previous block
Block
Block header
Parent hash
Ommers hash
Beneficiary
State root
Transaction root
Receipts root
Logs bloom
Difficulty
Number
Gas limit
Gas used
Time stamp
Extra data
Mix Hash
Nonce
Base fee per gas
Block body
Transactions
Ommers
World state trie
Root node
Transaction trie
Root node
Receipts trie
Root node

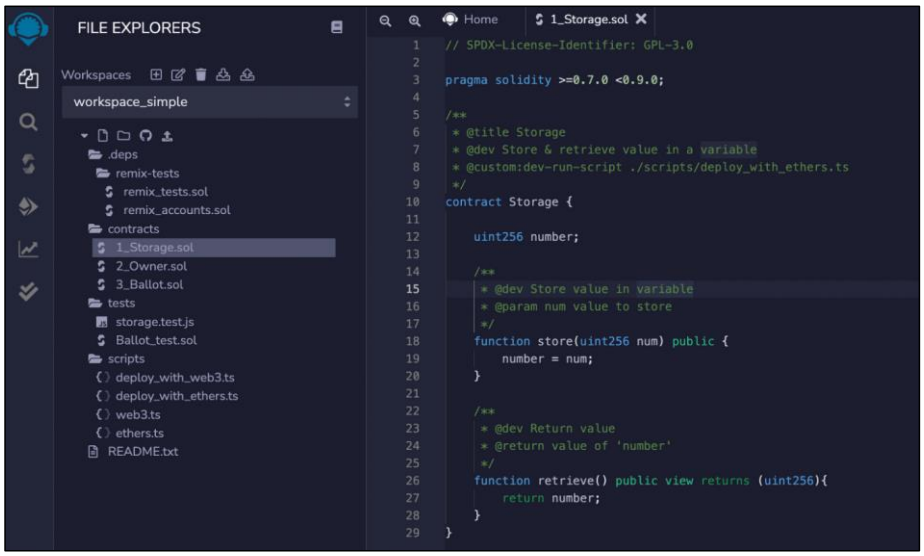

B0
B1
U2
B2
U3
U2
B1
B2
B3
B4
Heaviest – selected as "the main chain"
Note two "uncle (orphans, stale) blocks" are also considered
Longest – Not selected as the main chain

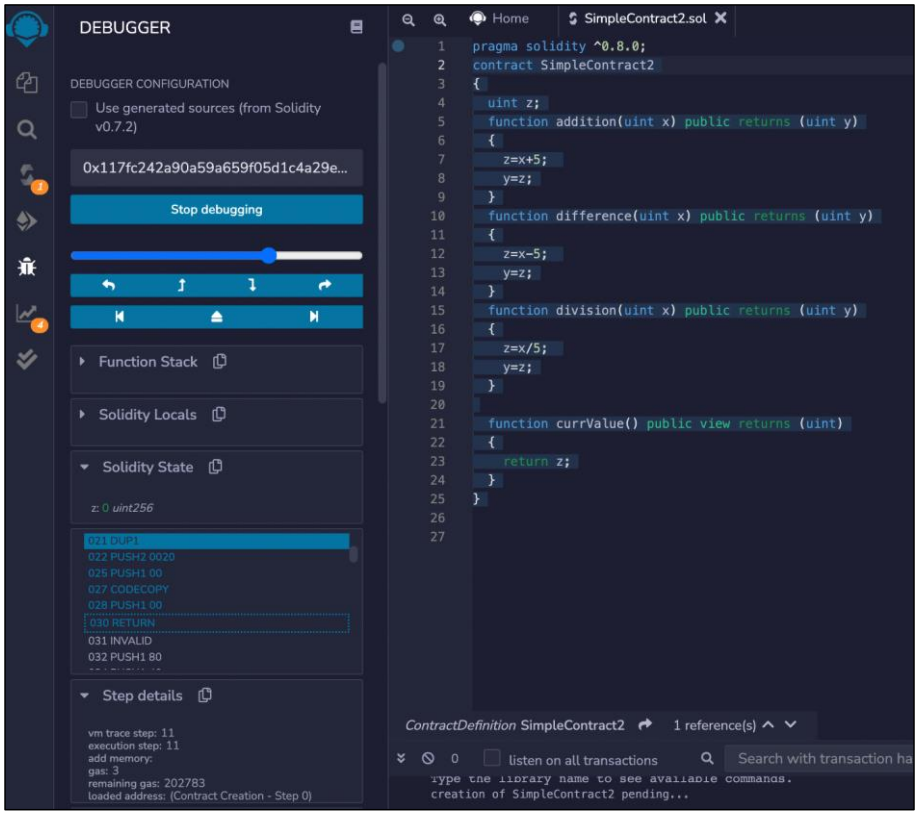

Node A
UDP
Node B
PING / PONG
FIND_NODE
NEIGHBORS
ENR-REQUEST / ENR_RESPONSE
Discovery protocol - DiscV4
Node A
TCP
Node B
ECIES Key exchange
RLPx
Node A
TCP
Node B
ECIES Key exchange
HELLO
PING / PONG
DISCONNECT
DevP2P - Wire protocol
Node A
TCP
Node B
STATUS
NEW_BLOCK_HASHES
TRANSACTIONS
GET_BLOCK_HEADERS / BLOCK_HEADERS
GET_BLOCK_BODIES / BLOCK_BODIES
NEW_BLOCK
GET_NODE_DATA / NODE_DATA
GET_RECEIPTS / RECEIPTS
Sub-protocol, such as Eth capability protocol

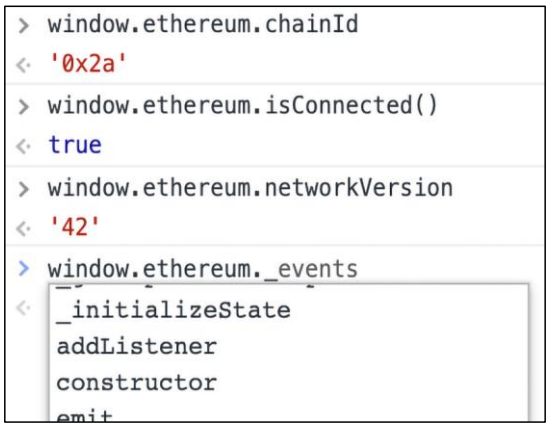

Ethereum
Decentralized computing
Swarm
Decentralized storage
Whisper
Decentralized messaging
DApps
Users

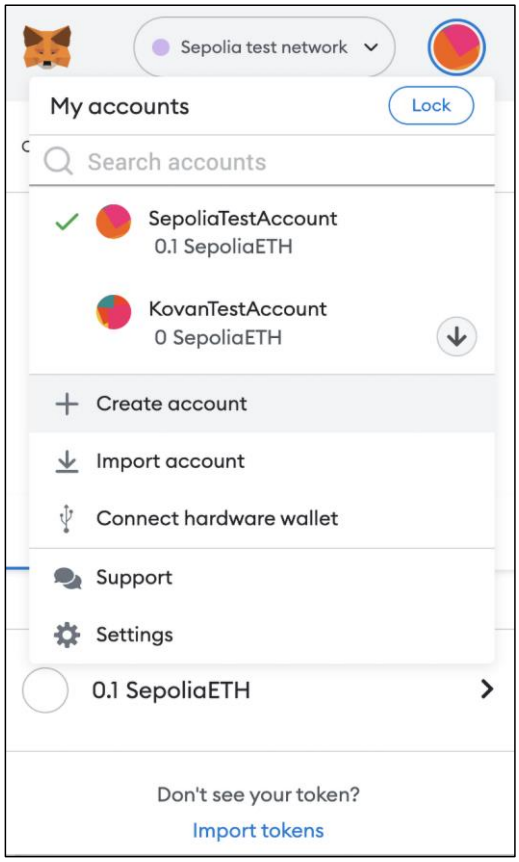
Chapter 10: Ethereum in Practice

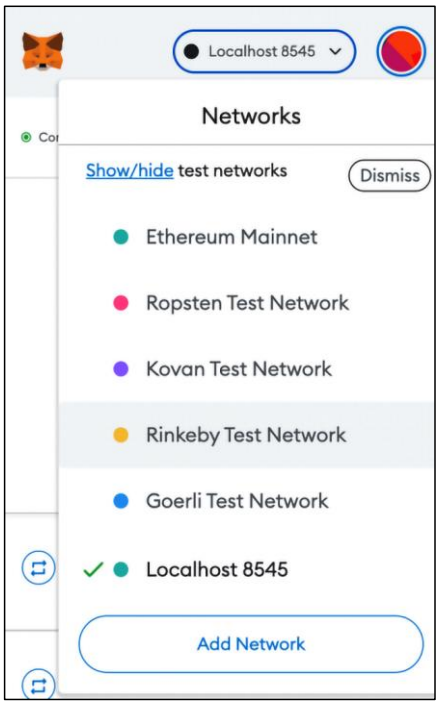


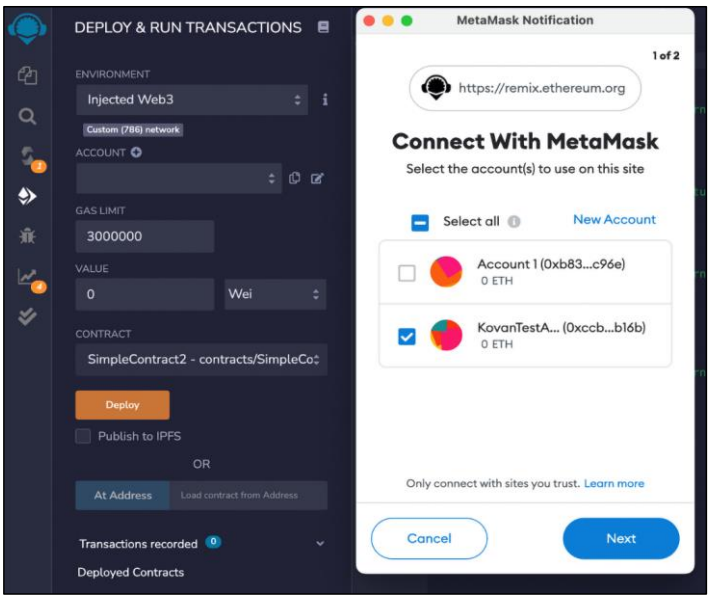

FILE EXPLORERS
Workspaces
workspace_simple
.deps
remix-tests
remix_tests.sol
remix_accounts.sol
contracts
1_Storage.sol
2_Owner.sol
3_Ballot.sol
tests
storage.test.js
Ballot_test.sol
scripts
deploy_with_web3.ts
deploy_with_ethers.ts
web3.ts
ethers.ts
README.txt
Home
1_Storage.sol
// SPDX-License-Identifier: GPL-3.0
pragma solidity >=0.7.0 <0.9.0;
/**
* @title Storage
* @dev Store & retrieve value in a variable
* @custom:dev-run-script ./scripts/deploy_with_ethers.ts
*/
contract Storage {
uint256 number;
/**
* @dev Store value in variable
* @param num value to store
*/
function store(uint256 num) public {
number = num;
}
/**
* @dev Return value
* @return value of 'number'
*/
function retrieve() public view returns (uint256){
return number;
}
}

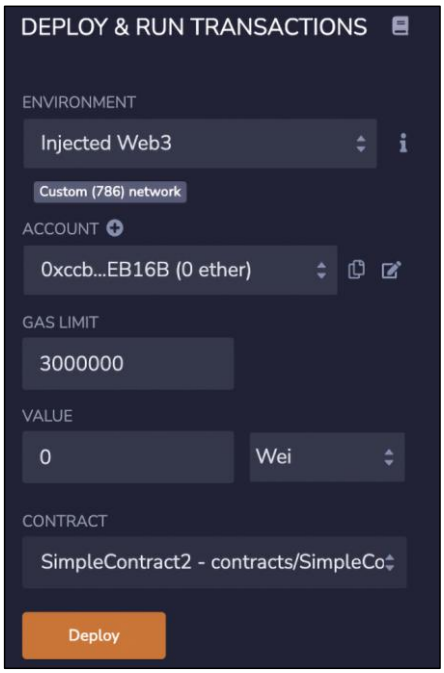

DEBUGGER
DEBUGGER CONFIGURATION
Use generated sources (from Solidity v0.7.2)
0x117fc242a90a59a659f05d1c4a29e...
Stop debugging
Function Stack
Solidity Locals
Solidity State
z: 0 uint256
021 DUP1
022 PUSH2 0020
025 PUSH1 00
027 CODECOPY
028 PUSH1 00
030 RETURN
031 INVALID
032 PUSH1 80
Step details
vm trace step: 11
execution step: 11
add memory:
gas: 3
remaining gas: 202783
loaded address: (Contract Creation - Step 0)
Home
SimpleContract2.sol
pragma solidity ^0.8.0;
contract SimpleContract2
{
uint z;
function addition(uint x) public returns (uint y)
{
z=x+5;
y=z;
}
function difference(uint x) public returns (uint y)
{
z=x-5;
y=z;
}
function division(uint x) public returns (uint y)
{
z=x/5;
y=z;
}
function currValue() public view returns (uint)
{
return z;
}
}
ContractDefinition SimpleContract2
1 reference(s)
listen on all transactions
Search with transaction ha
creation of SimpleContract2 pending...

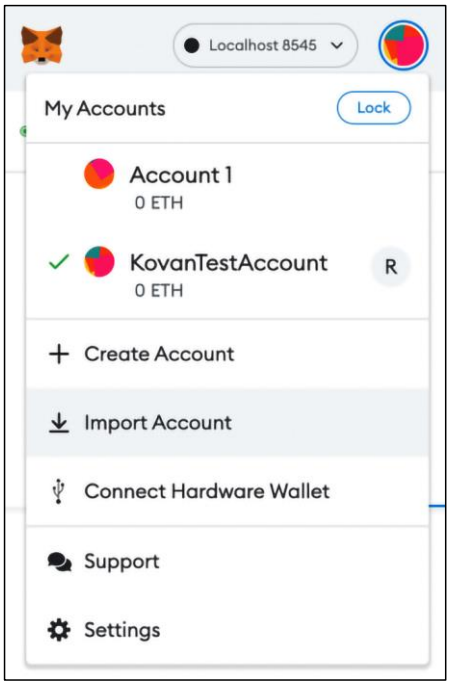

> window.ethereum.chainId
'0x2a'
> window.ethereum.isConnected()
true
> window.ethereum.networkVersion
'42'
> window.ethereum._events
_initializeState
addListener
constructor
emit

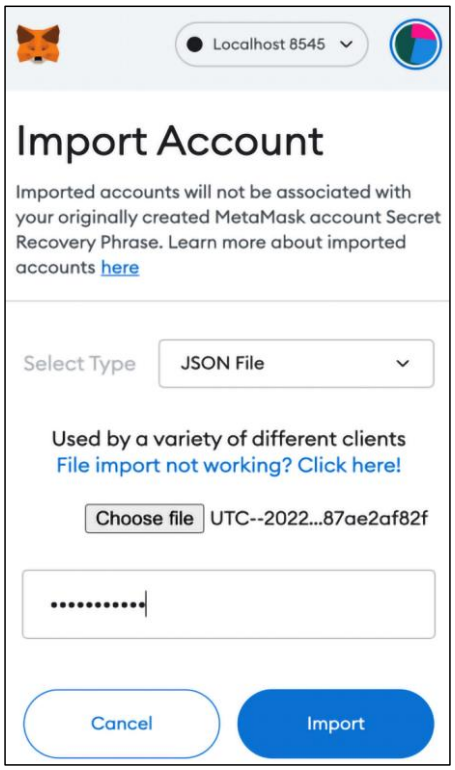

Sepolia test network
My accounts
Lock
Search accounts
SepoliaTestAccount
0.1 SepoliaETH
KovanTestAccount
0 SepoliaETH
Create account
Import account
Connect hardware wallet
Support
Settings
0.1 SepoliaETH
Don't see your token?
Import tokens

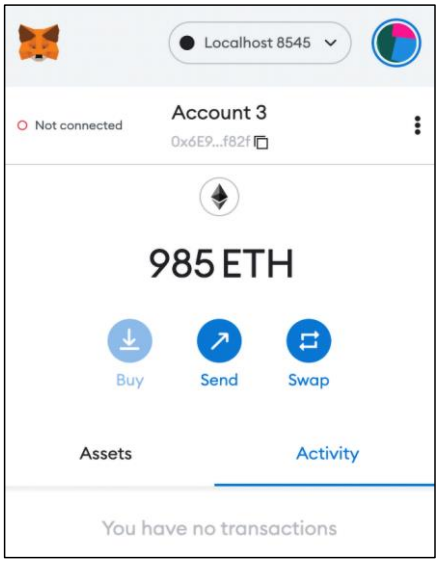

Localhost 8545
Networks
Show/hide test networks
Dismiss
Ethereum Mainnet
Ropsten Test Network
Kovan Test Network
Rinkeby Test Network
Goerli Test Network
Localhost 8545
Add Network

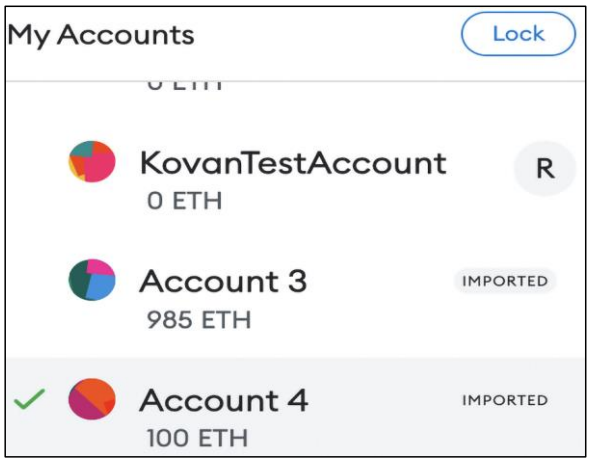

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Custom (786) network
ACCOUNT
GAS LIMIT
3000000
VALUE
0
Wei
CONTRACT
SimpleContract2 - contracts/SimpleCo
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded
Deployed Contracts
MetaMask Notification
1 of 2
https://remix.ethereum.org
Connect With MetaMask
Select the account(s) to use on this site
Select all
New Account
Account 1 (0xb83...c96e)
0 ETH
KovanTestA... (0xccb...b16b)
0 ETH
Only connect with sites you trust. Learn more
Cancel
Next

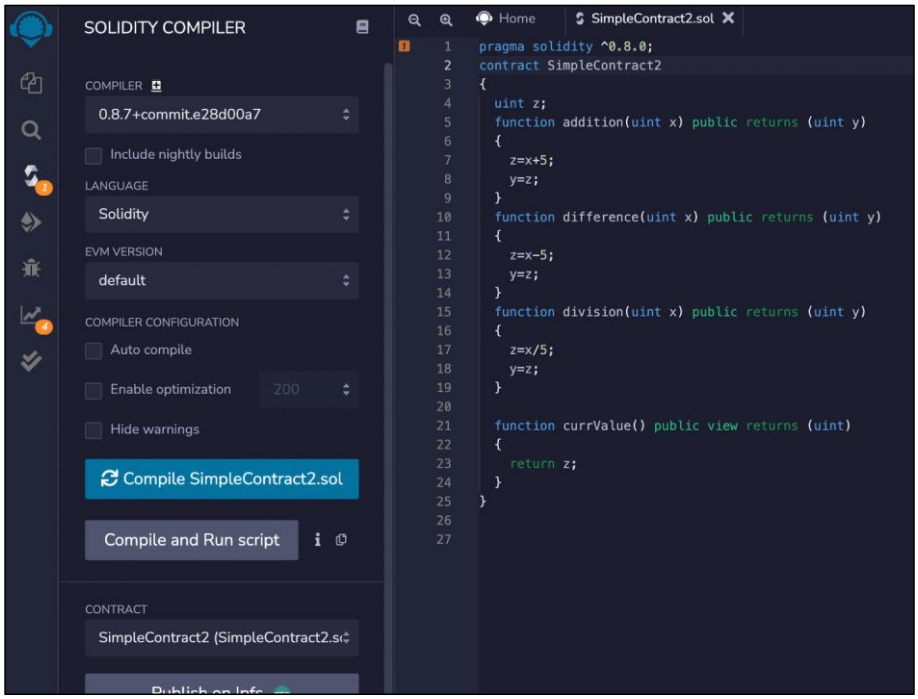

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Custom (786) network
ACCOUNT
0xccb...EB16B (0 ether)
GAS LIMIT
3000000
VALUE
0
Wei
CONTRACT
SimpleContract2 - contracts/SimpleCo
Deploy

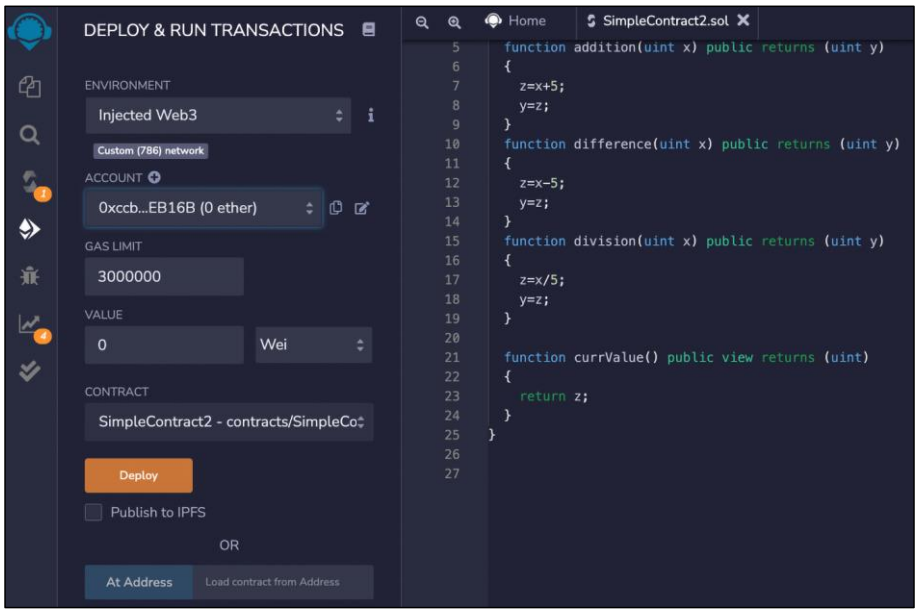

Localhost 8545
My Accounts
Lock
Account 1
0 ETH
KovanTestAccount
0 ETH
R
Create Account
Import Account
Connect Hardware Wallet
Support
Settings

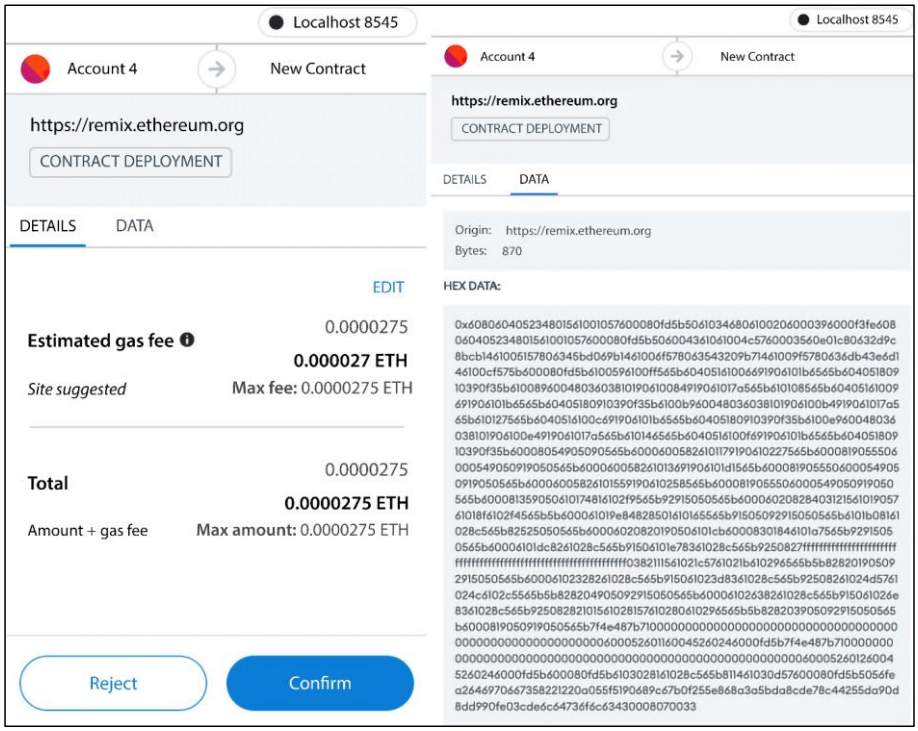

Localhost 8545
Import Account
Imported accounts will not be associated with your originally created MetaMask account Secret Recovery Phrase. Learn more about imported accounts here
Select Type
JSON File
Used by a variety of different clients
File import not working? Click here!
Choose file
UTC--2022...87ae2af82f
Cancel
Import

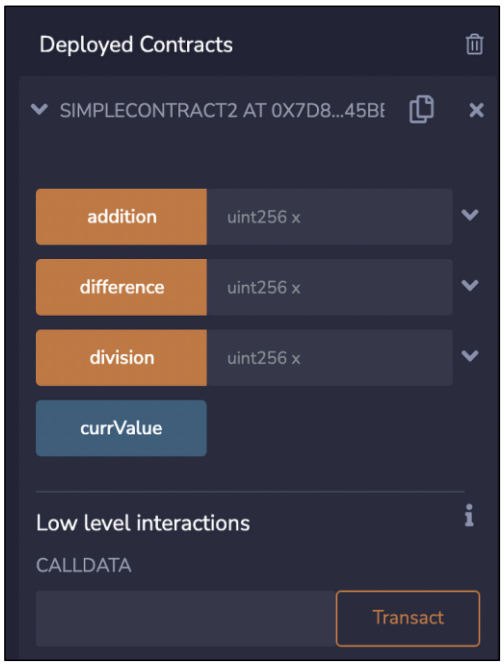

Localhost 8545
Not connected
Account 3
0x6E9...f82f
985 ETH
Buy
Send
Swap
Assets
Activity
You have no transactions

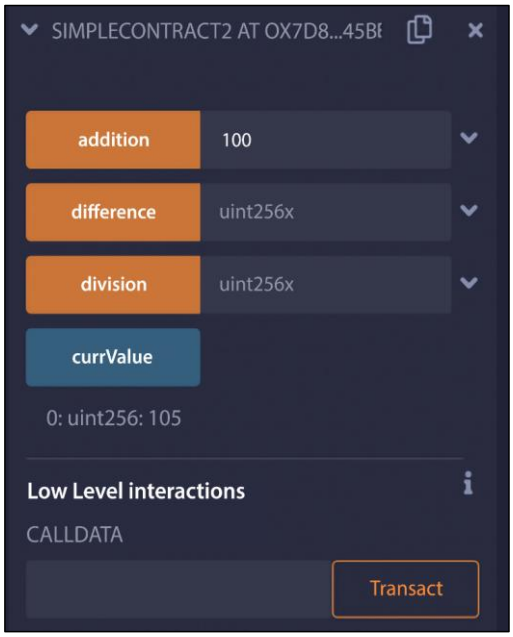

My Accounts
Lock
KovanTestAccount
0 ETH
R
Account 3
985 ETH
IMPORTED
Account 4
100 ETH
IMPORTED

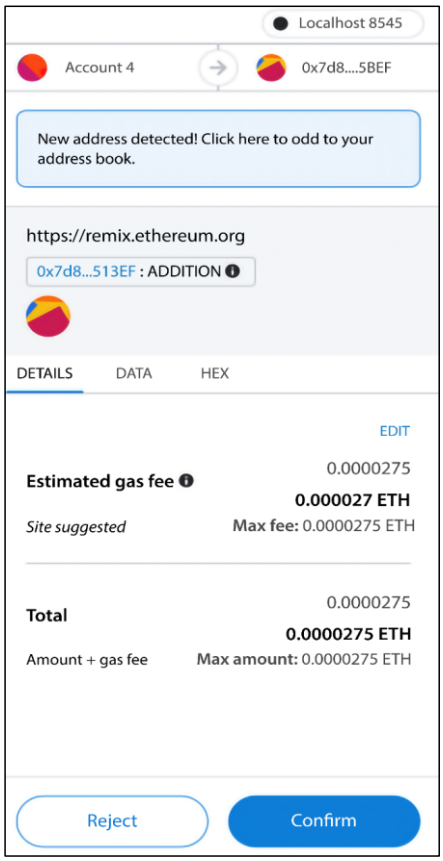

SOLIDITY COMPILER
COMPILER
0.8.7+commit.e28d00a7
Include nightly builds
LANGUAGE
Solidity
EVM VERSION
default
COMPILER CONFIGURATION
Auto compile
Enable optimization
200
Hide warnings
Compile SimpleContract2.sol
Compile and Run script
CONTRACT
SimpleContract2 (SimpleContract2.sc
Home
SimpleContract2.sol
pragma solidity ^0.8.0;
contract SimpleContract2
{
uint z;
function addition(uint x) public returns (uint y)
{
z=x+5;
y=z;
}
function difference(uint x) public returns (uint y)
{
z=x-5;
y=z;
}
function division(uint x) public returns (uint y)
{
z=x/5;
y=z;
}
function currValue() public view returns (uint)
{
return z;
}
}

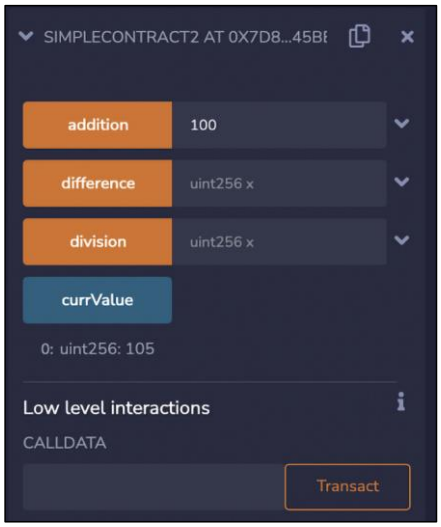

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Injected Web3
Custom (786) network
ACCOUNT
0xccb...EB16B (0 ether)
GAS LIMIT
3000000
VALUE
0
Wei
CONTRACT
SimpleContract2 - contracts/SimpleCo
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Home
SimpleContract2.sol
function addition(uint x) public returns (uint y)
{
z=x+5;
y=z;
}
function difference(uint x) public returns (uint y)
{
z=x-5;
y=z;
}
function division(uint x) public returns (uint y)
{
z=x/5;
y=z;
}
function currValue() public view returns (uint)
{
return z;
}
}

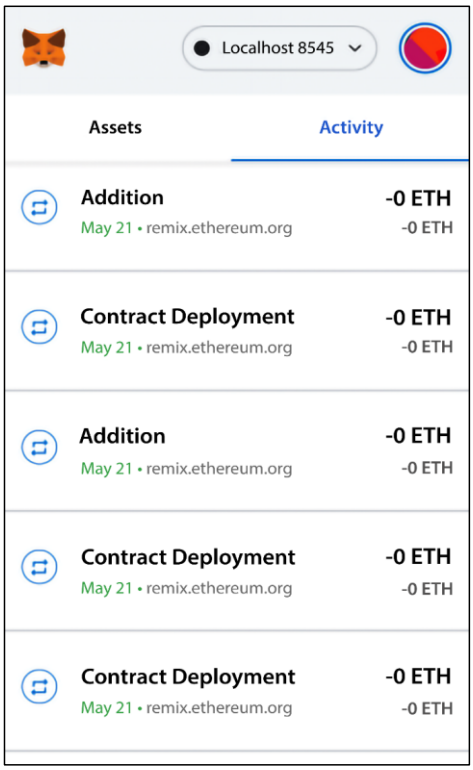

Localhost 8545
Account 4
New Contract
https://remix.ethereum.org
CONTRACT DEPLOYMENT
DETAILS
DATA
EDIT
Estimated gas fee
0.0000275
0.000027 ETH
Site suggested
Max fee: 0.0000275 ETH
Total
0.0000275
0.0000275 ETH
Amount + gas fee
Max amount: 0.0000275 ETH
Reject
Confirm
Origin: https://remix.ethereum.org
Bytes: 870
HEX DATA:

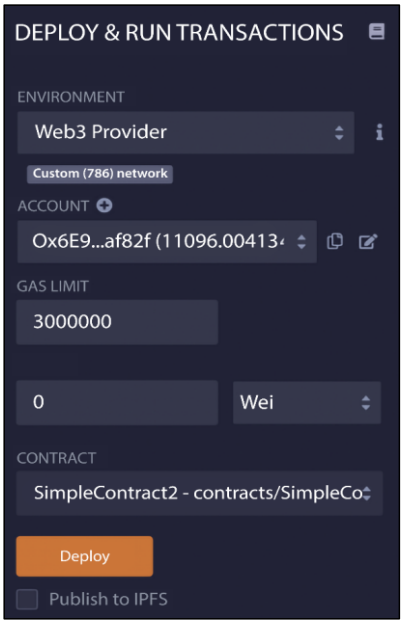

Deployed Contracts
SIMPLECONTRACT2 AT 0X7D8...45BE
addition
uint256 x
difference
uint256 x
division
uint256 x
currValue
Low level interactions
CALLDATA
Transact

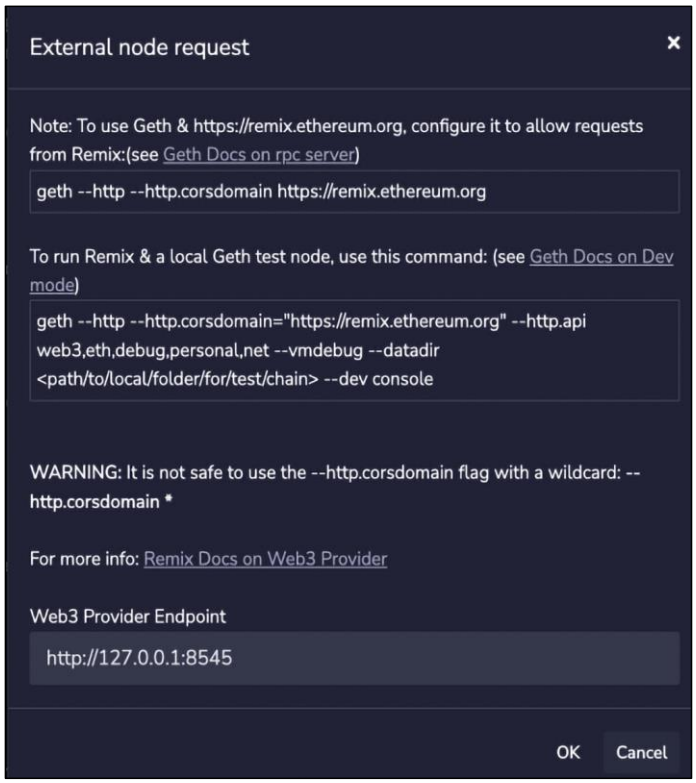

SIMPLECONTRACT2 AT OX7D8...45BE
addition
100
difference
uint256x
division
uint256x
currValue
0: uint256: 105
Low Level interactions
CALLDATA
Transact


Localhost 8545
Account 4
0x7d8....5BEF
New address detected! Click here to odd to your address book.
https://remix.ethereum.org
0x7d8...513EF : ADDITION
DETAILS
DATA
HEX
EDIT
Estimated gas fee
0.0000275
0.000027 ETH
Site suggested
Max fee: 0.0000275 ETH
Total
0.0000275
0.0000275 ETH
Amount + gas fee
Max amount: 0.0000275 ETH
Reject
Confirm

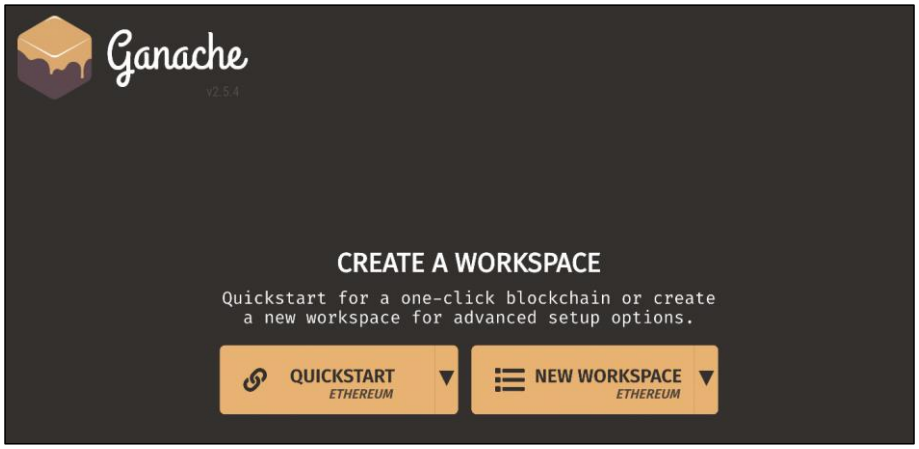

SIMPLECONTRACT2 AT 0X7D8...45BE
addition
100
difference
uint256 x
division
uint256 x
currValue
0: uint256: 105
Low level interactions
CALLDATA
Transact

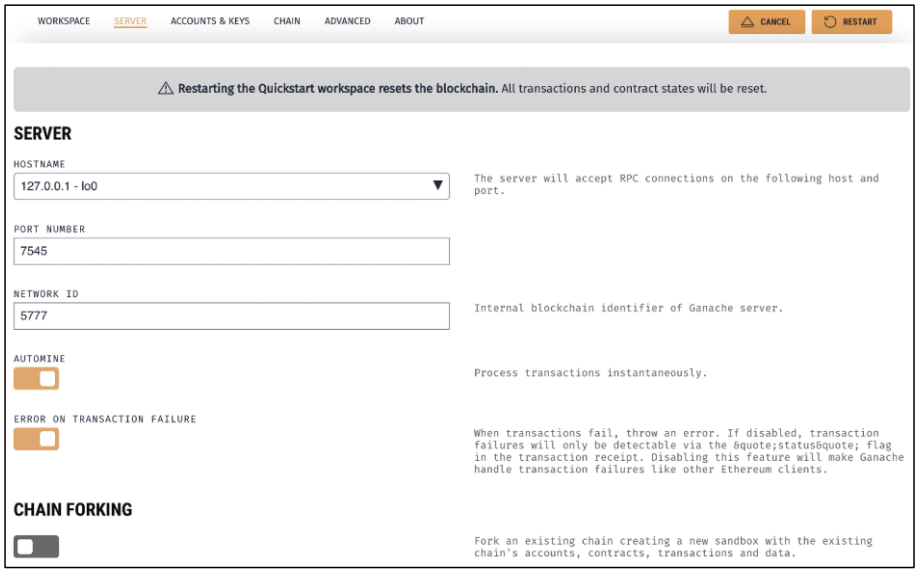

Localhost 8545
Assets
Activity
Addition
May 21 • remix.ethereum.org
-0 ETH
-0 ETH
Contract Deployment
May 21 • remix.ethereum.org
-0 ETH
-0 ETH
Addition
May 21 • remix.ethereum.org
-0 ETH
-0 ETH
Contract Deployment
May 21 • remix.ethereum.org
-0 ETH
-0 ETH
Contract Deployment
May 21 • remix.ethereum.org
-0 ETH
-0 ETH

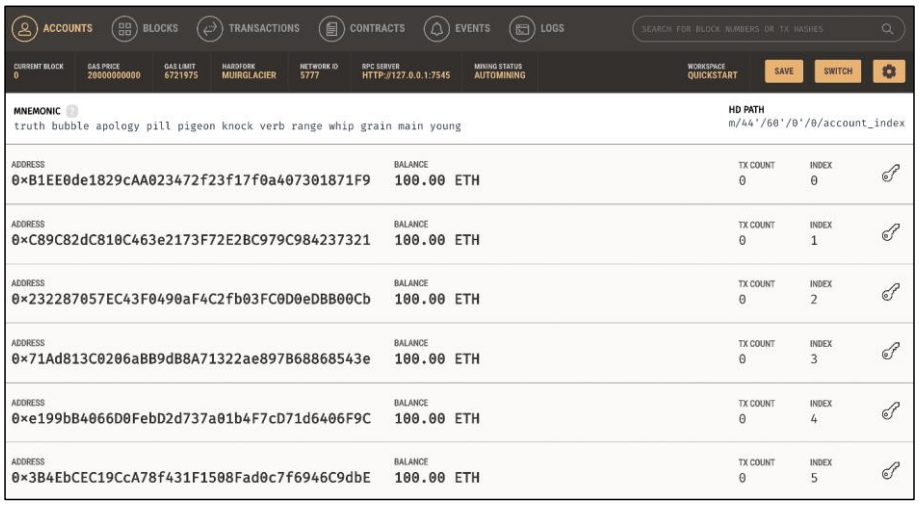

DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Web3 Provider
Custom (786) network
ACCOUNT
Ox6E9...af82f (11096.00413
GAS LIMIT
3000000
0
Wei
CONTRACT
SimpleContract2 - contracts/SimpleCo
Deploy
Publish to IPFS


External node request
Note: To use Geth & https://remix.ethereum.org, configure it to allow requests from Remix:(see Geth Docs on rpc server)
geth --http --http.corsdomain https://remix.ethereum.org
To run Remix & a local Geth test node, use this command: (see Geth Docs on Dev mode)
geth --http --http.corsdomain="https://remix.ethereum.org" --http.api web3,eth,debug,personal,net --vmdebug --datadir <path/to/local/folder/for/test/chain> --dev console
WARNING: It is not safe to use the --http.corsdomain flag with a wildcard: --http.corsdomain *
For more info: Remix Docs on Web3 Provider
Web3 Provider Endpoint
http://127.0.0.1:8545
OK
Cancel

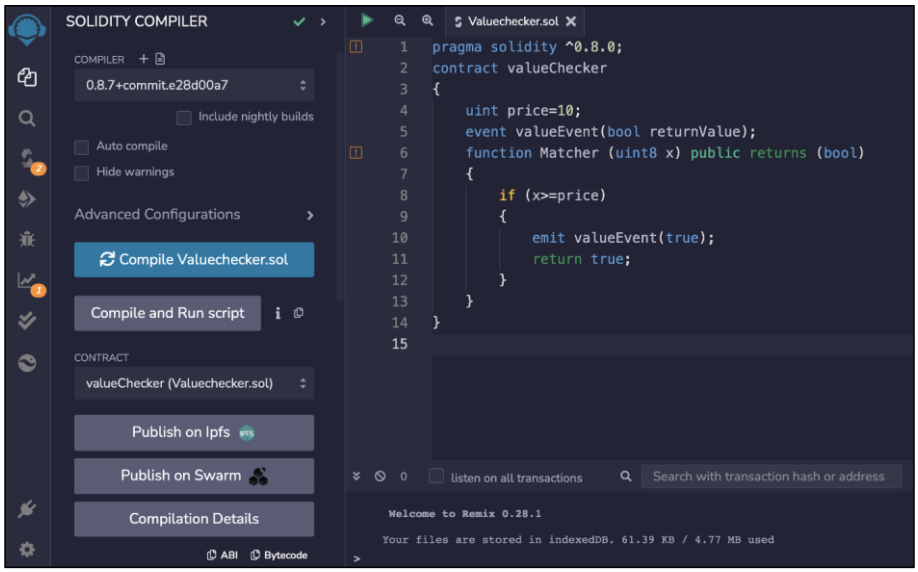
Chapter 11: Tools, Languages, and Frameworks for Ethereum Developers

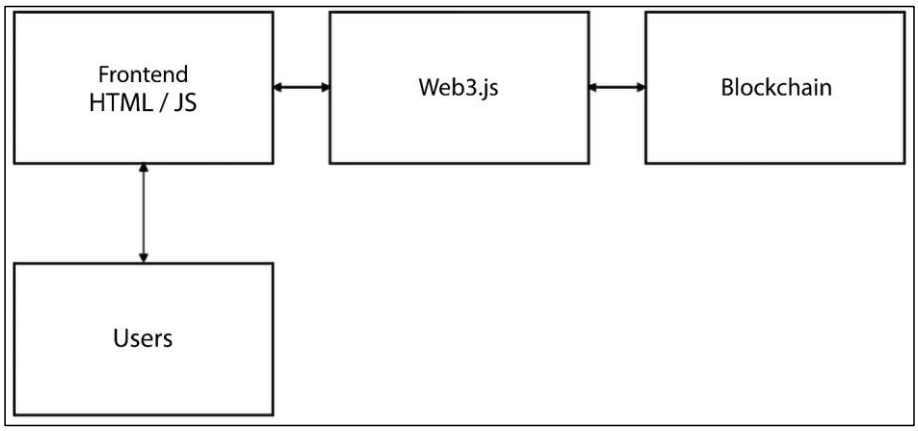


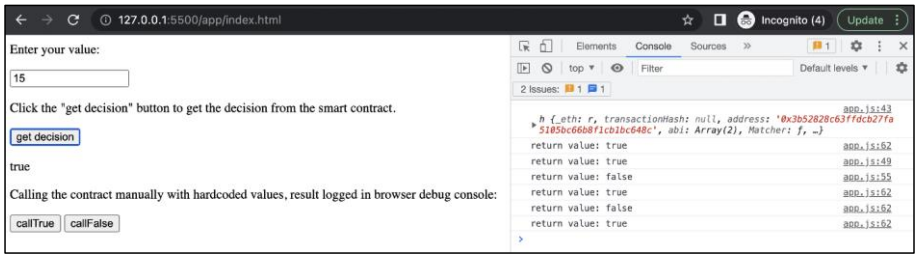


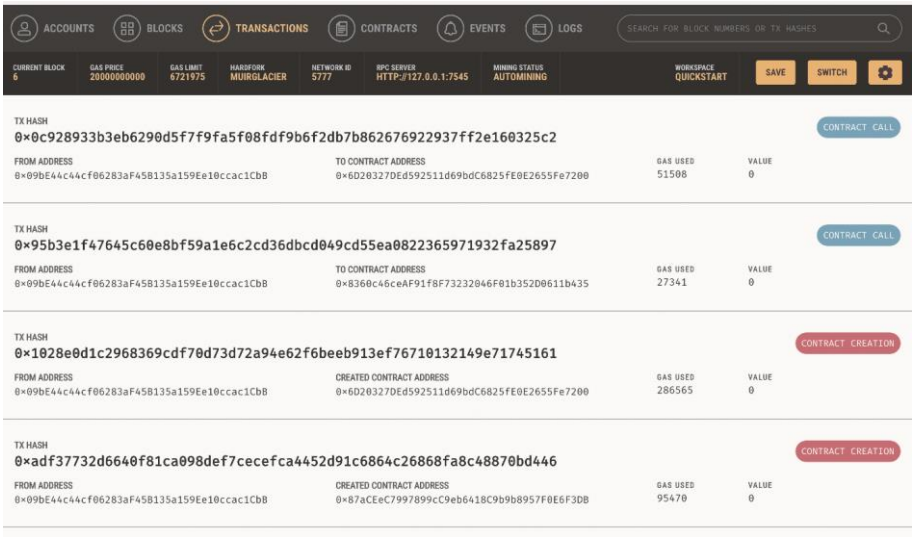

ACCOUNTS
BLOCKS
TRANSACTIONS
CONTRACTS
EVENTS
LOGS
SEARCH FOR BLOCK NUMBERS OR TX HASHES
CURRENT BLOCK 0
GAS PRICE 20000000000
GAS LIMIT 6721975
HARDFORK MUIRGLACIER
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING
WORKSPACE QUICKSTART
SAVE
SWITCH
MNEMONIC
truth bubble apology pill pigeon knock verb range whip grain main young
HD PATH
m/44'/60'/0'/0/account_index
ADDRESS 0x81EE0de1829cAA023472f23f17f0a407301871F9 BALANCE 100.00 ETH TX COUNT 0 INDEX 0
ADDRESS 0xC89C82dC810C463e2173F72E2BC979C984237321 BALANCE 100.00 ETH TX COUNT 0 INDEX 1
ADDRESS 0x232287057EC43F0490aF4C2fb03FC0D0eDBB00Cb BALANCE 100.00 ETH TX COUNT 0 INDEX 2
ADDRESS 0x71Ad813C0206aBB9dB8A71322ae897B68868543e BALANCE 100.00 ETH TX COUNT 0 INDEX 3
ADDRESS 0xe199bB4066D0FebD2d737a01b4F7cD71d6406F9C BALANCE 100.00 ETH TX COUNT 0 INDEX 4
ADDRESS 0x3B4EbCEC19CcA78f431F1508Fad0c7f6946C9dbE BALANCE 100.00 ETH TX COUNT 0 INDEX 5

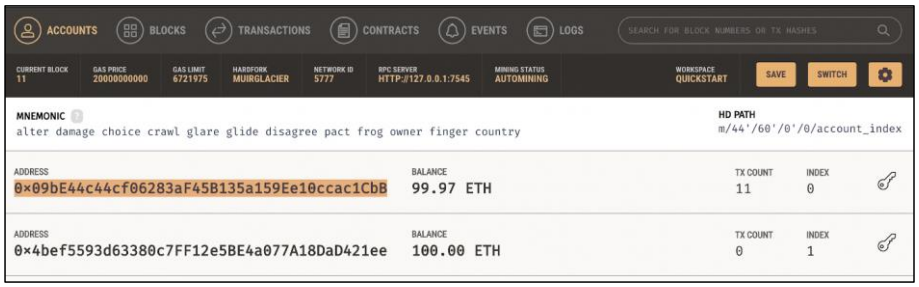
Chapter 12: Web3 Development Using Ethereum

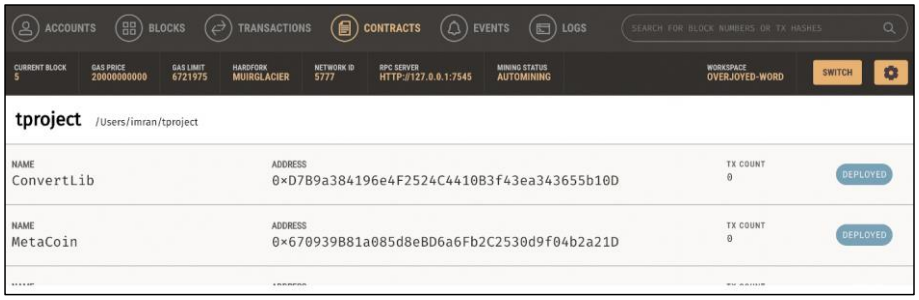


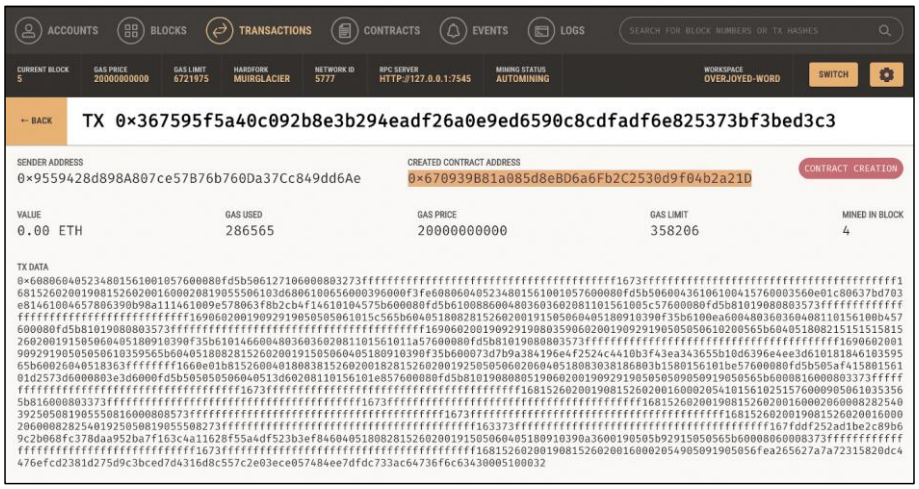




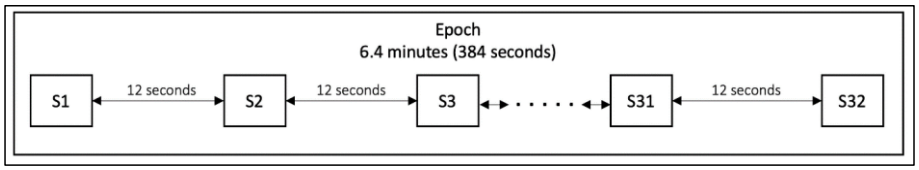

ACCOUNTS
BLOCKS
TRANSACTIONS
CONTRACTS
EVENTS
LOGS
SEARCH FOR BLOCK NUMBERS OR TX HASHES
CURRENT BLOCK 6
GAS PRICE 20000000000
GAS LIMIT 6721975
HARDFORK MUIRGLACIER
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING
WORKSPACE QUICKSTART
SAVE
SWITCH
TX HASH
0x0c928933b3eb6290d5f7f9fa5f08fdf9b6f2db7b862676922937ff2e160325c2
CONTRACT CALL
FROM ADDRESS
0x09bE44c44cf06283aF45B135a159Ee10ccac1CbB
TO CONTRACT ADDRESS
0x6D20327DEd592511d69bdC6825fE0E2655Fe7200
GAS USED
51508
VALUE
0
TX HASH
0x95b3e1f47645c60e8bf59a1e6c2cd36dbcd049cd55ea0822365971932fa25897
CONTRACT CALL
FROM ADDRESS
0x09bE44c44cf06283aF45B135a159Ee10ccac1CbB
TO CONTRACT ADDRESS
0x8360c46ceAF91f8F73232046F01b352D0611b435
GAS USED
27341
VALUE
0
TX HASH
0x1028e0d1c2968369cdf70d73d72a94e62f6beeb913ef76710132149e71745161
CONTRACT CREATION
FROM ADDRESS
0x09bE44c44cf06283aF45B135a159Ee10ccac1CbB
CREATED CONTRACT ADDRESS
0x6D20327DEd592511d69bdC6825fE0E2655Fe7200
GAS USED
286565
VALUE
0
TX HASH
0xadf37732d6640f81ca098def7cecefca4452d91c6864c26868fa8c48870bd446
CONTRACT CREATION
FROM ADDRESS
0x09bE44c44cf06283aF45B135a159Ee10ccac1CbB
CREATED CONTRACT ADDRESS
0x87aCEeC7997899cC9eb6418C9b9b8957F0E6F3DB
GAS USED
95470
VALUE
0

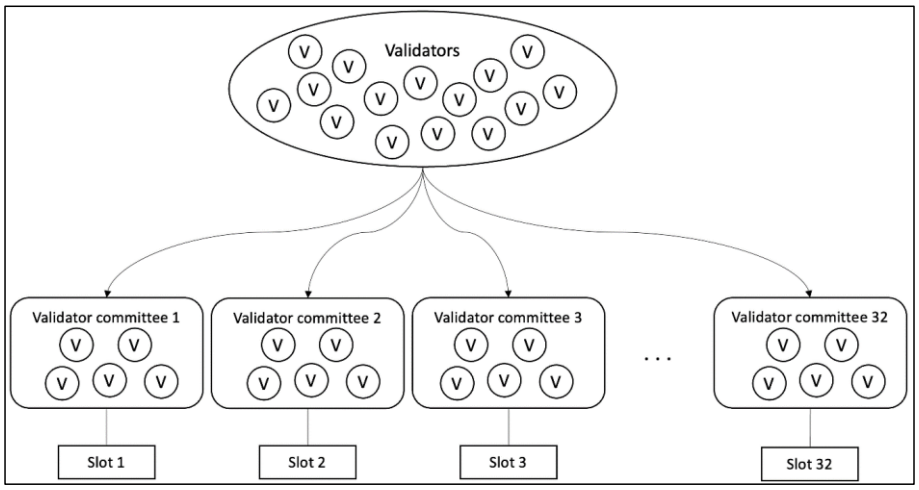

ACCOUNTS
BLOCKS
TRANSACTIONS
CONTRACTS
EVENTS
LOGS
SEARCH FOR BLOCK NUMBERS OR TX HASHES
CURRENT BLOCK 11
GAS PRICE 20000000000
GAS LIMIT 6721975
HARDFORK MUIRGLACIER
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING
WORKSPACE QUICKSTART
SAVE
SWITCH
MNEMONIC
alter damage choice crawl glare glide disagree pact frog owner finger country
HD PATH
m/44'/60'/0'/0/account_index
ADDRESS
0x09bE44c44cf06283aF45B135a159Ee10ccac1CbB
BALANCE
99.97 ETH
TX COUNT
11
INDEX
0
ADDRESS
0x4bef5593d63380c7FF12e5BE4a077A18DaD421ee
BALANCE
100.00 ETH
TX COUNT
0
INDEX
1

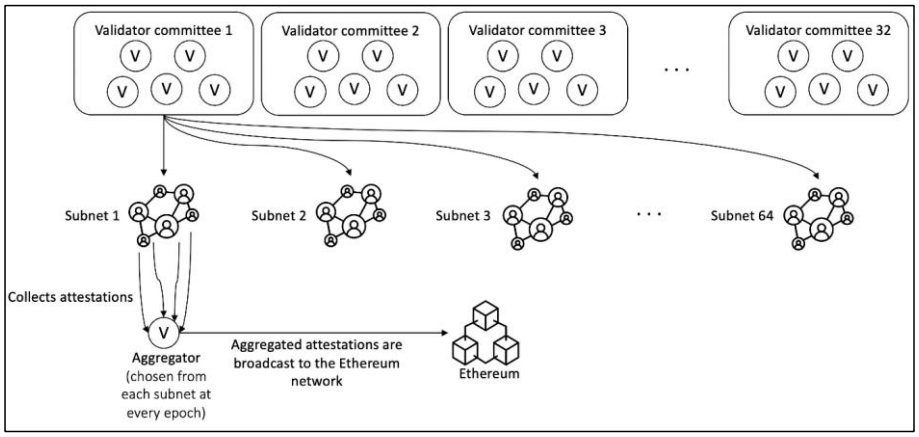

ACCOUNTS
BLOCKS
TRANSACTIONS
CONTRACTS
EVENTS
LOGS
SEARCH FOR BLOCK NUMBERS OR TX HASHES
CURRENT BLOCK 5
GAS PRICE 20000000000
GAS LIMIT 6721975
HARDFORK MUIRGLACIER
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING
WORKSPACE OVERJOYED-WORD
SWITCH
tproject
/Users/imran/tproject
NAME
ConvertLib
ADDRESS
0xD7B9a384196e4F2524C4410B3f43ea343655b10D
TX COUNT
0
DEPLOYED
NAME
MetaCoin
ADDRESS
0x670939B81a085d8eBD6a6Fb2C2530d9f04b2a21D
TX COUNT
0
DEPLOYED

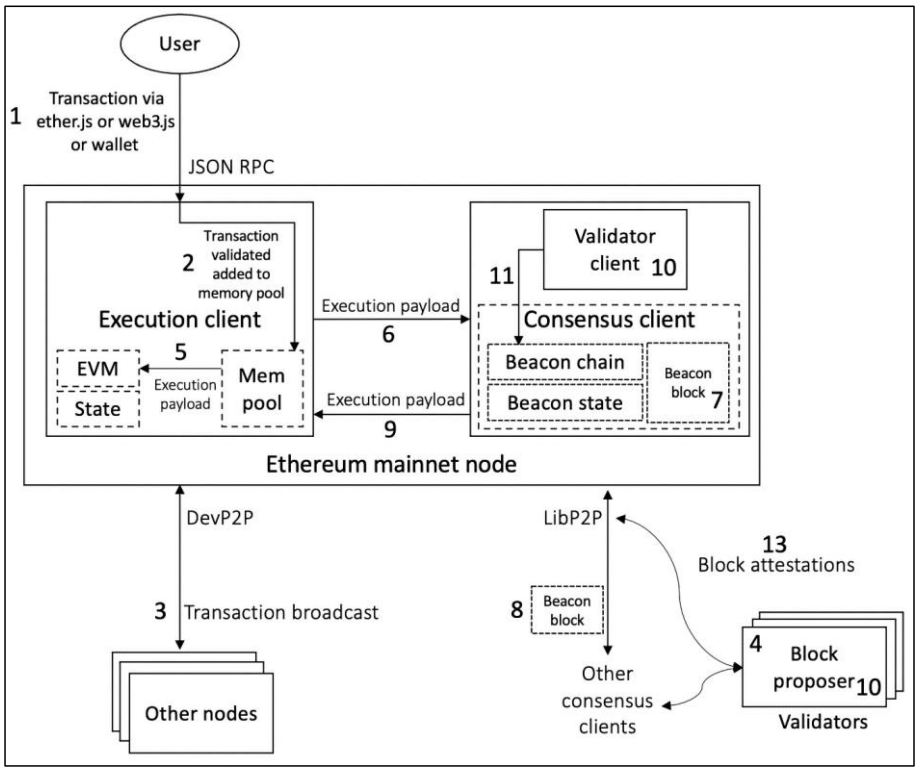

ACCOUNTS
BLOCKS
TRANSACTIONS
CONTRACTS
EVENTS
LOGS
SEARCH FOR BLOCK NUMBERS OR TX HASHES
CURRENT BLOCK 5
GAS PRICE 20000000000
GAS LIMIT 6721975
HARDFORK MUIRGLACIER
NETWORK ID 5777
RPC SERVER HTTP://127.0.0.1:7545
MINING STATUS AUTOMINING
WORKSPACE OVERJOYED-WORD
SWITCH
← BACK
TX 0×367595f5a40c092b8e3b294eadf26a0e9ed6590c8cdfadf6e825373bf3bed3c3
SENDER ADDRESS
0×9559428d898A807ce57B76b760Da37Cc849dd6Ae
CREATED CONTRACT ADDRESS
0×670939B81a085d8eBD6a6Fb2C2530d9f04b2a21D
CONTRACT CREATION
VALUE
0.00 ETH
GAS USED
286565
GAS PRICE
20000000000
GAS LIMIT
358206
MINED IN BLOCK
4
TX DATA

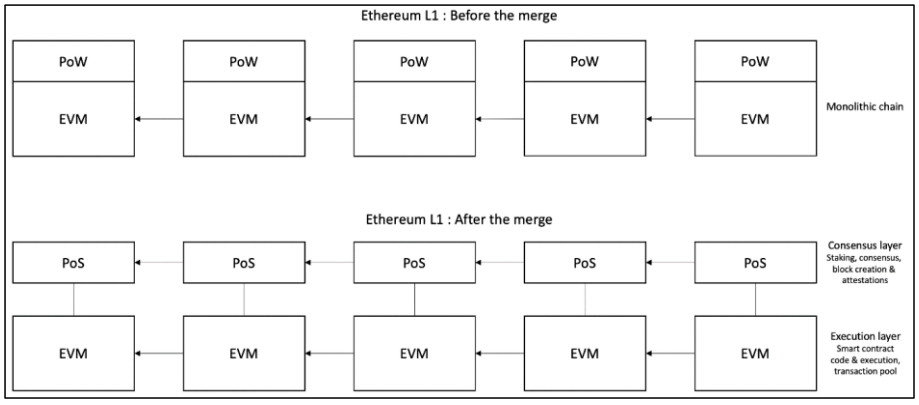
Chapter 13: The Merge and Beyond

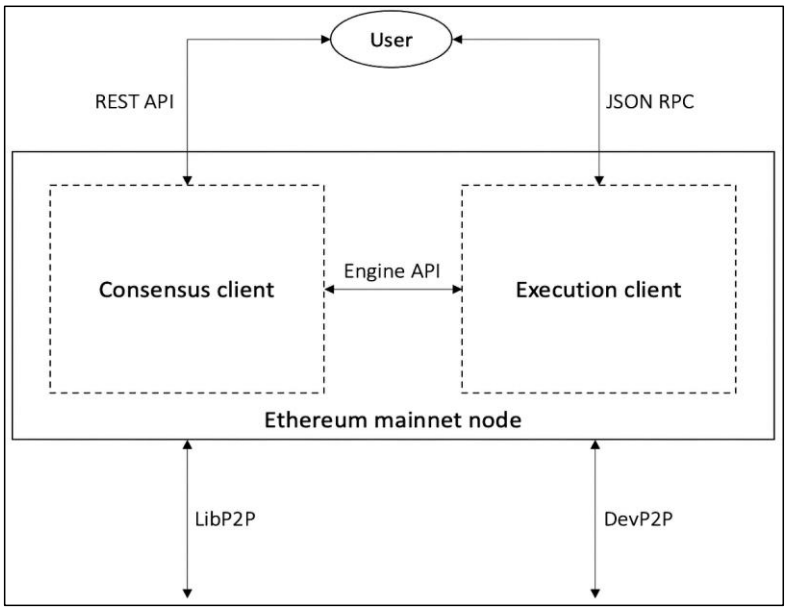


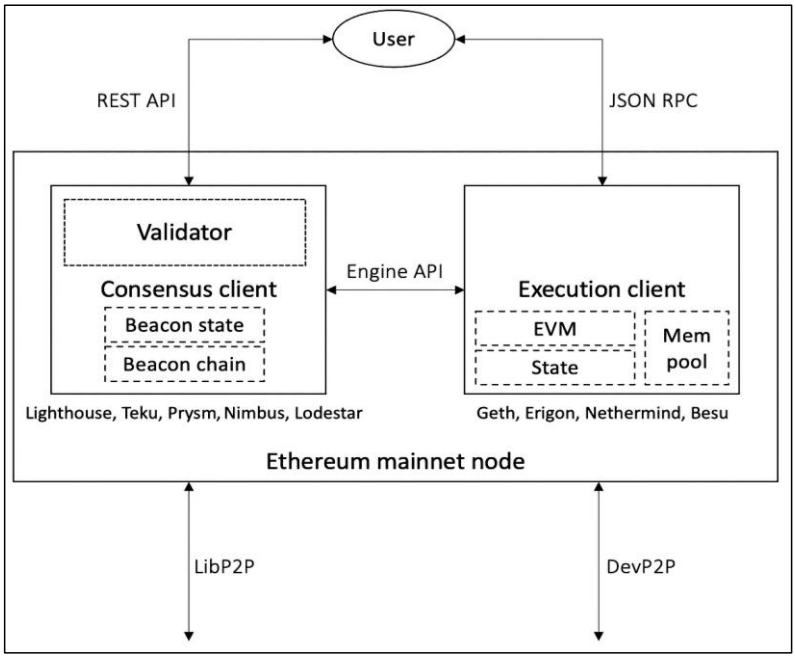


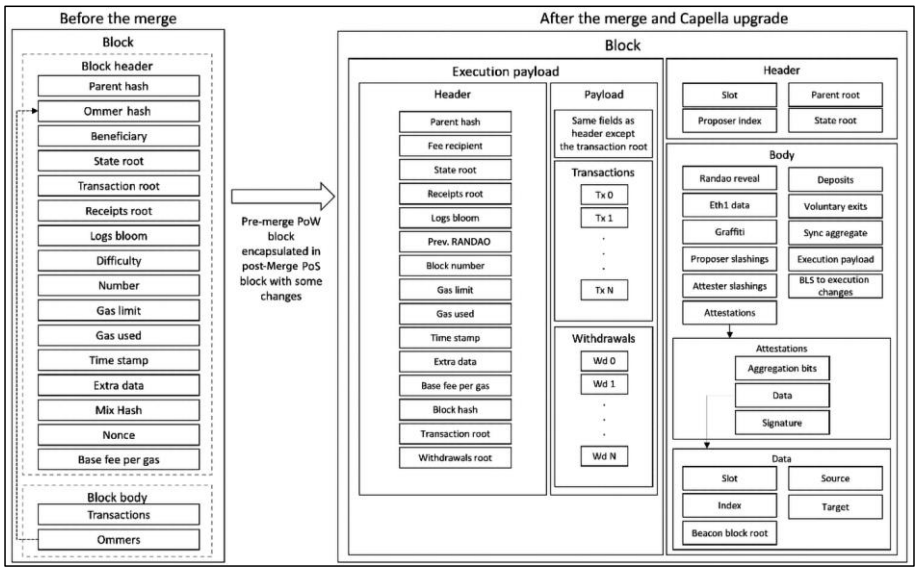


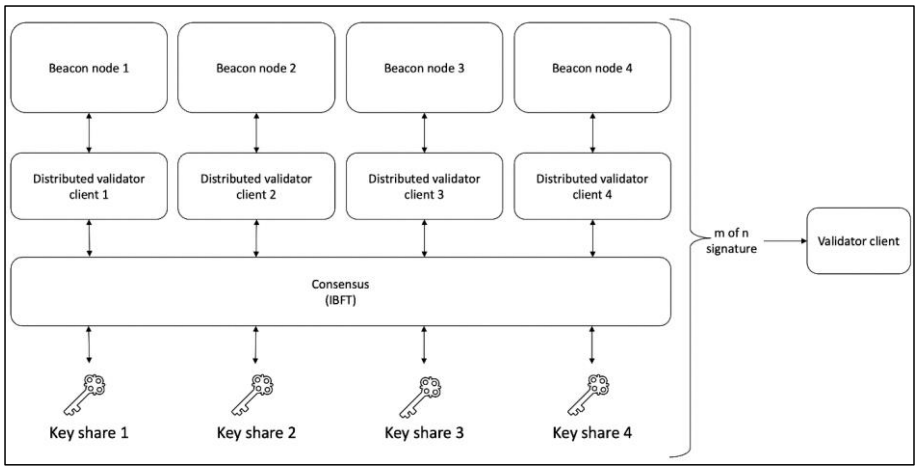

User
1
Transaction via ether.js or web3.js or wallet
JSON RPC
2
Transaction validated added to memory pool
Execution client
5
EVM
State
Execution payload
Mem pool
Execution payload
6
Execution payload
9
Validator client 10
11
Consensus client
Beacon chain
Beacon state
Beacon block 7
Ethereum mainnet node
DevP2P
3
Transaction broadcast
Other nodes
LibP2P
13
Block attestations
8
Beacon block
Other consensus clients
4
Block proposer 10
Validators

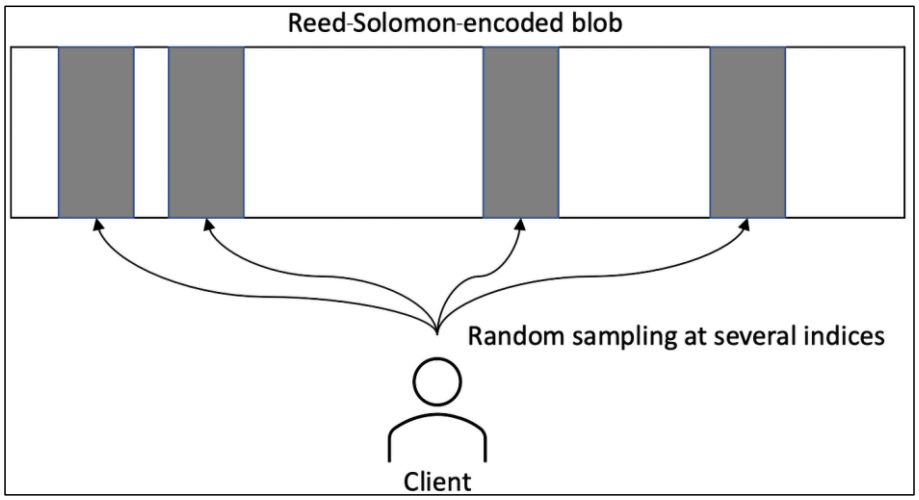

Ethereum L1 : Before the merge
PoW
EVM
Monolithic chain
Ethereum L1 : After the merge
PoS
Consensus layer
Staking, consensus, block creation & attestations
EVM
Execution layer
Smart contract code & execution, transaction pool

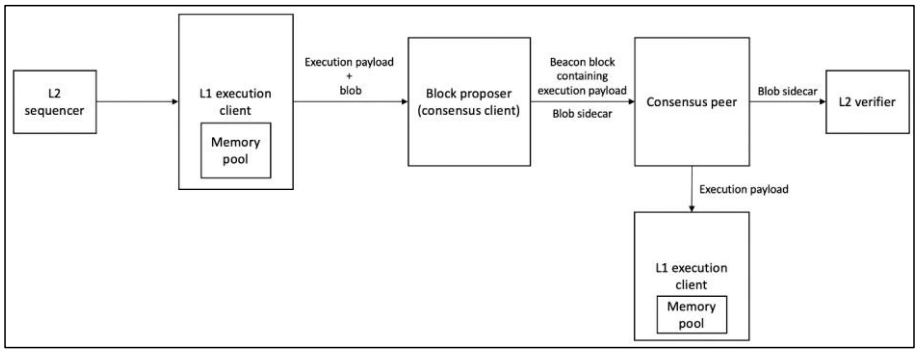

User
REST API
JSON RPC
Consensus client
Engine API
Execution client
Ethereum mainnet node
LibP2P
DevP2P

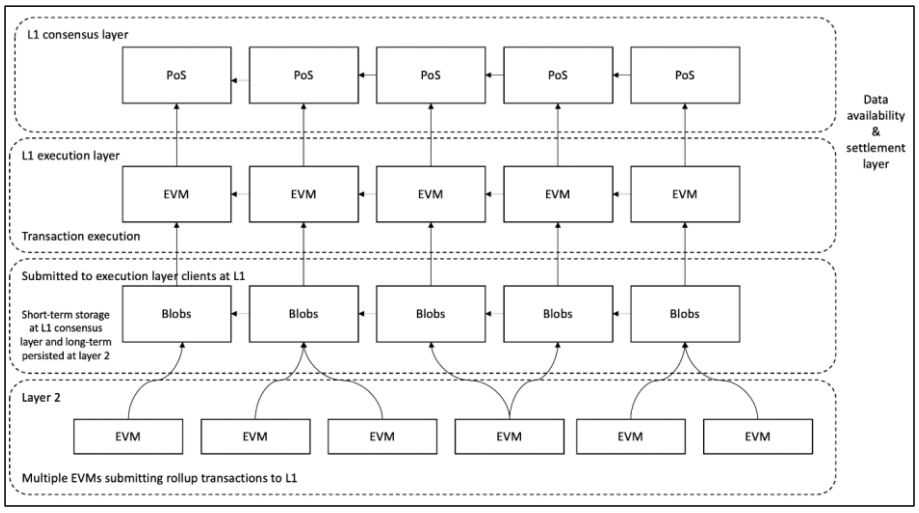

User
REST API
JSON RPC
Validator
Consensus client
Beacon state
Beacon chain
Engine API
Execution client
EVM
State
Mem pool
Lighthouse, Teku, Prysm, Nimbus, Lodestar
Geth, Erigon, Nethermind, Besu
Ethereum mainnet node
LibP2P
DevP2P

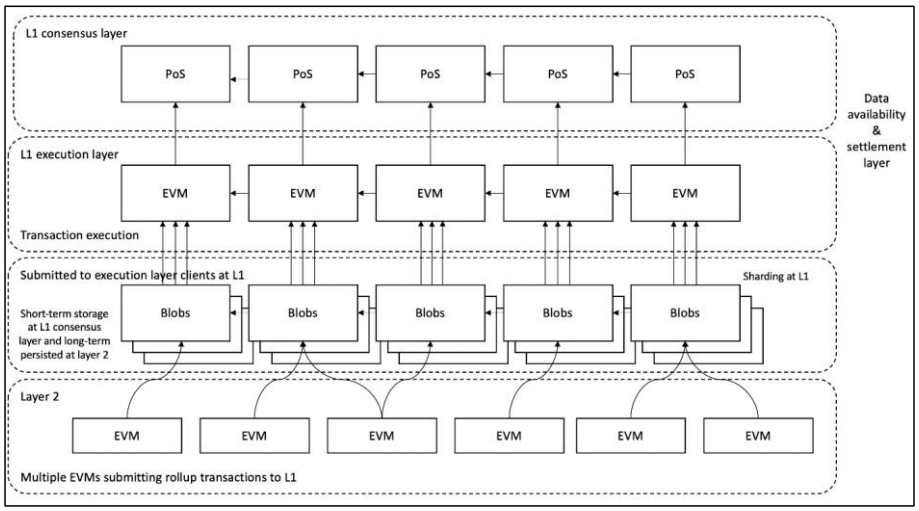

Before the merge
Block
Block header
Parent hash
Ommer hash
Beneficiary
State root
Transaction root
Receipts root
Logs bloom
Difficulty
Number
Gas limit
Gas used
Time stamp
Extra data
Mix Hash
Nonce
Base fee per gas
Block body
Transactions
Ommers
Pre-merge PoW block encapsulated in post-Merge PoS block with some changes
After the merge and Capella upgrade
Block
Execution payload
Header
Parent hash
Fee recipient
State root
Receipts root
Logs bloom
Prev. RANDAO
Block number
Gas limit
Gas used
Time stamp
Extra data
Base fee per gas
Block hash
Transaction root
Withdrawals root
Payload
Same fields as header except the transaction root
Transactions
Tx 0
Tx 1
Tx N
Withdrawals
Wd 0
Wd 1
Wd N
Header
Slot
Parent root
Proposer index
State root
Body
Randao reveal
Deposits
Eth1 data
Voluntary exits
Graffiti
Sync aggregate
Proposer slashings
Execution payload
Attester slashings
BLS to execution changes
Attestations
Attestations
Aggregation bits
Data
Signature
Data
Slot
Source
Index
Target
Beacon block root


Beacon node 1
Beacon node 2
Beacon node 3
Beacon node 4
Distributed validator client 1
Distributed validator client 2
Distributed validator client 3
Distributed validator client 4
Consensus (IBFT)
m of n signature
Validator client
Key share 1
Key share 2
Key share 3
Key share 4

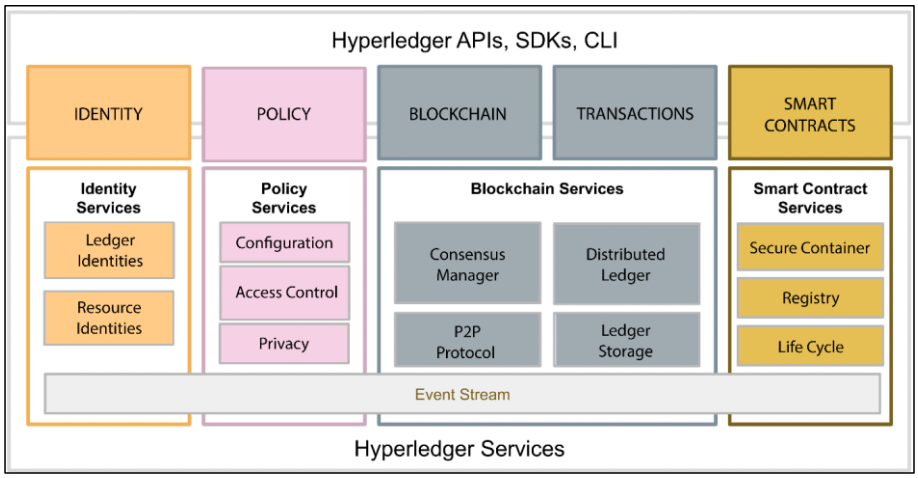

Reed-Solomon-encoded blob
Random sampling at several indices
Client

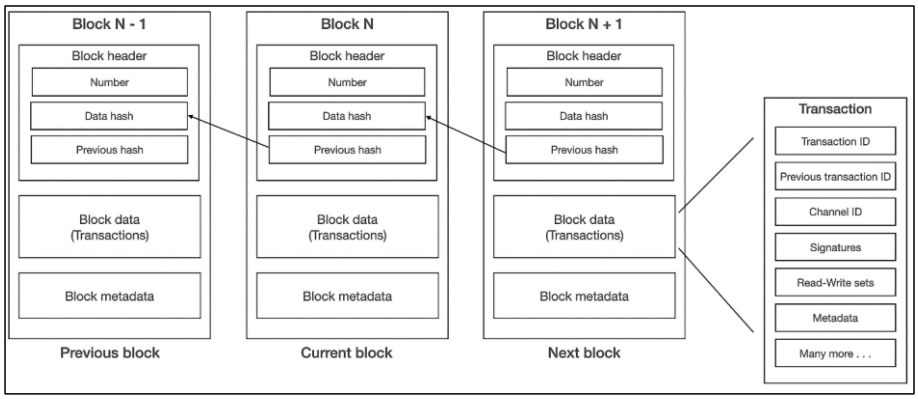

L2 sequencer
L1 execution client
Memory pool
Execution payload + blob
Block proposer (consensus client)
Beacon block containing execution payload
Blob sidecar
Consensus peer
Blob sidecar
L2 verifier
Execution payload
L1 execution client
Memory pool

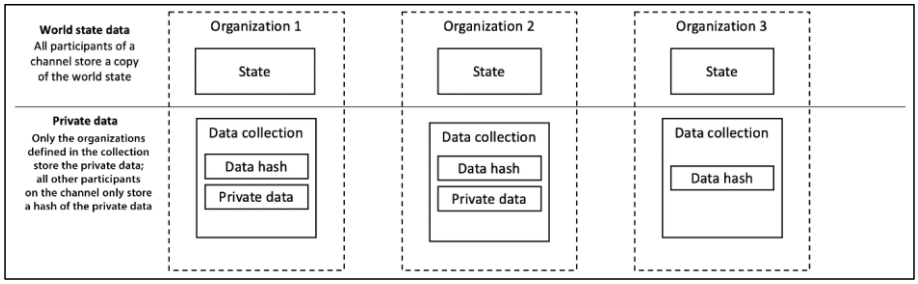

L1 consensus layer
PoS
PoS
PoS
PoS
PoS
Data availability & settlement layer
L1 execution layer
EVM
EVM
EVM
EVM
EVM
Transaction execution
Submitted to execution layer clients at L1
Short-term storage at L1 consensus layer and long-term persisted at layer 2
Blobs
Blobs
Blobs
Blobs
Blobs
Layer 2
EVM
EVM
EVM
EVM
EVM
EVM
Multiple EVMs submitting rollup transactions to L1

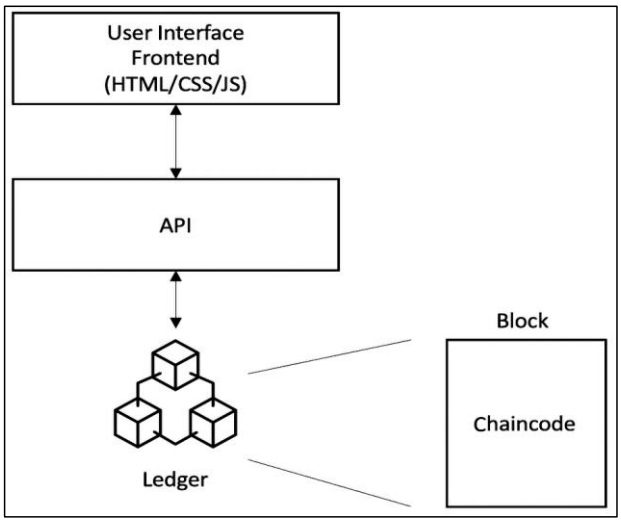

L1 consensus layer
PoS
PoS
PoS
PoS
PoS
Data availability & settlement layer
L1 execution layer
EVM
EVM
EVM
EVM
EVM
Transaction execution
Submitted to execution layer clients at L1
Sharding at L1
Short-term storage at L1 consensus layer and long-term persisted at layer 2
Blobs
Blobs
Blobs
Blobs
Blobs
Layer 2
EVM
EVM
EVM
EVM
EVM
EVM
Multiple EVMs submitting rollup transactions to L1

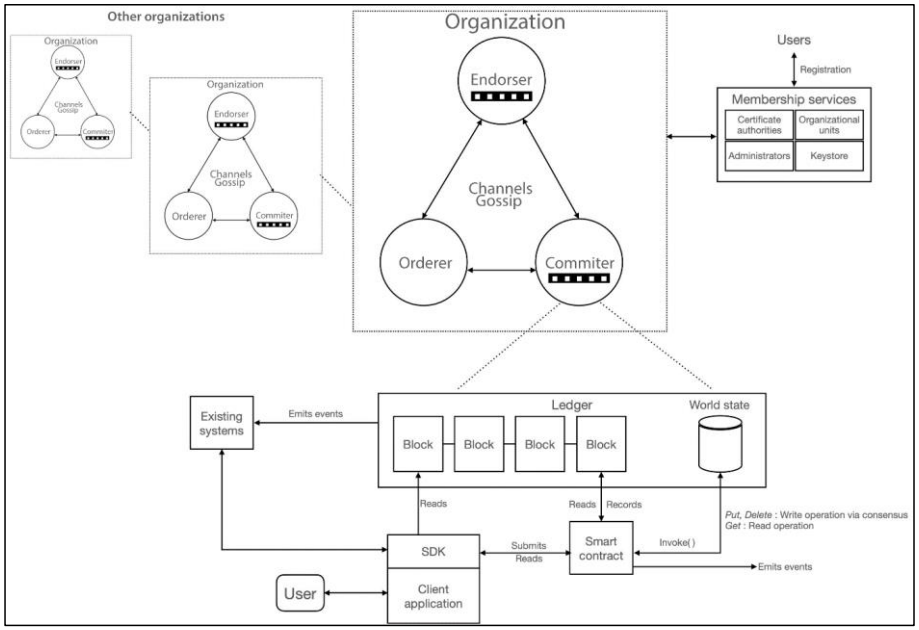
Chapter 14: Hyperledger

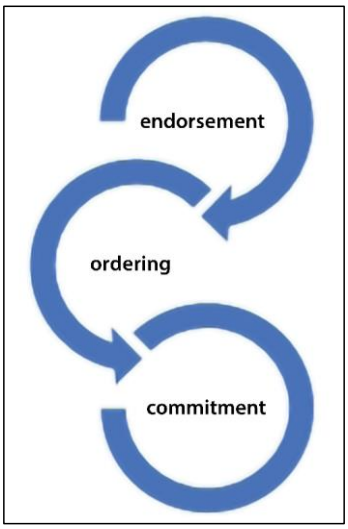


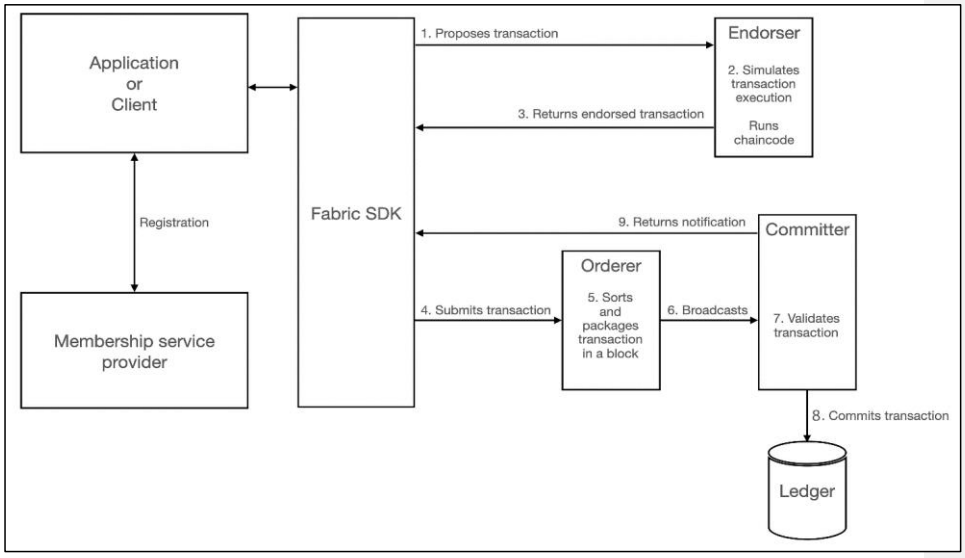


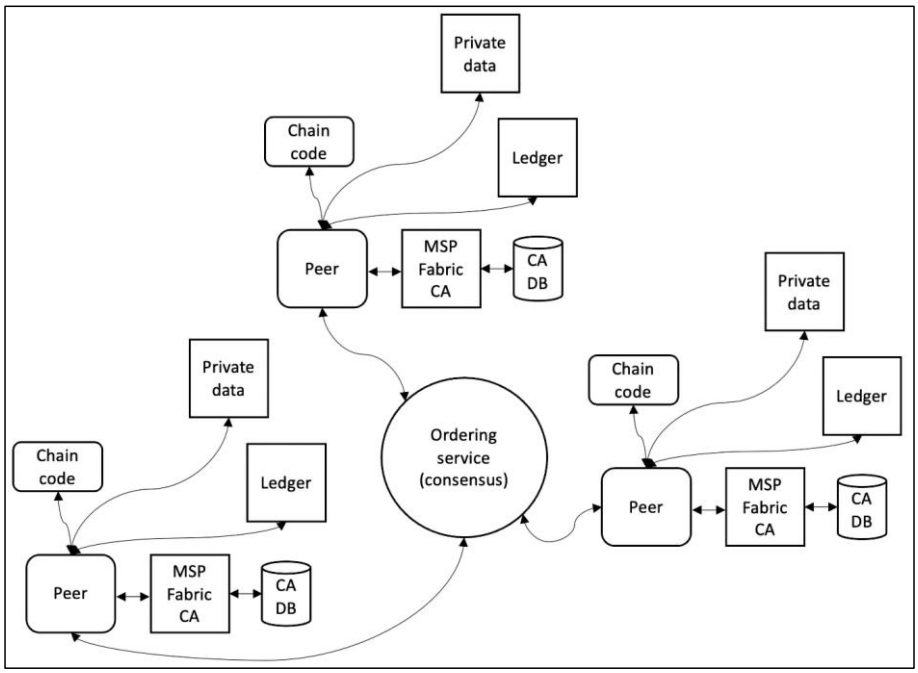



User Interface
Frontend
(HTML/CSS/JS)
API
Block
Chaincode
Ledger

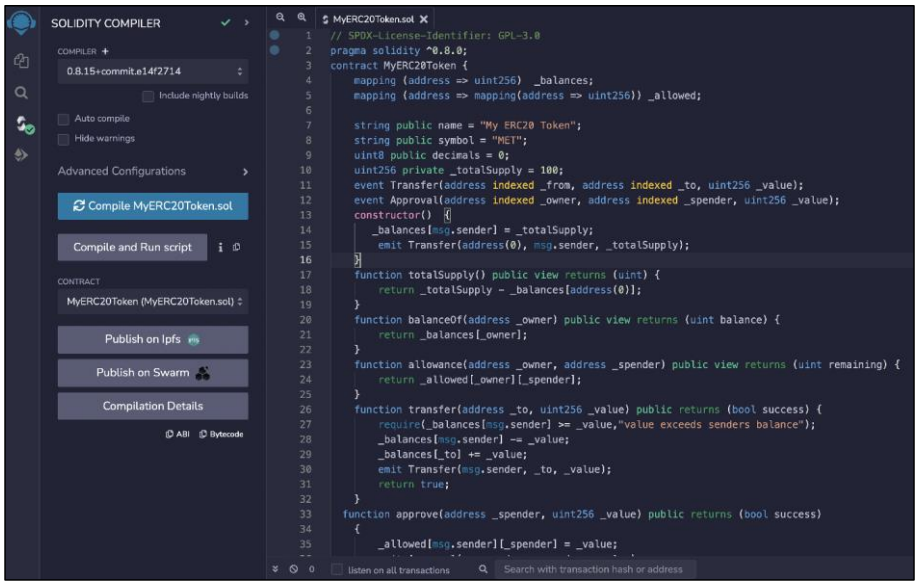

Other organizations
Organization
Endorser
Channels Gossip
Orderer
Commiter
Organization
Endorser
Channels Gossip
Orderer
Commiter
Organization
Endorser
Channels Gossip
Orderer
Commiter
Users
Registration
Membership services
Certificate authorities
Organizational units
Administrators
Keystore
Existing systems
Emits events
Ledger
World state
Block
Block
Block
Block
Reads
Reads
Records
Put, Delete : Write operation via consensus
Get : Read operation
SDK
Submits
Reads
Smart contract
Invoke( )
Emits events
User
Client application

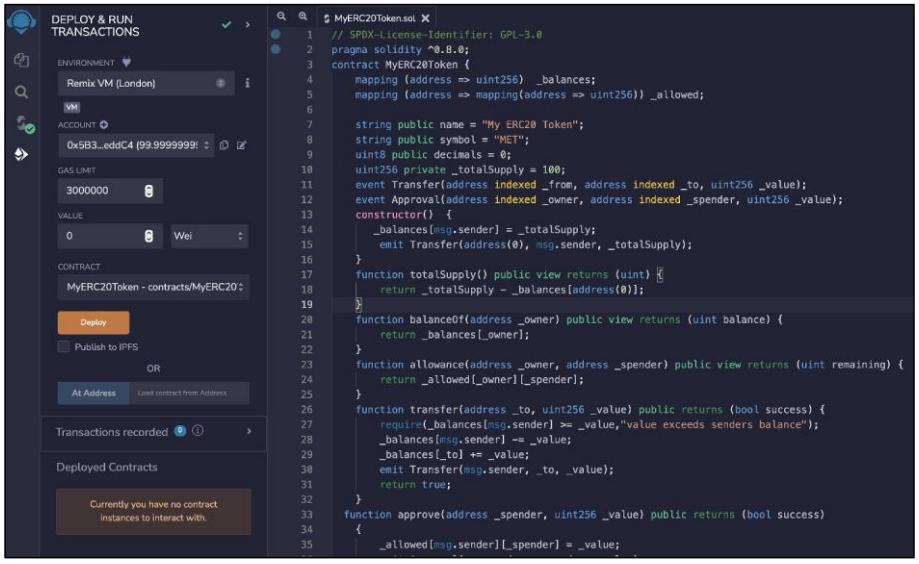

endorsement
ordering
commitment

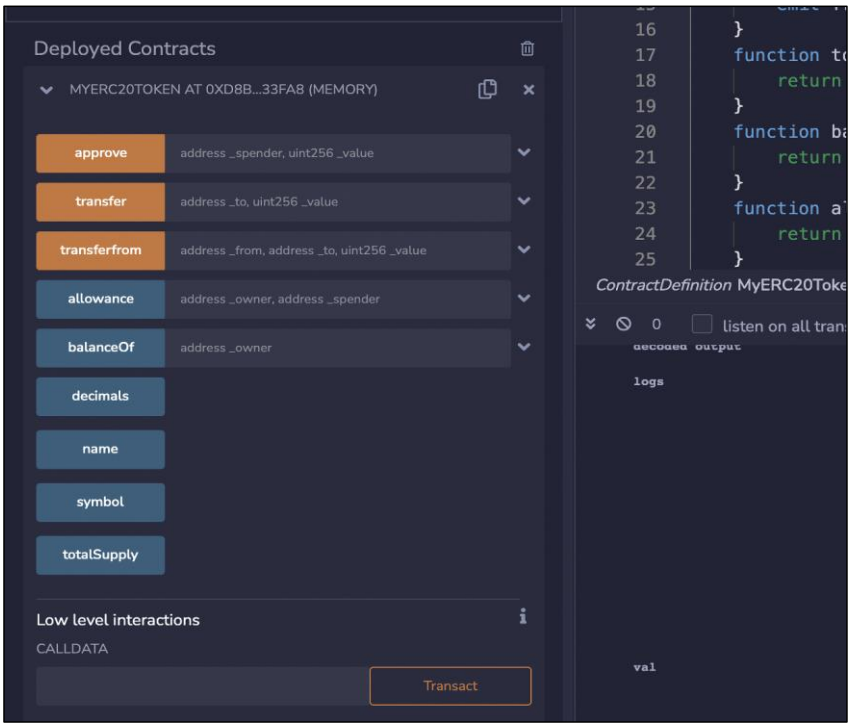

Application
or
Client
Registration
Membership service
provider
Fabric SDK
1. Proposes transaction
Endorser
2. Simulates
transaction
execution
Runs
chaincode
3. Returns endorsed transaction
9. Returns notification
Committer
Orderer
5. Sorts
and
packages
transaction
in a block
4. Submits transaction
6. Broadcasts
7. Validates
transaction
8. Commits transaction
Ledger

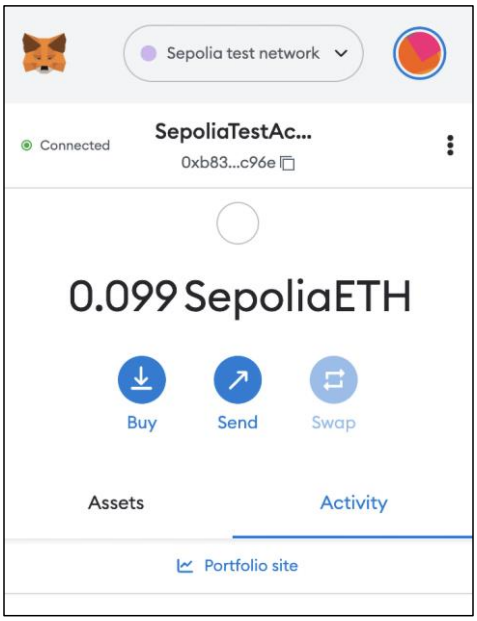

Private data
Chain code
Ledger
Peer
MSP Fabric CA
CA DB
Private data
Chain code
Ledger
Private data
Chain code
Ledger
Ordering service (consensus)
Peer
MSP Fabric CA
CA DB
Peer
MSP Fabric CA
CA DB

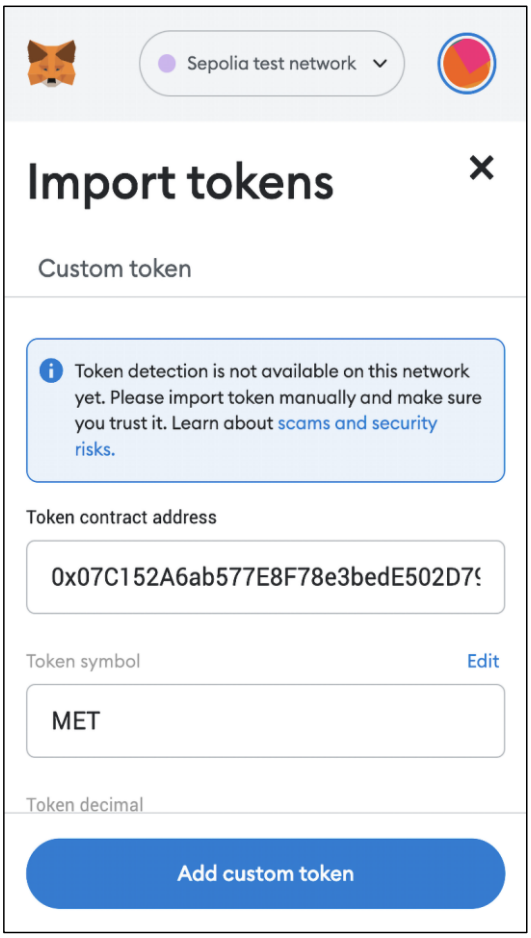
Chapter 15: Tokenization



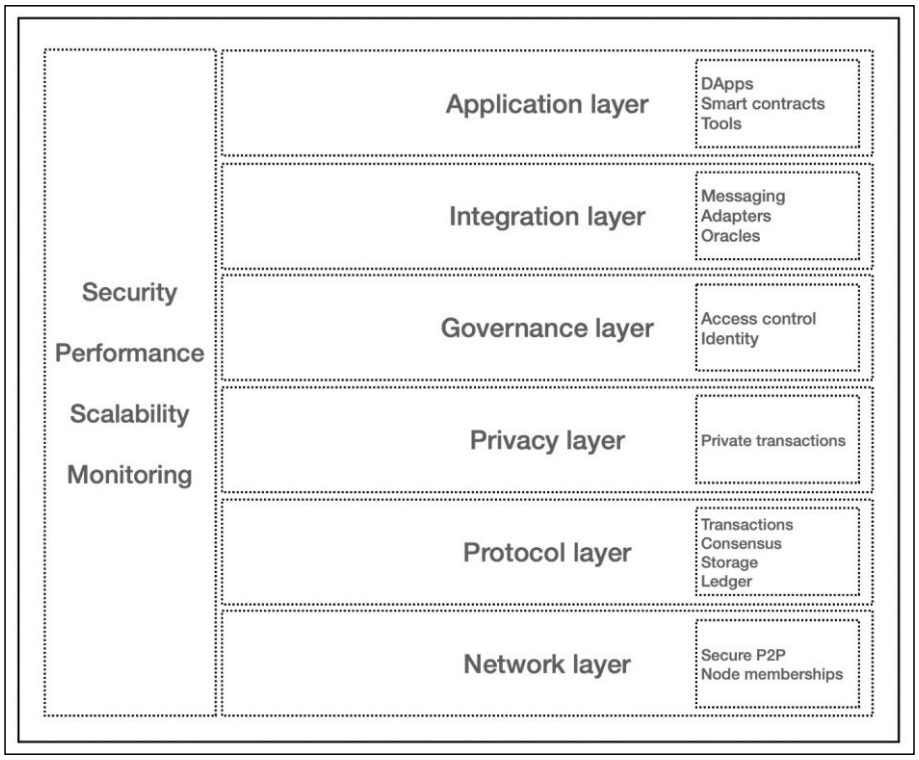


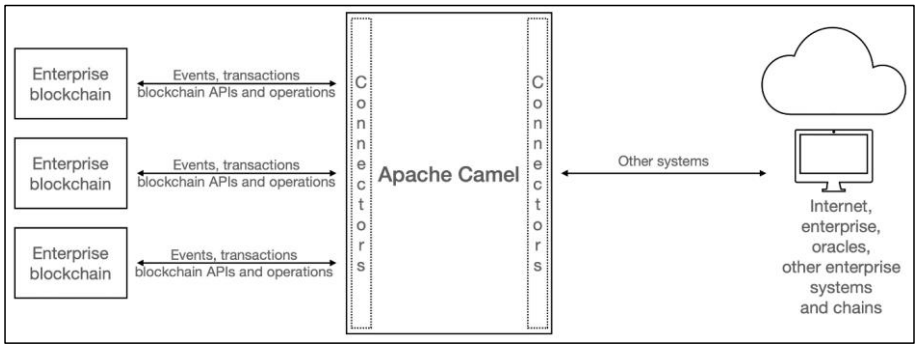

Deployed Contracts
MYERC20TOKEN AT 0XD8B...33FA8 (MEMORY)
approve
address _spender, uint256 _value
transfer
address _to, uint256 _value
transferfrom
address _from, address _to, uint256 _value
allowance
address _owner, address _spender
balanceOf
address _owner
decimals
name
symbol
totalSupply
Low level interactions
CALLDATA
Transact
16 }
17 function t
18 return
19 }
20 function b
21 return
22 }
23 function a
24 return
25 }
ContractDefinition MyERC20Toke
0
listen on all tran
logs
val

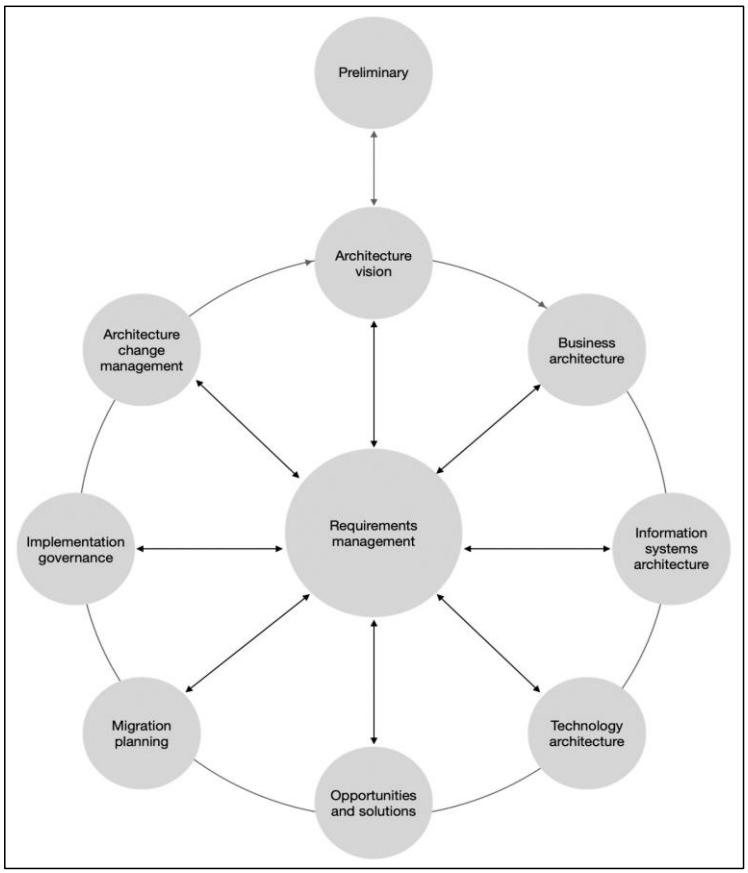

Sepolia test network
Connected
SepoliaTestAc...
0xb83...c96e
0.099 SepoliaETH
Buy
Send
Swap
Assets
Activity
Portfolio site

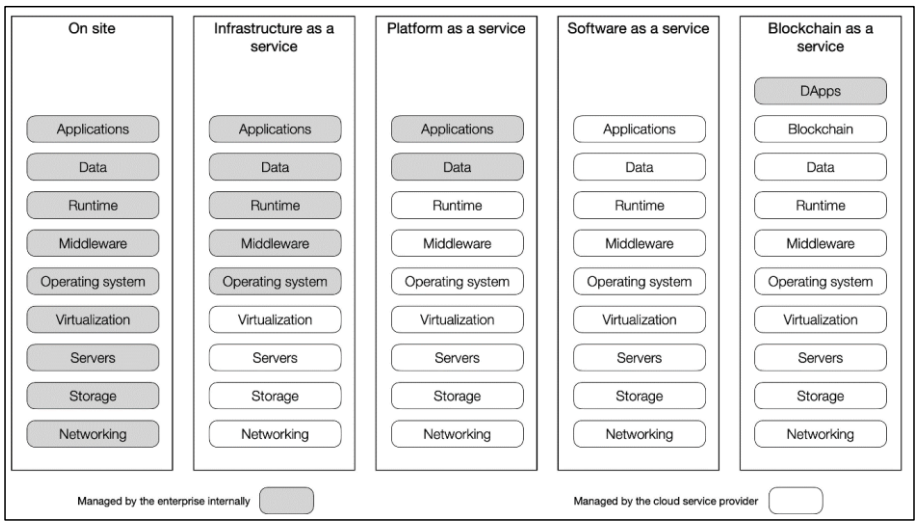

Sepolia test network
Import tokens
Custom token
Token detection is not available on this network yet. Please import token manually and make sure you trust it. Learn about scams and security risks.
Token contract address
0x07C152A6ab577E8F78e3bedE502D7
Token symbol
Edit
MET
Token decimal
Add custom token

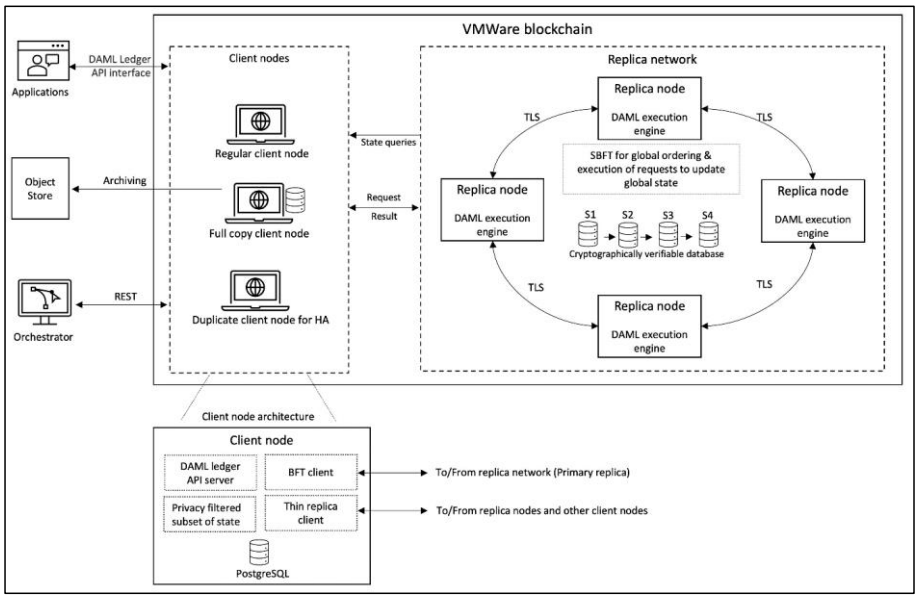
Chapter 16: Enterprise Blockchain

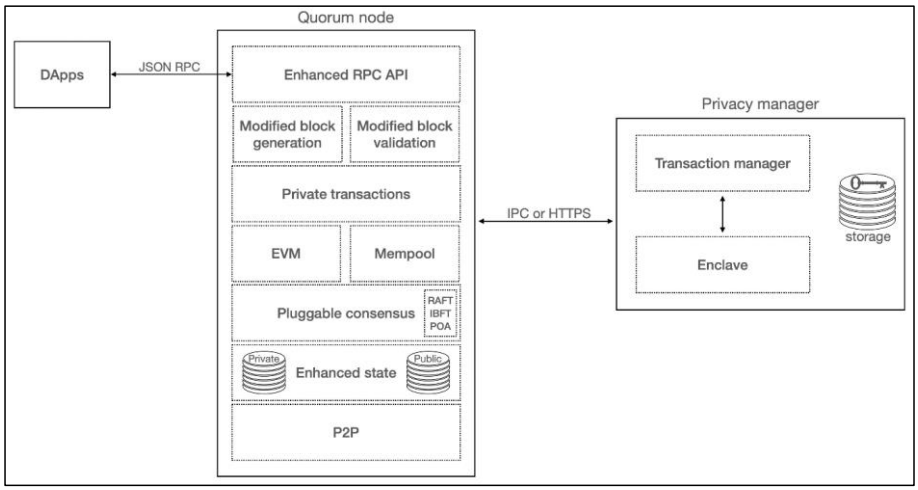


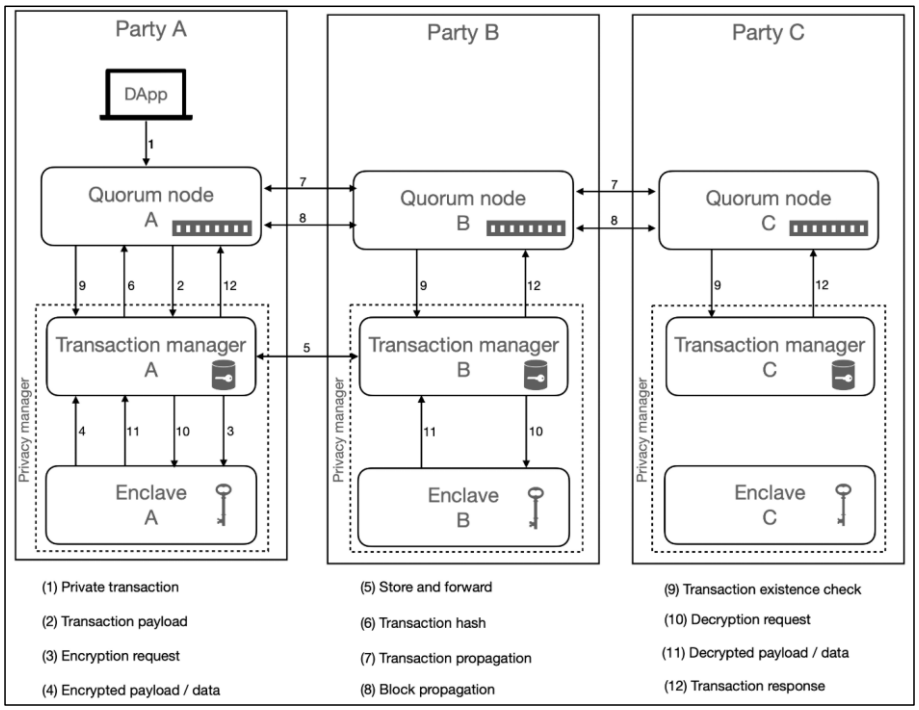


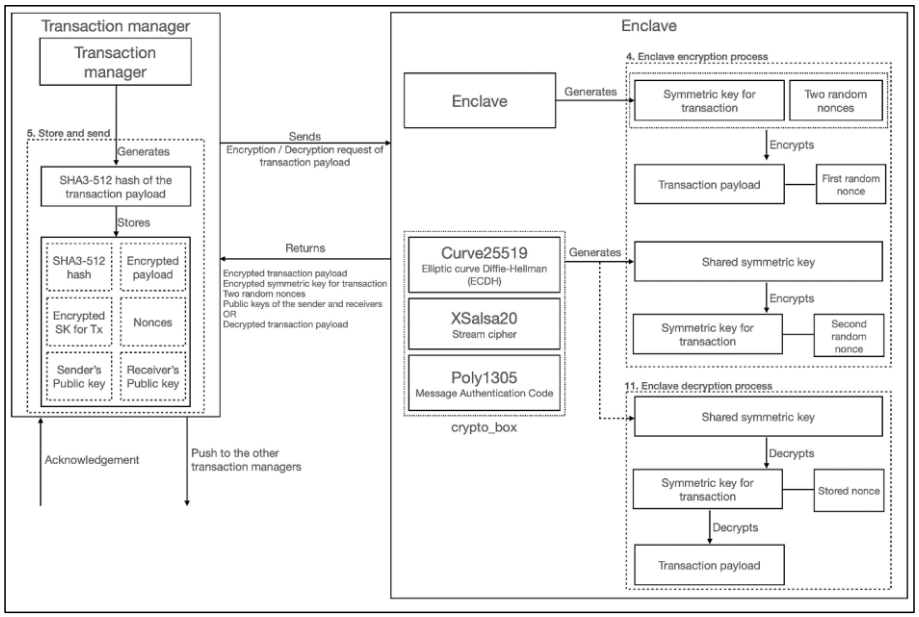

Preliminary
Architecture vision
Business architecture
Technology architecture
Opportunities and solutions
Migration planning
Implementation governance
Architecture change management
Requirements management
Information systems architecture

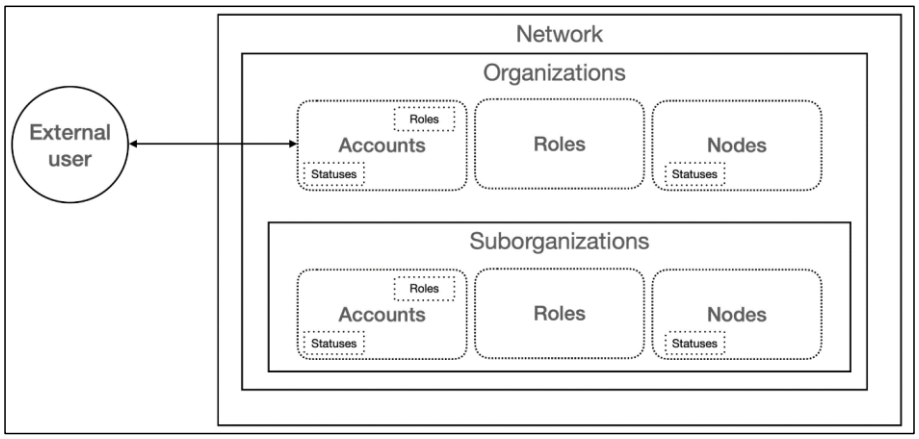

On site
Infrastructure as a service
Platform as a service
Software as a service
Blockchain as a service
DApps
Applications
Blockchain
Data
Runtime
Middleware
Operating system
Virtualization
Servers
Storage
Networking
Managed by the enterprise internally
Managed by the cloud service provider


VMWare blockchain
Applications
DAML Ledger API interface
Client nodes
Regular client node
Full copy client node
Duplicate client node for HA
Object Store
Archiving
Orchestrator
REST
State queries
Request
Result
Replica network
Replica node
DAML execution engine
TLS
SBFT for global ordering & execution of requests to update global state
S1
S2
S3
S4
Cryptographically verifiable database
Client node architecture
Client node
DAML ledger API server
BFT client
Privacy filtered subset of state
Thin replica client
PostgreSQL
To/From replica network (Primary replica)
To/From replica nodes and other client nodes

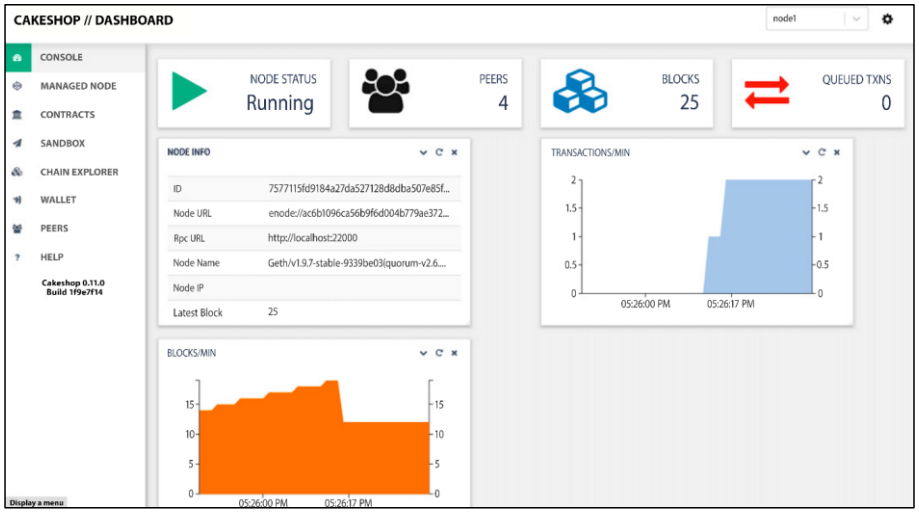

Quorum node
DApps
JSON RPC
Enhanced RPC API
Modified block generation
Modified block validation
Private transactions
EVM
Mempool
Pluggable consensus
RAFT
IBFT
POA
Private
Enhanced state
Public
P2P
IPC or HTTPS
Privacy manager
Transaction manager
Enclave
storage

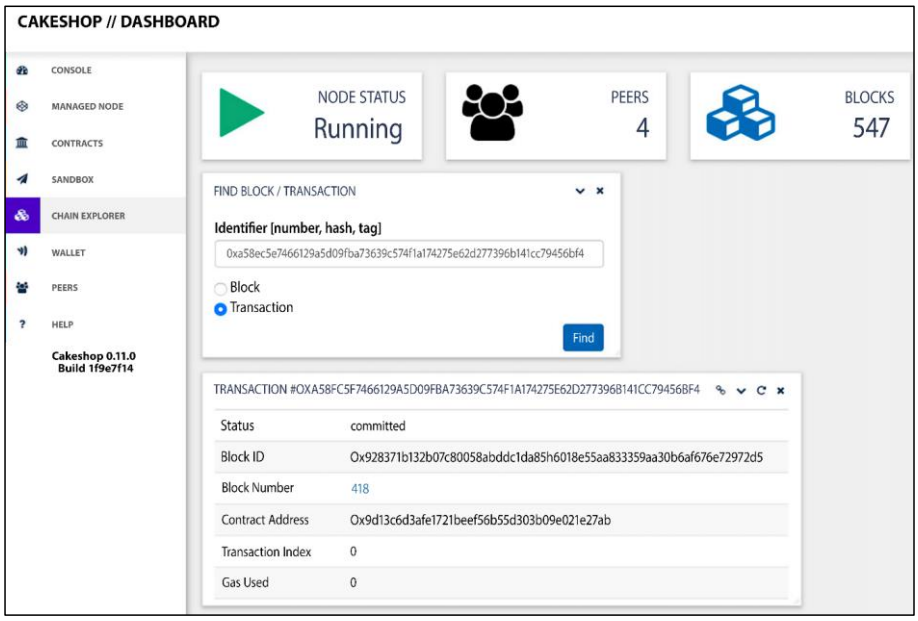

Party A
Party B
Party C
DApp
Quorum node A
Quorum node B
Quorum node C
Transaction manager A
Transaction manager B
Transaction manager C
Enclave A
Enclave B
Enclave C
Privacy manager
(1) Private transaction
(2) Transaction payload
(3) Encryption request
(4) Encrypted payload / data
(5) Store and forward
(6) Transaction hash
(7) Transaction propagation
(8) Block propagation
(9) Transaction existence check
(10) Decryption request
(11) Decrypted payload / data
(12) Transaction response


Transaction manager
Transaction manager
5. Store and send
Generates
SHA3-512 hash of the transaction payload
Stores
SHA3-512 hash
Encrypted payload
Encrypted SK for Tx
Nonces
Sender's Public key
Receiver's Public key
Sends
Encryption / Decryption request of transaction payload
Returns
Encrypted transaction payload
Encrypted symmetric key for transaction
Two random nonces
Public keys of the sender and receivers
OR
Decrypted transaction payload
Acknowledgement
Push to the other transaction managers
Enclave
Enclave
Generates
4. Enclave encryption process
Symmetric key for transaction
Two random nonces
Encrypts
Transaction payload
First random nonce
Curve25519
Elliptic curve Diffie-Hellman (ECDH)
XSalsa20
Stream cipher
Poly1305
Message Authentication Code
crypto_box
Generates
Shared symmetric key
Encrypts
Symmetric key for transaction
Second random nonce
11. Enclave decryption process
Shared symmetric key
Decrypts
Symmetric key for transaction
Stored nonce
Decrypts
Transaction payload

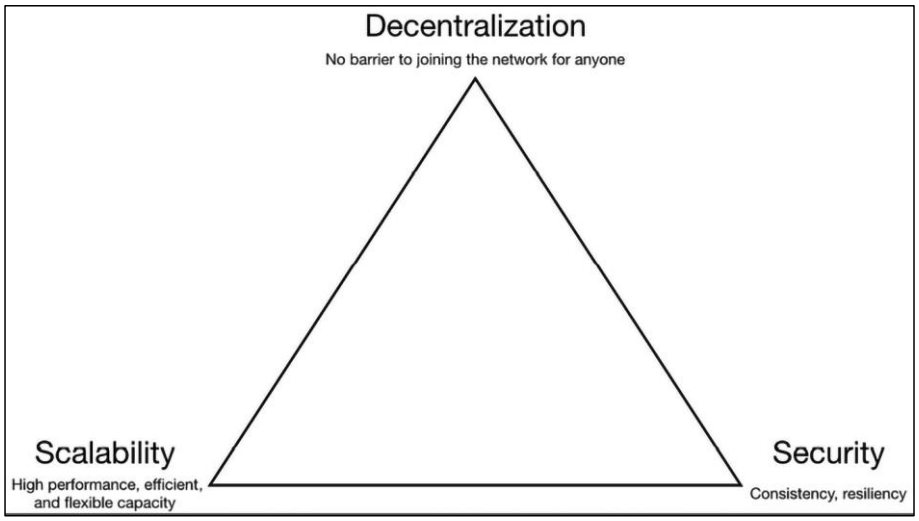


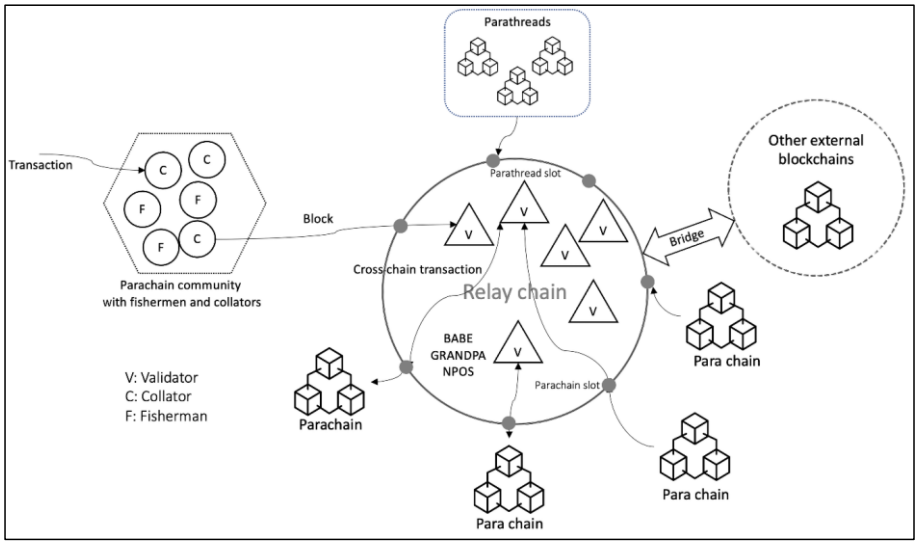

```
Welcome to Quorum Wizard!

This tool allows you to easily create bash, docker, and kubernetes files to start up a quorum network.
You can control consensus, privacy, network details and more for a customized setup.
Additionally you can choose to deploy our chain explorer, Cakeshop, to easily view and monitor your network.

We have 3 options to help you start exploring Quorum:

  1.  Quickstart - our 1 click option to create a 3 node raft network with tessera and cakeshop

  2.  Simple Network - using pregenerated keys from quorum 7nodes example,
      this option allows you to choose the number of nodes (7 max), consensus mechanism, transaction manager, and the option to deploy cakeshop

  3.  Custom Network - In addition to the options available in #2, this selection allows for further customization of your network.
      Choose to generate keys, customize ports for both bash and docker, or change the network id

Quorum Wizard will generate your startup files and everything required to bring up your network.
All you need to do is go to the specified location and run ./start.sh

 (Use arrow Keys)
> Quickstart (3-node raft network with tessera and cakeshop)
  Simple Network
  Custom Network
  Exit
```

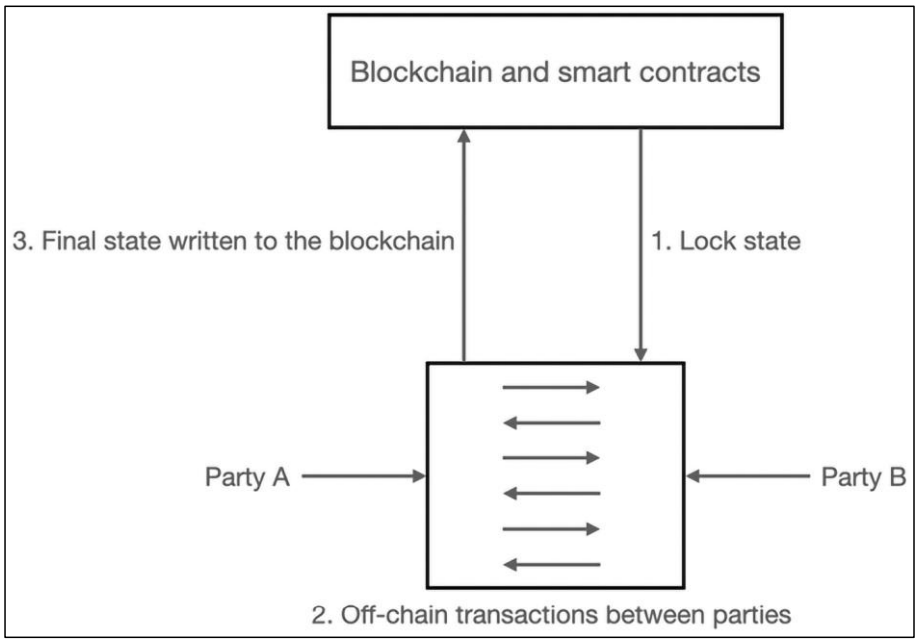


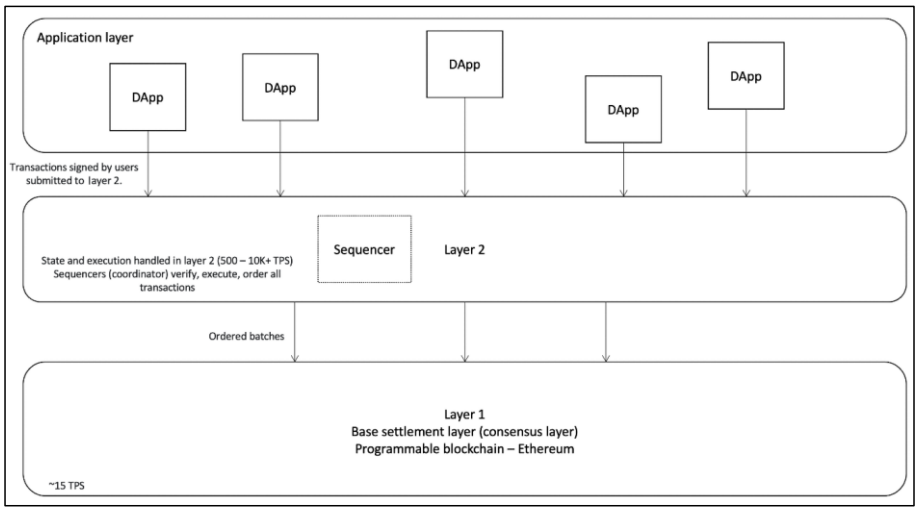

CAKESHOP // DASHBOARD
CONSOLE
MANAGED NODE
CONTRACTS
SANDBOX
CHAIN EXPLORER
WALLET
PEERS
HELP
Cakeshop 0.11.0
Build 1f9e7f14
NODE STATUS
Running
PEERS
4
BLOCKS
547
FIND BLOCK / TRANSACTION
Identifier [number, hash, tag]
0xa58ec5e7466129a5d09fba73639c574f1a174275e62d277396b141cc79456bf4
Block
Transaction
Find
TRANSACTION #0XA58FC5F7466129A5D09FBA73639C574F1A174275E62D277396B141CC79456BF4
Status
committed
Block ID
0x928371b132b07c80058abddc1da85h6018e55aa833359aa30b6af676e72972d5
Block Number
418
Contract Address
0x9d13c6d3afe1721beef56b55d303b09e021e27ab
Transaction Index
0
Gas Used
0

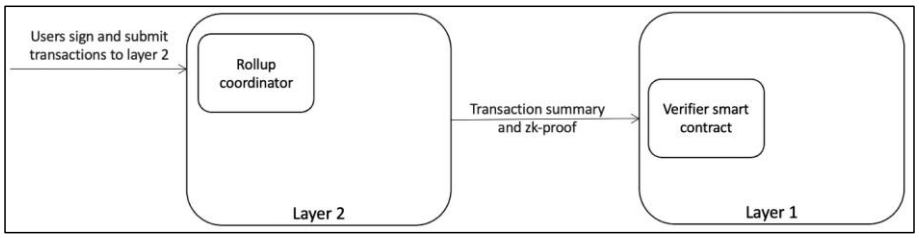
Chapter 17: Scalability

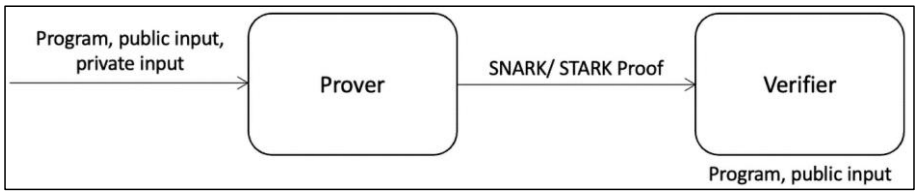


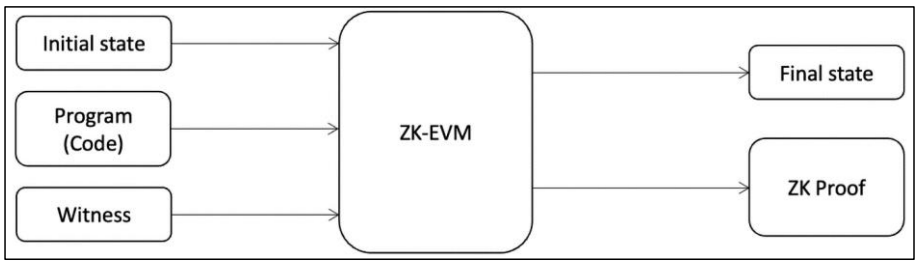


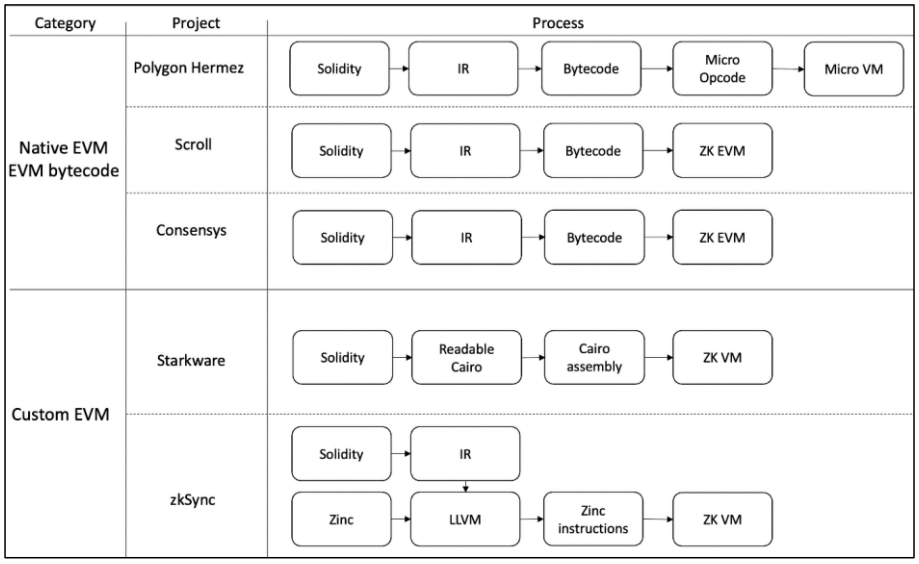

Blockchain and smart contracts
3. Final state written to the blockchain
1. Lock state
Party A
Party B
2. Off-chain transactions between parties

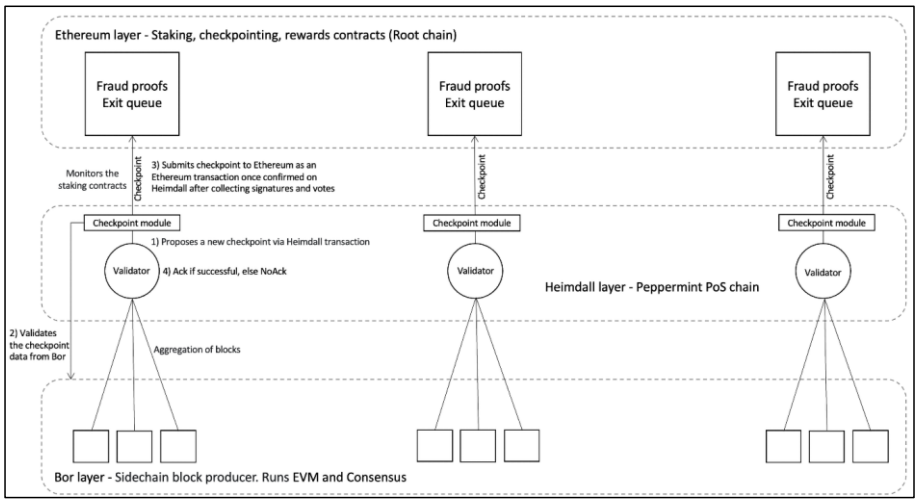

Application layer
DApp
DApp
DApp
DApp
DApp
Transactions signed by users submitted to layer 2.
State and execution handled in layer 2 (500 – 10K+ TPS)
Sequencers (coordinator) verify, execute, order all transactions
Sequencer
Layer 2
Ordered batches
Layer 1
Base settlement layer (consensus layer)
Programmable blockchain – Ethereum
~15 TPS

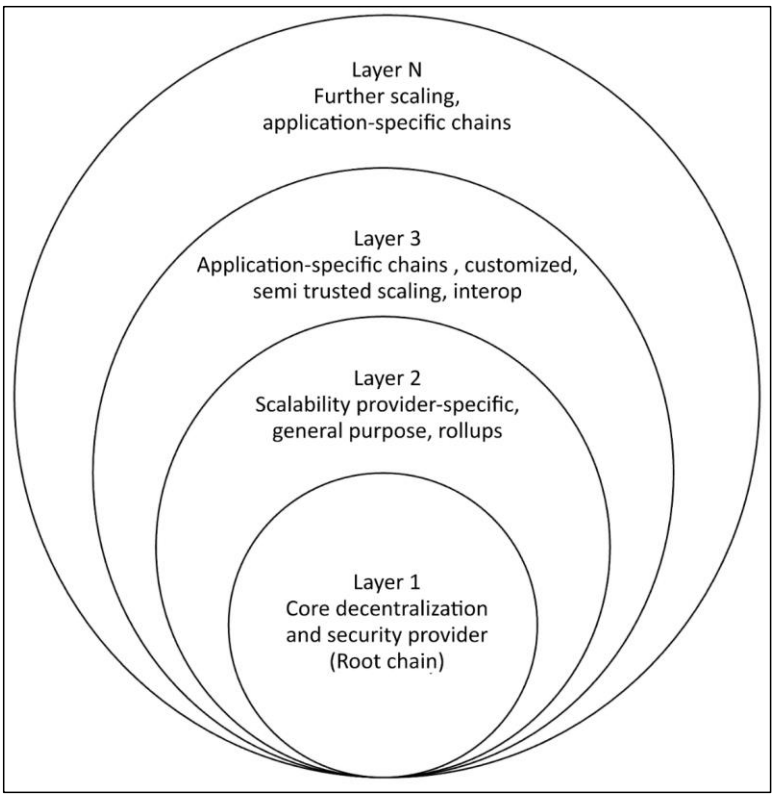

Users sign and submit transactions to layer 2
Rollup coordinator
Layer 2
Transaction summary and zk-proof
Verifier smart contract
Layer 1


Program, public input, private input
Prover
SNARK/ STARK Proof
Verifier
Program, public input

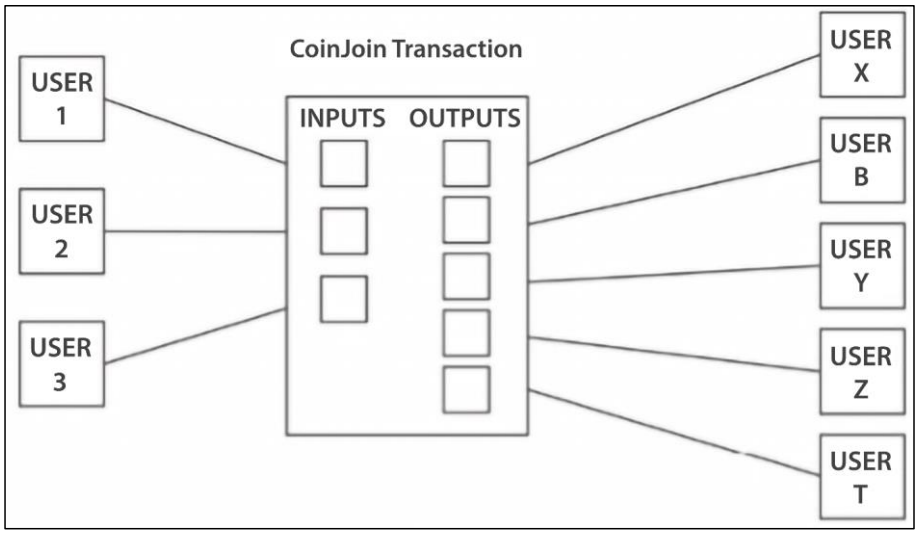

Initial state
Program (Code)
Witness
ZK-EVM
Final state
ZK Proof

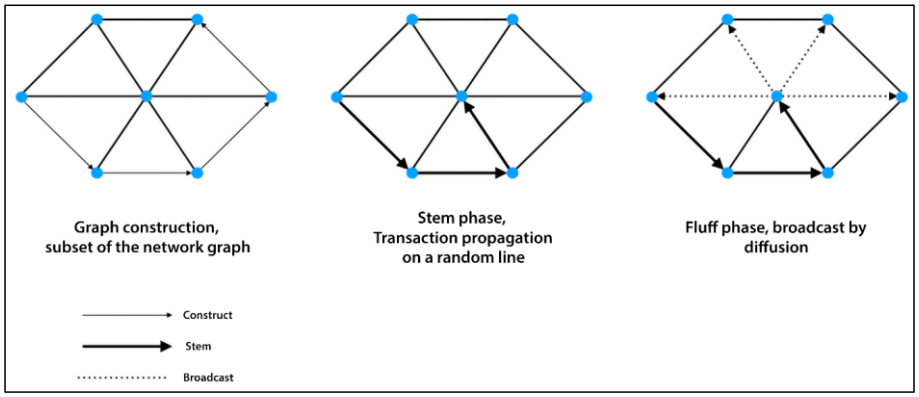

Category
Project
Process
Native EVM EVM bytecode
Polygon Hermez
Solidity
IR
Bytecode
Micro Opcode
Micro VM
Scroll
Solidity
IR
Bytecode
ZK EVM
Consensys
Solidity
IR
Bytecode
ZK EVM
Custom EVM
Starkware
Solidity
Readable Cairo
Cairo assembly
ZK VM
zkSync
Solidity
IR
Zinc
LLVM
Zinc instructions
ZK VM

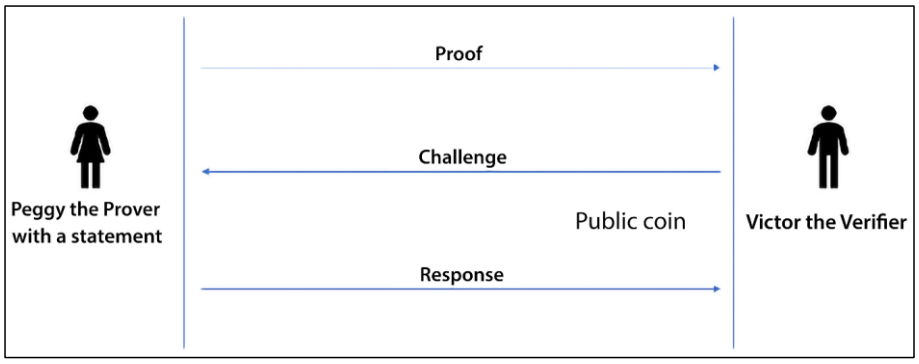

Ethereum layer - Staking, checkpointing, rewards contracts (Root chain)
Fraud proofs Exit queue
Fraud proofs Exit queue
Fraud proofs Exit queue
Monitors the staking contracts
Checkpoint
3) Submits checkpoint to Ethereum as an Ethereum transaction once confirmed on Heimdall after collecting signatures and votes
Checkpoint
Checkpoint
Checkpoint module
Checkpoint module
Checkpoint module
1) Proposes a new checkpoint via Heimdall transaction
Validator
4) Ack if successful, else NoAck
Validator
Validator
Heimdall layer - Peppermint PoS chain
2) Validates the checkpoint data from Bor
Aggregation of blocks
Bor layer - Sidechain block producer. Runs EVM and Consensus

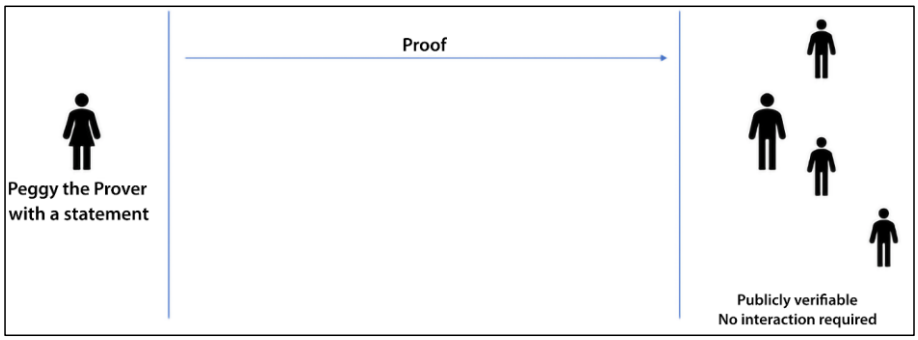

Layer N
Further scaling,
application-specific chains
Layer 3
Application-specific chains , customized,
semi trusted scaling, interop
Layer 2
Scalability provider-specific,
general purpose, rollups
Layer 1
Core decentralization
and security provider
(Root chain)

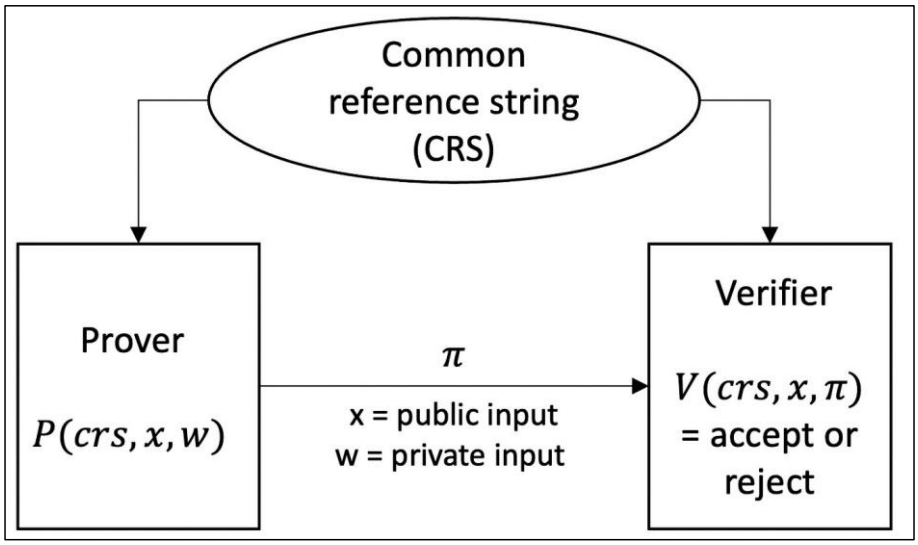
Chapter 18: Blockchain Privacy

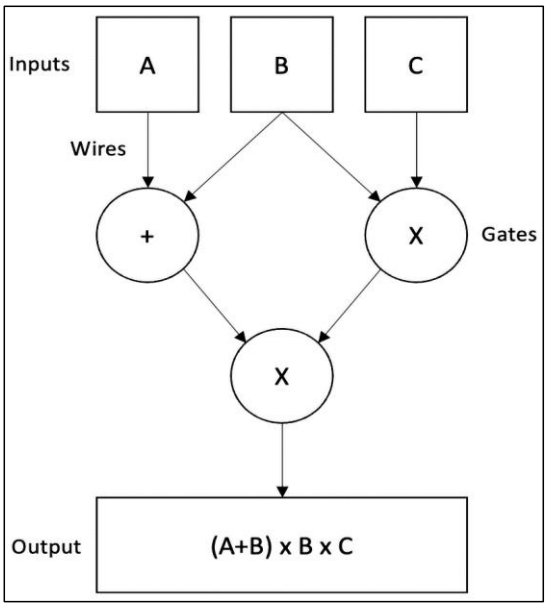


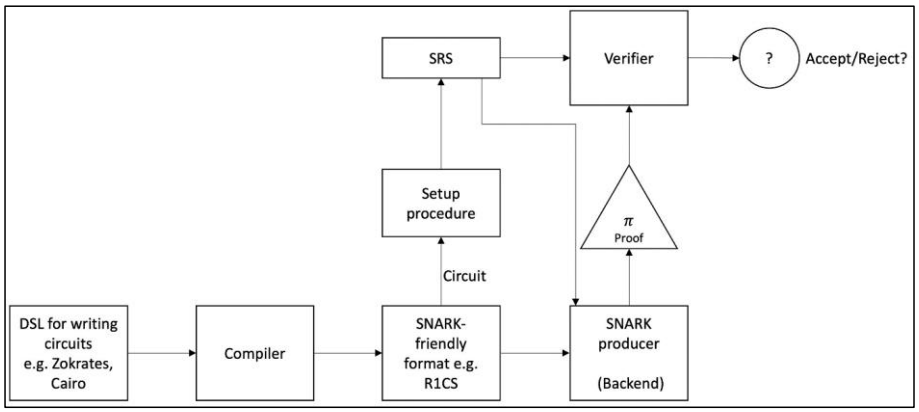


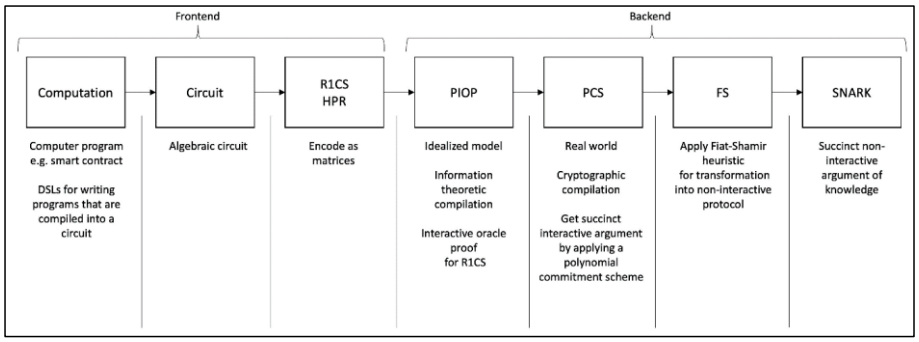


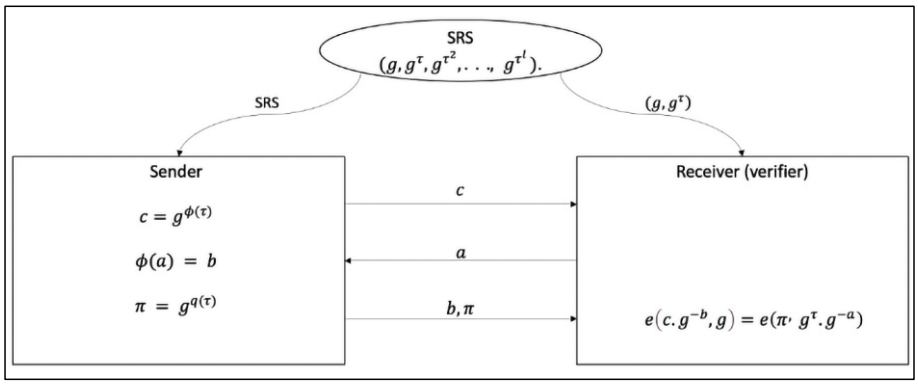

Proof
Peggy the Prover
with a statement
Publicly verifiable
No interaction required

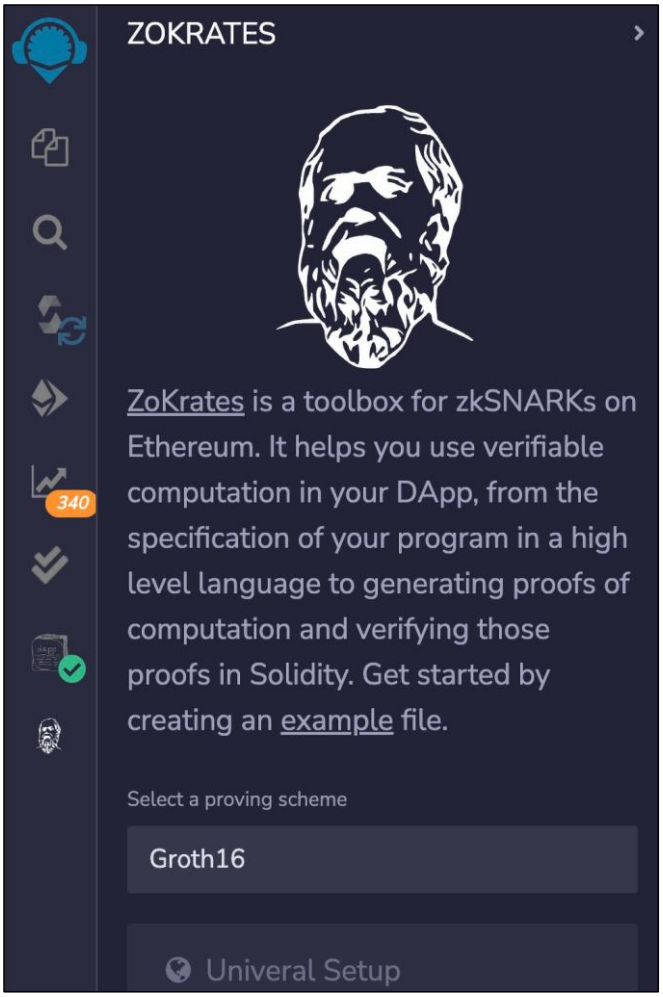

Common
reference string
(CRS)
Prover
$P(crs, x, w)$
π
x = public input
w = private input
Verifier
$V(crs, x, \pi)$
= accept or
reject


Inputs
A
B
C
Wires
+
X
Gates
X
Output
(A+B) x B x C

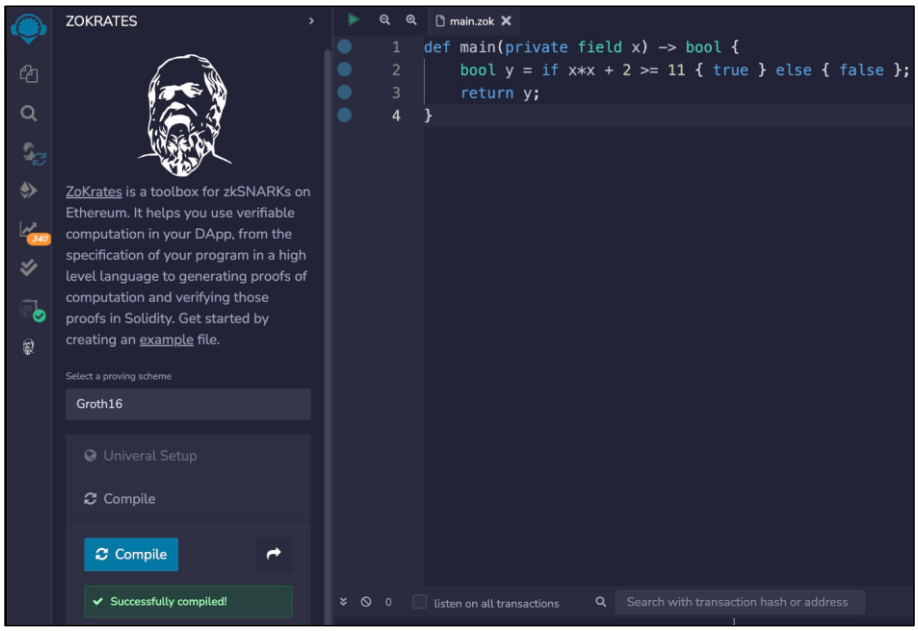

SRS
Verifier
?
Accept/Reject?
Setup procedure
π
Proof
Circuit
DSL for writing circuits e.g. Zokrates, Cairo
Compiler
SNARK-friendly format e.g. R1CS
SNARK producer
(Backend)

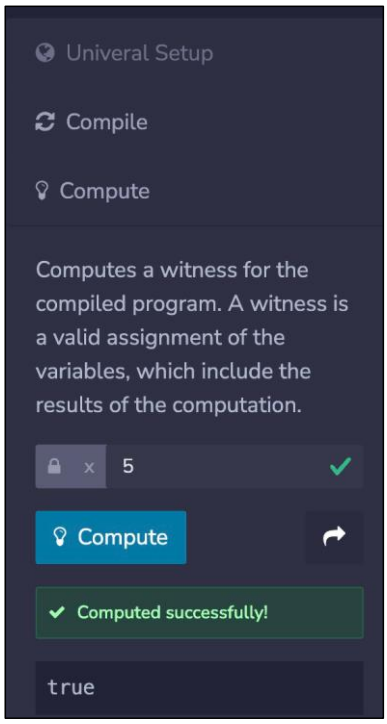

Frontend
Backend
Computation
Circuit
R1CS HPR
PIOP
PCS
FS
SNARK
Computer program e.g. smart contract
DSLs for writing programs that are compiled into a circuit
Algebraic circuit
Encode as matrices
Idealized model
Information theoretic compilation
Interactive oracle proof for R1CS
Real world
Cryptographic compilation
Get succinct interactive argument by applying a polynomial commitment scheme
Apply Fiat-Shamir heuristic for transformation into non-interactive protocol
Succinct non-interactive argument of knowledge

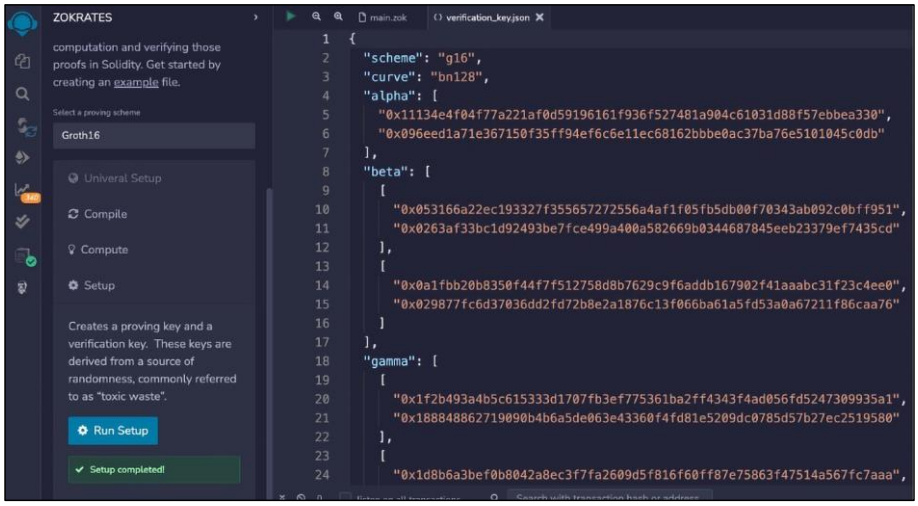

SRS
$(g, g^{\tau}, g^{\tau^2}, \ldots, g^{\tau^l})$.
SRS
(g, g^{τ})
Sender
$c = g^{\phi(\tau)}$
$\phi(a) = b$
$\pi = g^{q(\tau)}$
c
a
b, π
Receiver (verifier)
$e(c.g^{-b}, g) = e(\pi, g^{\tau}.g^{-a})$

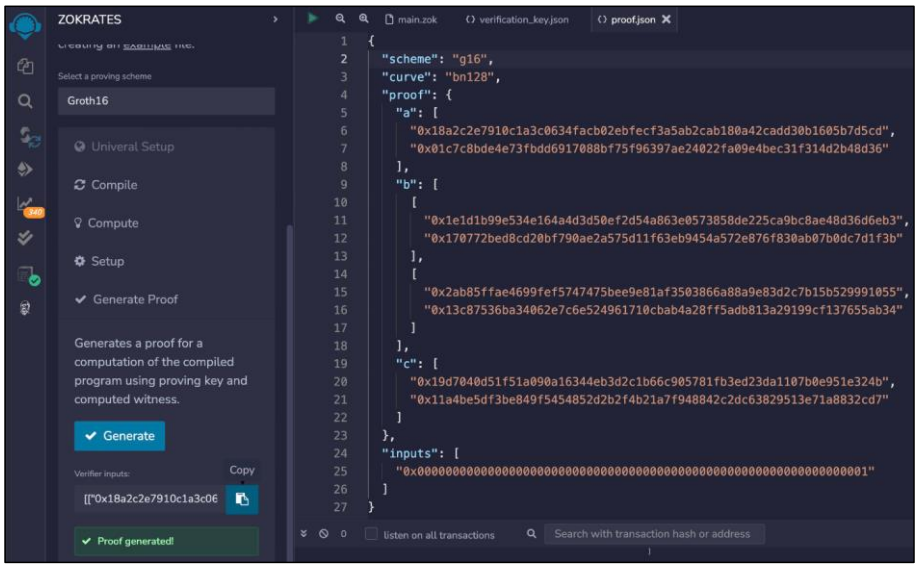

ZOKRATES
340
ZoKrates is a toolbox for zkSNARKs on Ethereum. It helps you use verifiable computation in your DApp, from the specification of your program in a high level language to generating proofs of computation and verifying those proofs in Solidity. Get started by creating an example file.
Select a proving scheme
Groth16
Univeral Setup

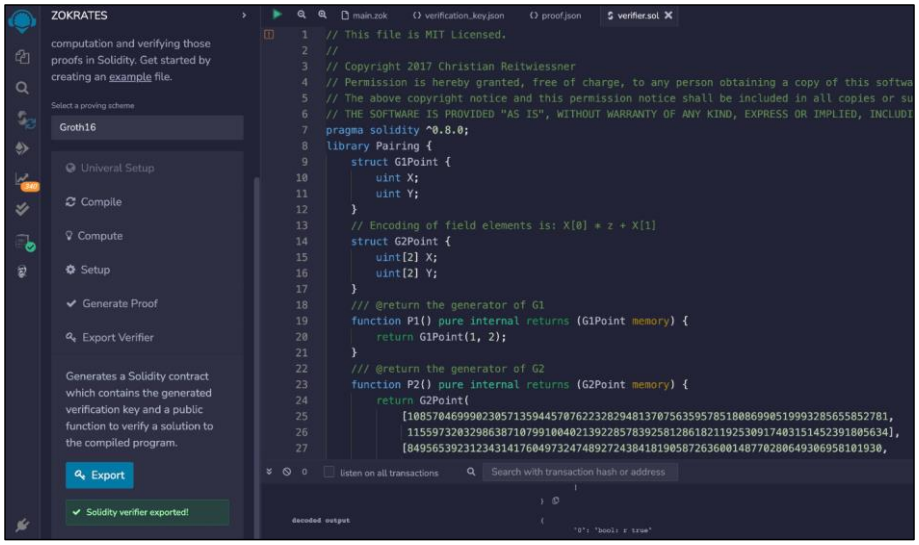

FILE EXPLORER
WORKSPACES
default_workspace
contracts
scripts
tests
.deps
artifacts
README.txt
main.zok
main.zok
1 def main(private field x) -> bool {
2 bool y = if x**x + 2 >= 11 { true } else { false };
3 return y;
4 }

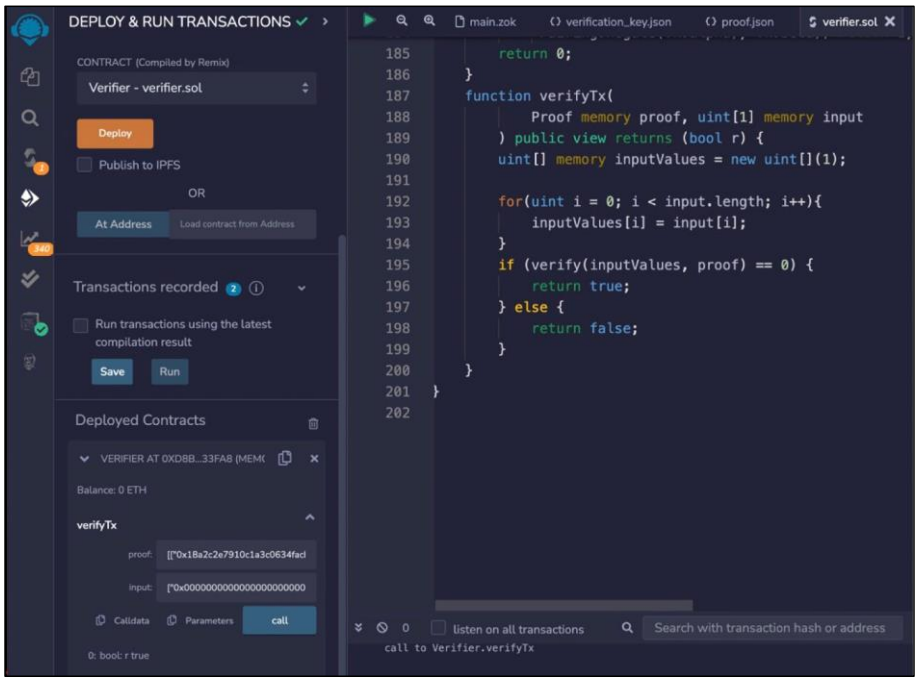

ZOKRATES
ZoKrates is a toolbox for zkSNARKs on Ethereum. It helps you use verifiable computation in your DApp, from the specification of your program in a high level language to generating proofs of computation and verifying those proofs in Solidity. Get started by creating an example file.
Select a proving scheme
Groth16
Univeral Setup
Compile
Compile
Successfully compiled!
main.zok
1 def main(private field x) -> bool {
2 bool y = if x**x + 2 >= 11 { true } else { false };
3 return y;
4 }
0
listen on all transactions
Search with transaction hash or address


Univeral Setup
Compile
Compute
Computes a witness for the compiled program. A witness is a valid assignment of the variables, which include the results of the computation.
x
5
Compute
Computed successfully!
true

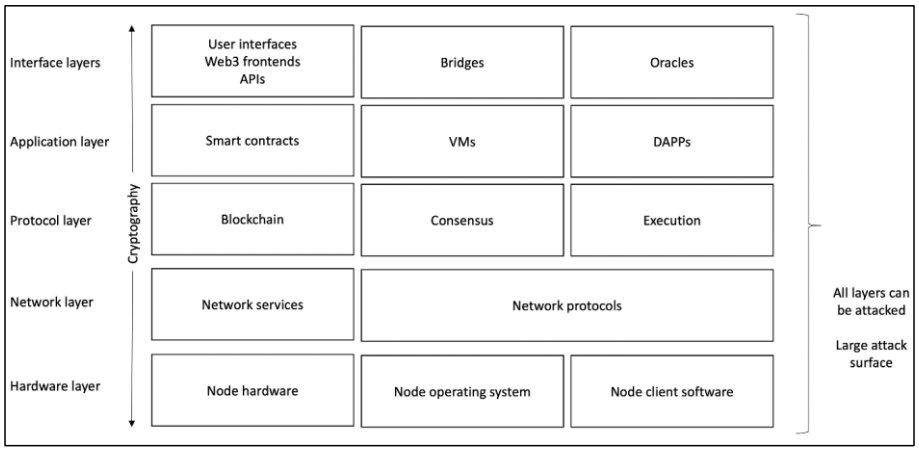

ZOKRATES
computation and verifying those proofs in Solidity. Get started by creating an example file.
Select a proving scheme
Groth16
Univeral Setup
Compile
Compute
Setup
Creates a proving key and a verification key. These keys are derived from a source of randomness, commonly referred to as "toxic waste".
Run Setup
Setup completed!
main.zok
verification_key.json
{
"scheme": "g16",
"curve": "bn128",
"alpha": [
"0x11134e4f04f77a221af0d59196161f936f527481a904c61031d88f57ebbea330",
"0x096eed1a71e367150f35ff94ef6c6e11ec68162bbbe0ac37ba76e5101045c0db"
],
"beta": [
[
"0x053166a22ec193327f355657272556a4af1f05fb5db00f70343ab092c0bff951",
"0x0263af33bc1d92493be7fce499a400a582669b0344687845eeb23379ef7435cd"
],
[
"0x0a1fbb20b8350f44f7f512758d8b7629c9f6addb167902f41aaabc31f23c4ee0",
"0x029877fc6d37036dd2fd72b8e2a1876c13f066ba61a5fd53a0a67211f86caa76"
]
],
"gamma": [
[
"0x1f2b493a4b5c615333d1707fb3ef775361ba2ff4343f4ad056fd5247309935a1",
"0x188848862719090b4b6a5de063e43360f4fd81e5209dc0785d57b27ec2519580"
],
[
"0x1d8b6a3bef0b8042a8ec3f7fa2609d5f816f60ff87e75863f47514a567fc7aaa",

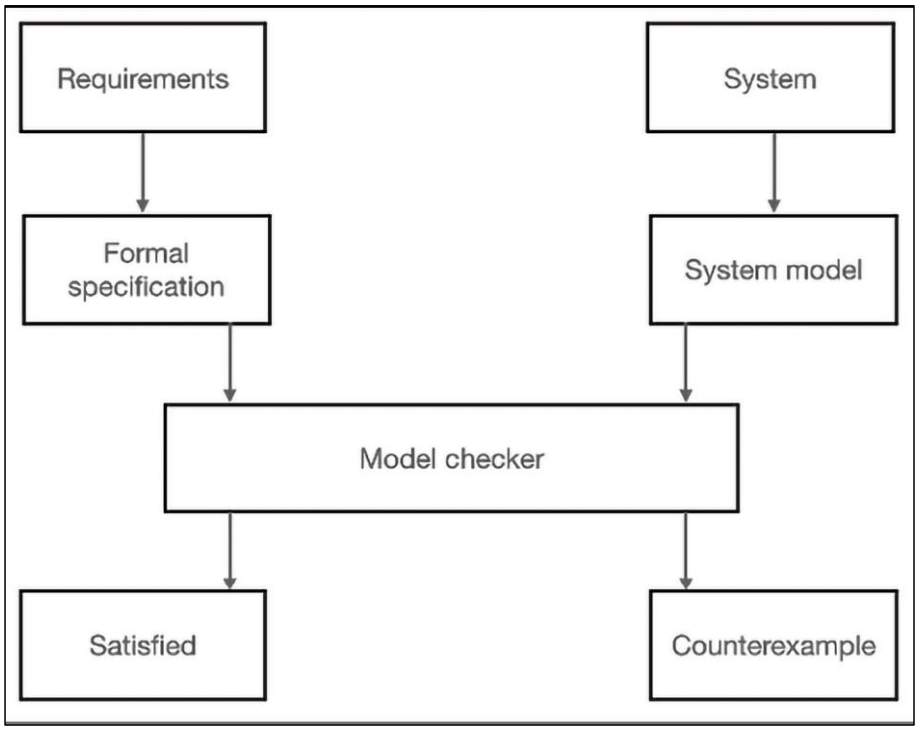

ZOKRATES
Select a proving scheme
Groth16
Univeral Setup
Compile
Compute
Setup
Generate Proof
Generates a proof for a computation of the compiled program using proving key and computed witness.
Generate
Verifier inputs:
Copy
[["0x18a2c2e7910c1a3c06
Proof generated!
main.zok
verification_key.json
proof.json
{
"scheme": "g16",
"curve": "bn128",
"proof": {
"a": [
"0x18a2c2e7910c1a3c0634facb02ebfecf3a5ab2cab180a42cadd30b1605b7d5cd",
"0x01c7c8bde4e73fbdd6917088bf75f96397ae24022fa09e4bec31f314d2b48d36"
],
"b": [
[
"0x1e1d1b99e534e164a4d3d50ef2d54a863e0573858de225ca9bc8ae48d36d6eb3",
"0x170772bed8cd20bf790ae2a575d11f63eb9454a572e876f830ab07b0dc7d1f3b"
],
[
"0x2ab85ffae4699fef5747475bee9e81af3503866a88a9e83d2c7b15b529991055",
"0x13c87536ba34062e7c6e524961710cbab4a28ff5adb813a29199cf137655ab34"
]
],
"c": [
"0x19d7040d51f51a090a16344eb3d2c1b66c905781fb3ed23da1107b0e951e324b",
"0x11a4be5df3be849f5454852d2b2f4b21a7f948842c2dc63829513e71a8832cd7"
]
},
"inputs": [
"0x0000000000000000000000000000000000000000000000000000000000000001"
]
}
listen on all transactions
Search with transaction hash or address


ZOKRATES
computation and verifying those proofs in Solidity. Get started by creating an example file.
Select a proving scheme
Groth16
Univeral Setup
Compile
Compute
Setup
Generate Proof
Export Verifier
Generates a Solidity contract which contains the generated verification key and a public function to verify a solution to the compiled program.
Export
Solidity verifier exported!
main.zok
verification_key.json
proof.json
verifier.sol
// This file is MIT Licensed.
//
// Copyright 2017 Christian Reitwiessner
// Permission is hereby granted, free of charge, to any person obtaining a copy of this softwa
// The above copyright notice and this permission notice shall be included in all copies or su
// THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDI
pragma solidity ^0.8.0;
library Pairing {
struct G1Point {
uint X;
uint Y;
}
// Encoding of field elements is: X[0] * z + X[1]
struct G2Point {
uint[2] X;
uint[2] Y;
}
/// @return the generator of G1
function P1() pure internal returns (G1Point memory) {
return G1Point(1, 2);
}
/// @return the generator of G2
function P2() pure internal returns (G2Point memory) {
return G2Point(
[10857046999023057135944570762232829481370756359578518086990519993285655852781,
11559732032986387107991004021392285783925812861821192530917403151452391805634],
[8495653923123431417604973247489272438418190587263600148770280649306958101930,
listen on all transactions
Search with transaction hash or address
decoded output
'0': 'bool: r true'


DEPLOY & RUN TRANSACTIONS
CONTRACT (Compiled by Remix)
Verifier - verifier.sol
Deploy
Publish to IPFS
OR
At Address
Load contract from Address
Transactions recorded
Run transactions using the latest compilation result
Save
Run
Deployed Contracts
VERIFIER AT 0XD8B...33FA8 (MEM
Balance: 0 ETH
verifyTx
proof:
input:
Calldata
Parameters
call
0: bool: r true
main.zok
verification_key.json
proof.json
verifier.sol
return 0;
}
function verifyTx(
Proof memory proof, uint[1] memory input
) public view returns (bool r) {
uint[] memory inputValues = new uint[](1);
for(uint i = 0; i < input.length; i++){
inputValues[i] = input[i];
}
if (verify(inputValues, proof) == 0) {
return true;
} else {
return false;
}
}
}
listen on all transactions
Search with transaction hash or address
call to Verifier.verifyTx

Chapter 19: Blockchain Security

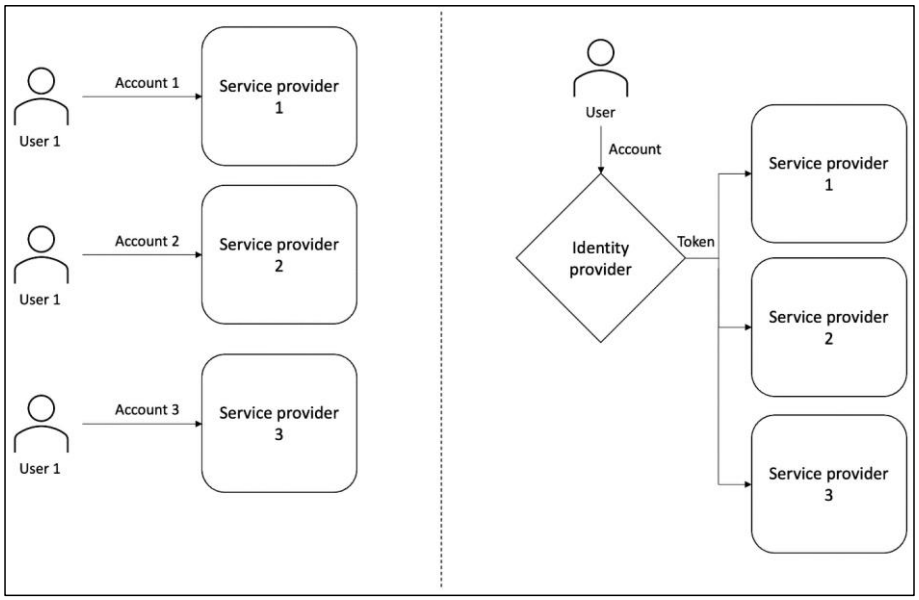


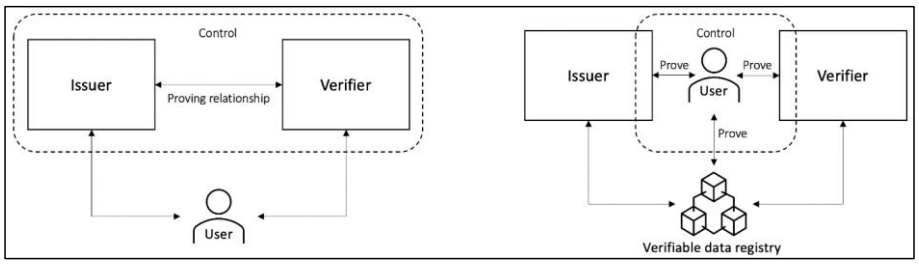


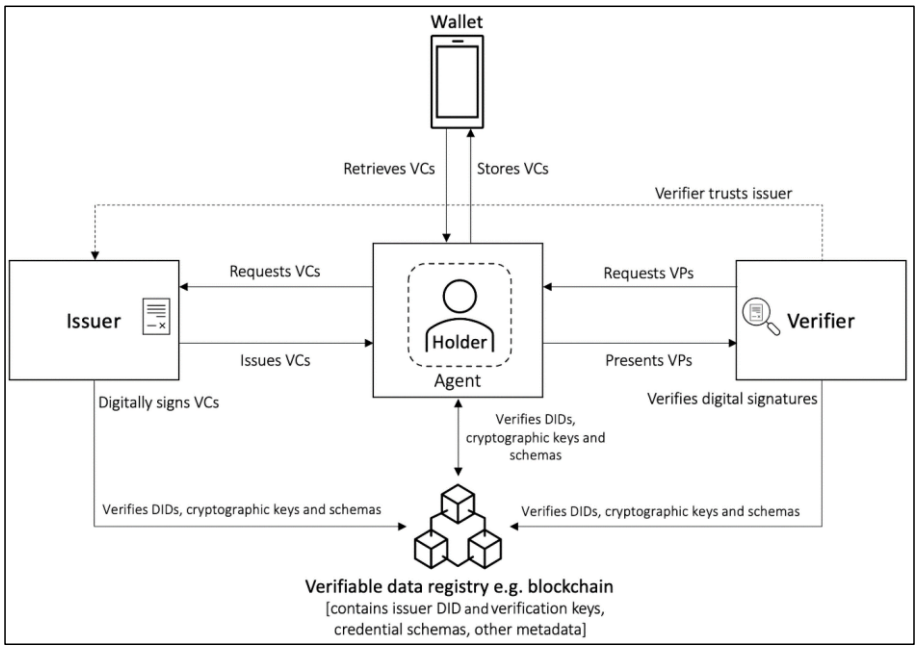
```
1  pragma solidity ^0.4.0;
2  contract Fund {
3      mapping(address => uint) shares;
4      function withdraw() public {
5          if (msg.sender.call.value(shares[msg.sender])())
6          shares[msg.sender] = 0;
7
8      }
9  }
```

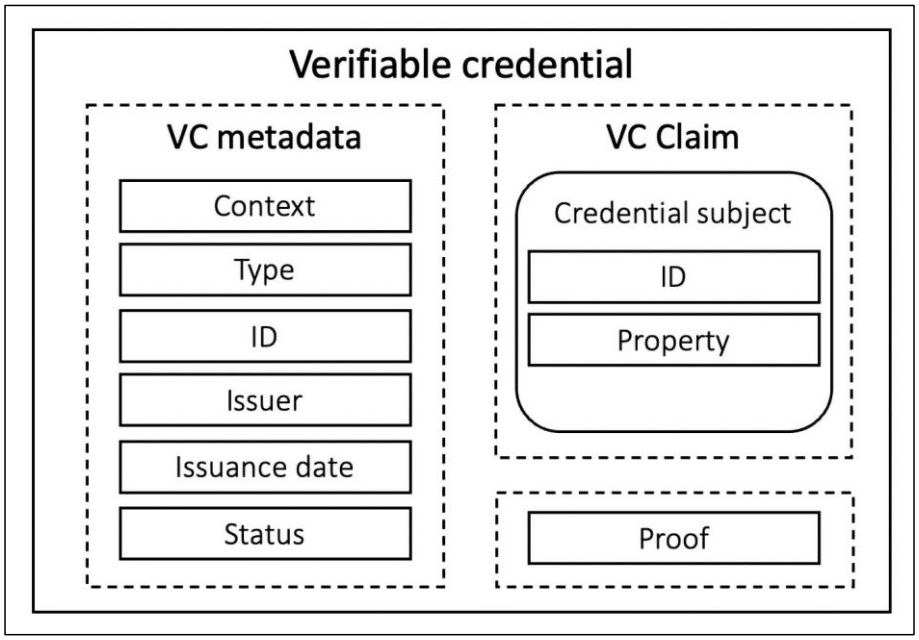
```
root@fa9ef6ac8455: /home/oyente/oyente
(venv)root@fa9ef6ac8455:/home/oyente/oyente# python oyente.py a1.sol
Contract Fund:
Running, please wait...
        ============ Results ===========
          CallStack Attack:      False
THIS IS A CALLLLLLLLL
{'path_condition': [Iv >= 0, init_Is >= Iv, init_Ia >= 0, If(Id_0/
   26959946667150639794667015087019630673637144422540572481103610249216 ==
   1020253707,
   1,
   0) !=
0, Not(Iv != 0)], 'Is': Is, 'Iv': Iv, 'some_var_1': some_var_1, 'Id_0': Id
_0, 'Ia_store_some_var_1': Ia_store_some_var_1, 'Ia': Ia}

 This is the global state
{'Ia': {'some_var_1': 0}, 'miu_i': 3L, 'balance': {'Ia': init_Ia + Iv, 'Is
': init_Is - Iv}}
{64: 96, 0: Is & 1461501637330902918203684832716283019655932542975, 32: 0}

CALL params

Is & 1461501637330902918203684832716283019655932542975

Ia_store_some_var_1


 =>>>>>> New PC: []

Reentrancy_bug? True

Added True
          Concurrency Bug:       False
          Time Dependency:       False
          Reentrancy bug exists: True
        ====== Analysis Completed ======
(venv)root@fa9ef6ac8455:/home/oyente/oyente#
```

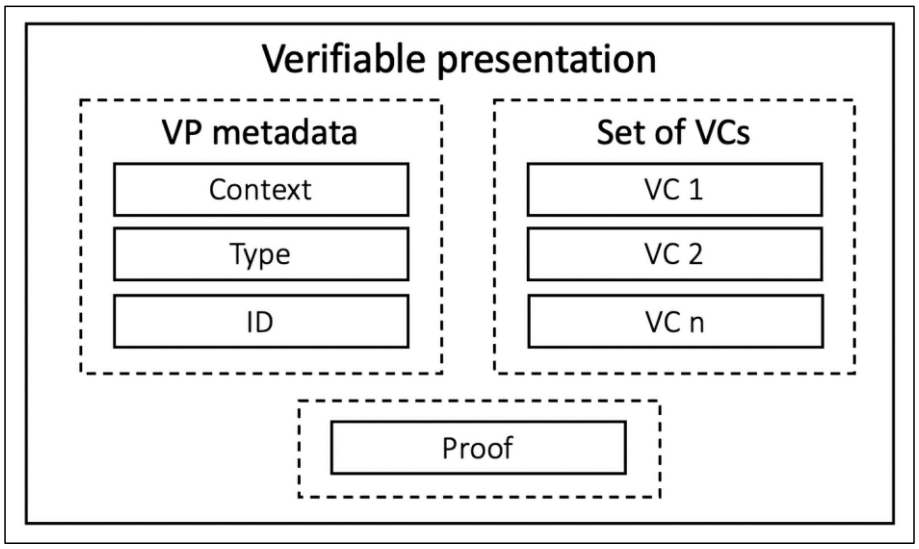
Chapter 20: Decentralized Identity

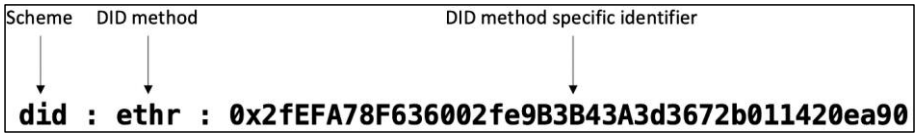


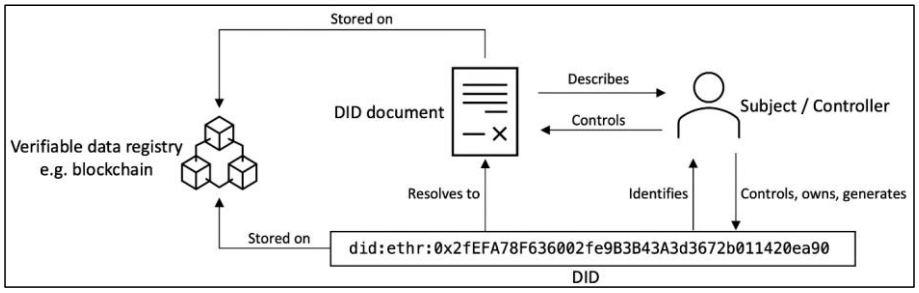


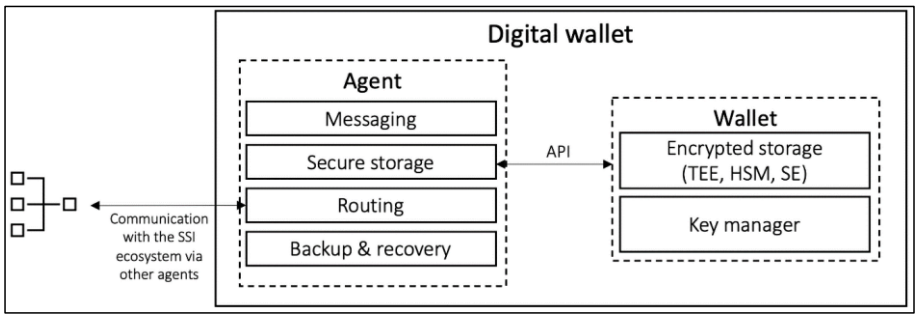

Wallet
Retrieves VCs
Stores VCs
Verifier trusts issuer
Issuer
Requests VCs
Issues VCs
Holder
Agent
Requests VPs
Presents VPs
Verifier
Digitally signs VCs
Verifies DIDs, cryptographic keys and schemas
Verifies digital signatures
Verifies DIDs, cryptographic keys and schemas
Verifies DIDs, cryptographic keys and schemas
Verifiable data registry e.g. blockchain
[contains issuer DID and verification keys, credential schemas, other metadata]

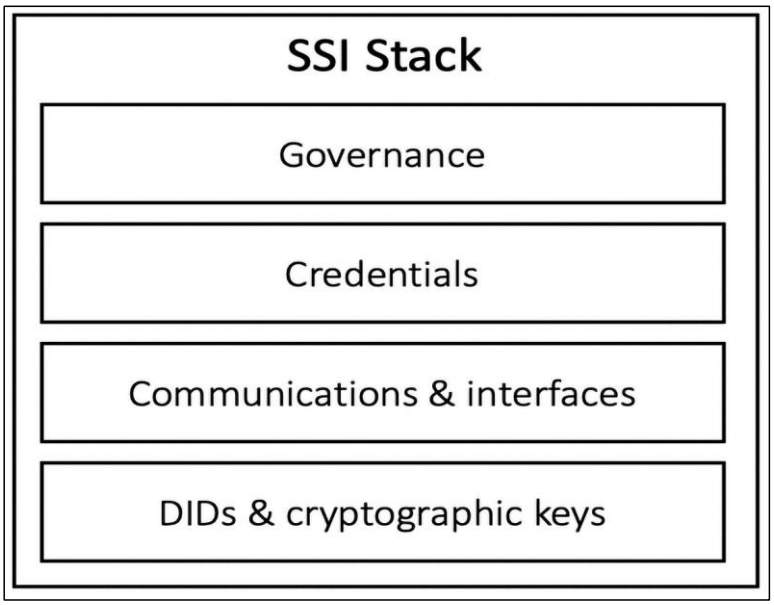

Verifiable credential
VC metadata
Context
Type
ID
Issuer
Issuance date
Status
VC Claim
Credential subject
ID
Property
Proof

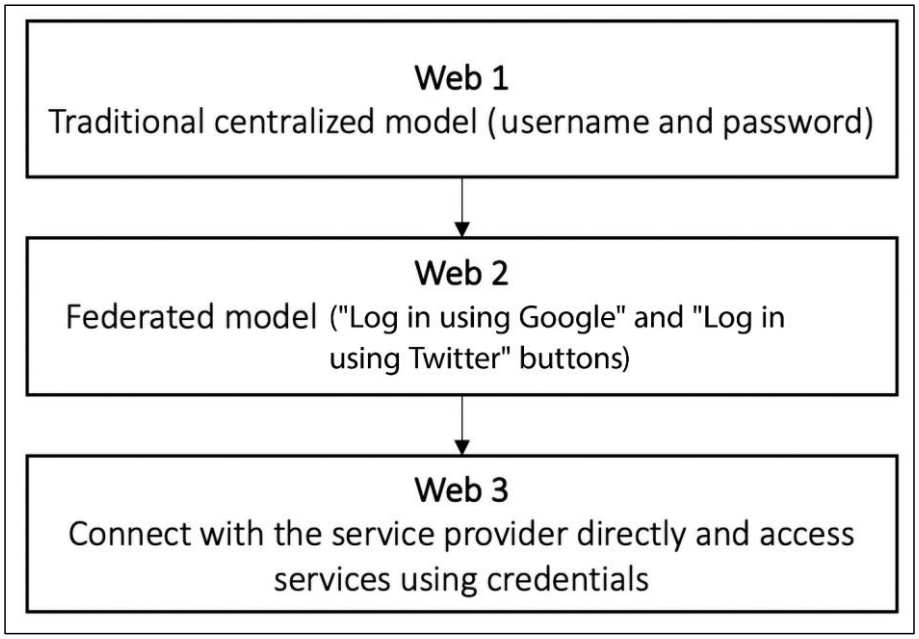

Verifiable presentation
VP metadata
Context
Type
ID
Set of VCs
VC 1
VC 2
VC n
Proof


Scheme
DID method
DID method specific identifier
did : ethr : 0x2fEFA78F636002fe9B3B43A3d3672b011420ea90

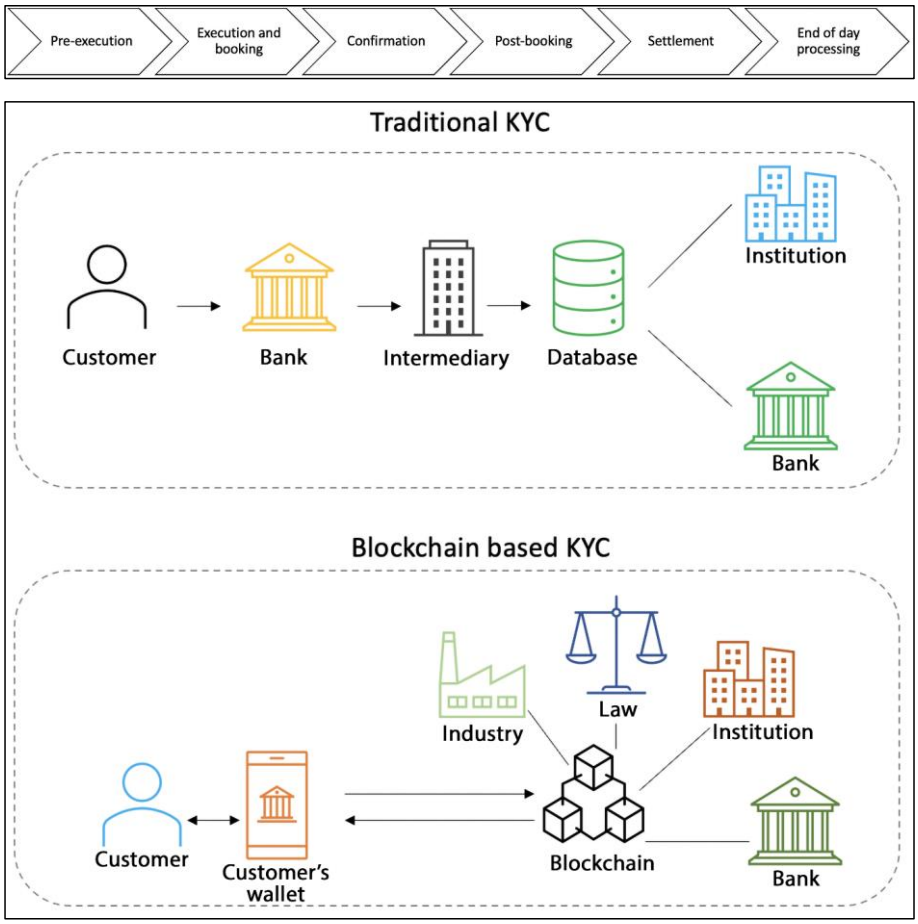

Stored on
DID document
Describes
Controls
Subject / Controller
Verifiable data registry
e.g. blockchain
Resolves to
Identifies
Controls, owns, generates
Stored on
did:ethr:0x2fEFA78F636002fe9B3B43A3d3672b011420ea90
DID

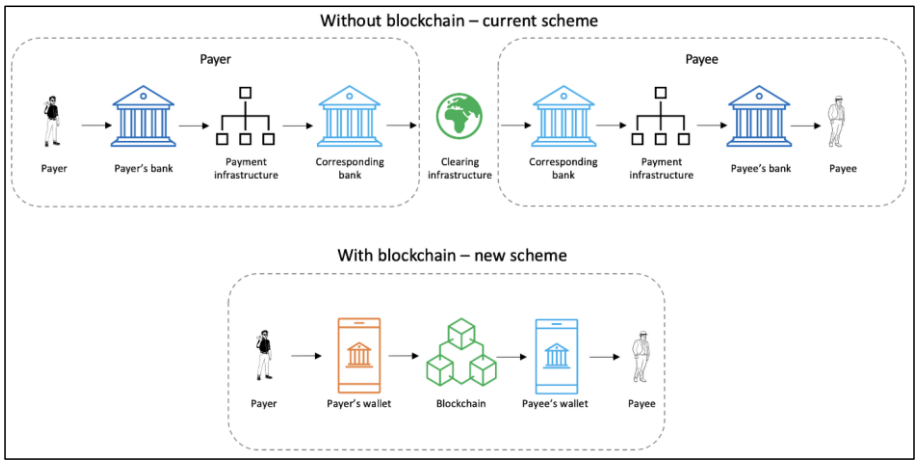

Digital wallet
Agent
Messaging
Secure storage
Routing
Backup & recovery
API
Wallet
Encrypted storage
(TEE, HSM, SE)
Key manager
Communication
with the SSI
ecosystem via
other agents

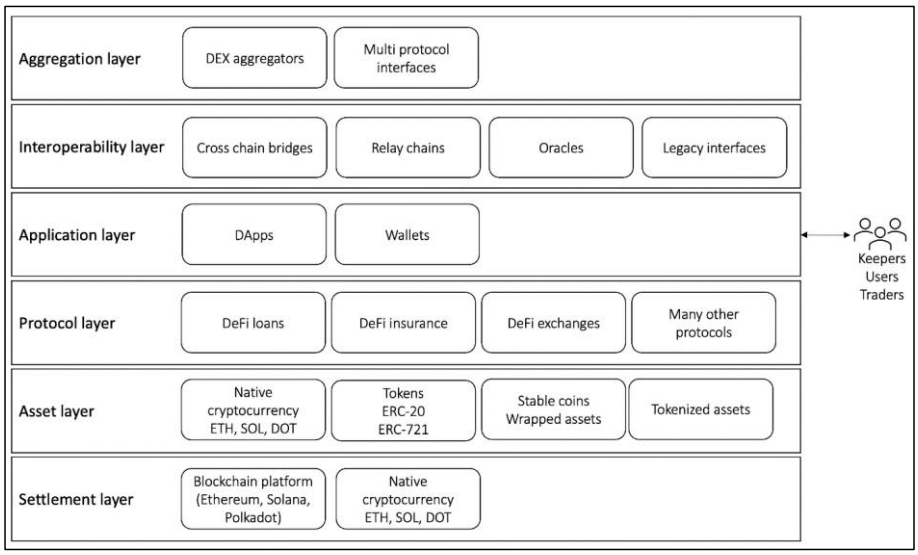

SSI Stack
Governance
Credentials
Communications & interfaces
DIDs & cryptographic keys

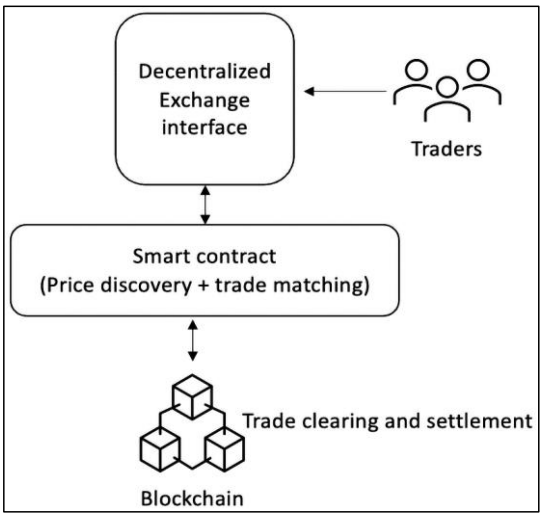

Web 1
Traditional centralized model (username and password)
Web 2
Federated model ("Log in using Google" and "Log in using Twitter" buttons)
Web 3
Connect with the service provider directly and access services using credentials

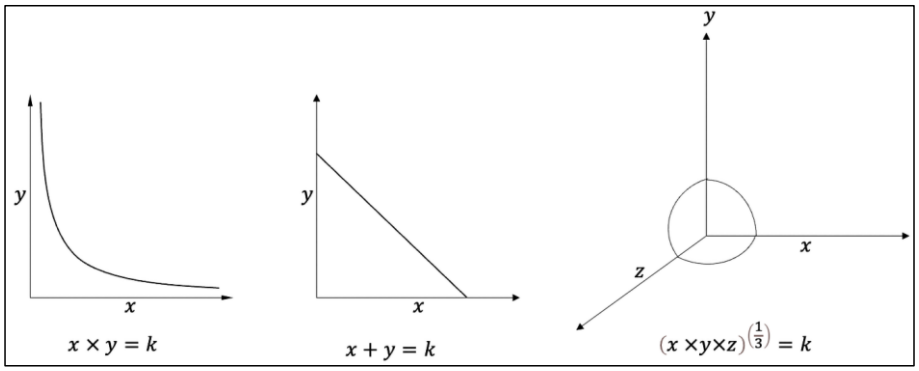
Chapter 21: Decentralized Finance

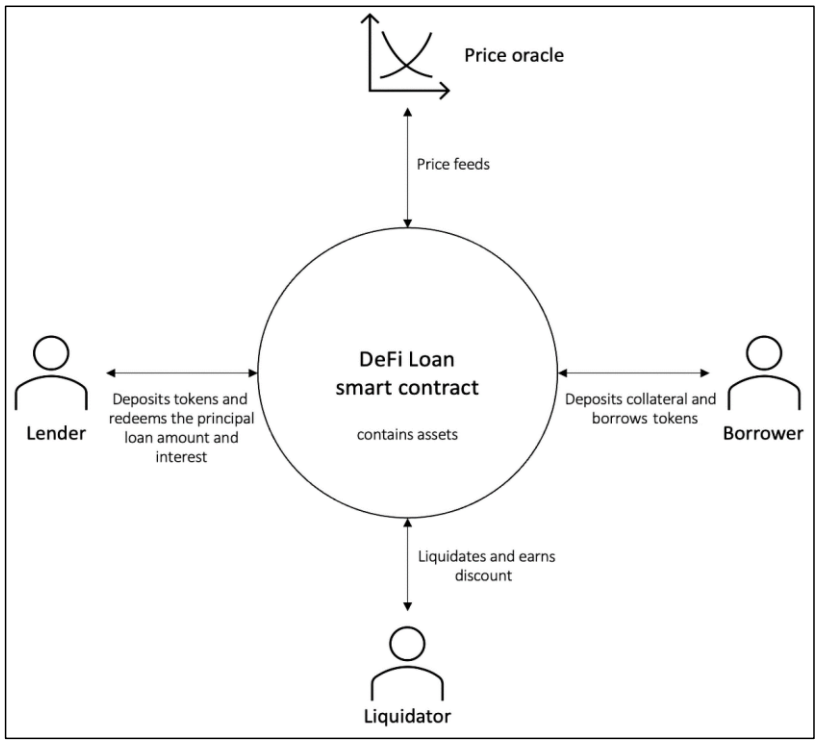


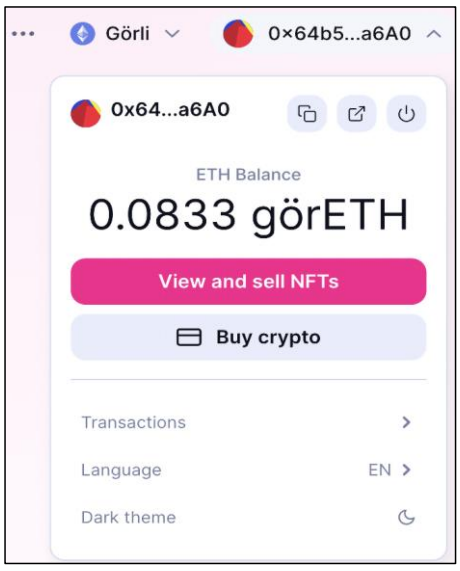

Without blockchain – current scheme
Payer
Payee
Payer
Payer's bank
Payment infrastructure
Corresponding bank
Clearing infrastructure
Corresponding bank
Payment infrastructure
Payee's bank
Payee
With blockchain – new scheme
Payer
Payer's wallet
Blockchain
Payee's wallet
Payee

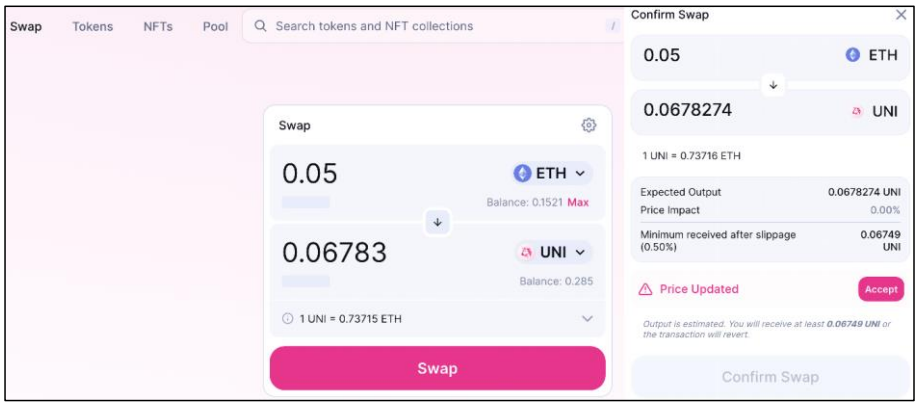

Aggregation layer
DEX aggregators
Multi protocol interfaces
Interoperability layer
Cross chain bridges
Relay chains
Oracles
Legacy interfaces
Application layer
DApps
Wallets
Protocol layer
DeFi loans
DeFi insurance
DeFi exchanges
Many other protocols
Asset layer
Native cryptocurrency ETH, SOL, DOT
Tokens ERC-20 ERC-721
Stable coins Wrapped assets
Tokenized assets
Settlement layer
Blockchain platform (Ethereum, Solana, Polkadot)
Native cryptocurrency ETH, SOL, DOT
Keepers
Users
Traders

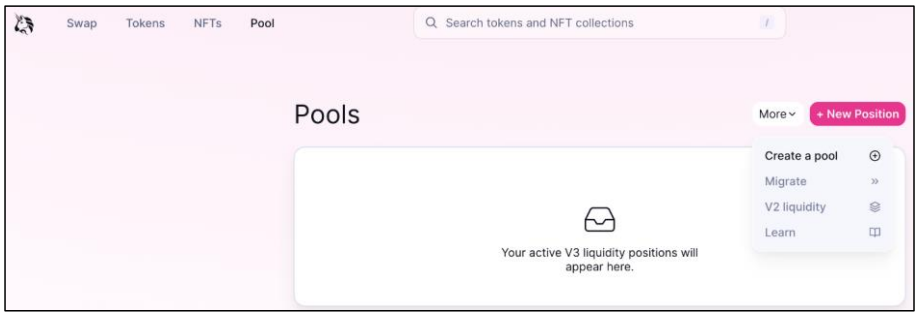

Decentralized
Exchange
interface
Traders
Smart contract
(Price discovery + trade matching)
Trade clearing and settlement
Blockchain

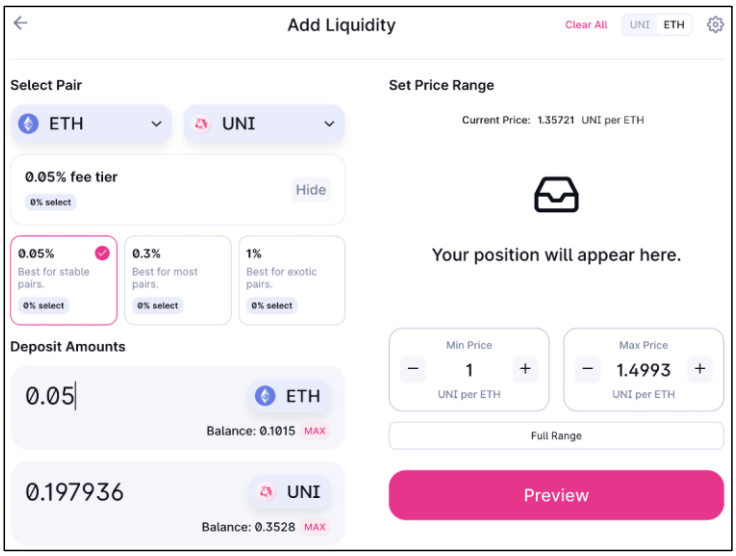

y
x
$x \times y = k$
y
x
$x + y = k$
y
x
z
$(x \times y \times z)^{(\frac{1}{3})} = k$

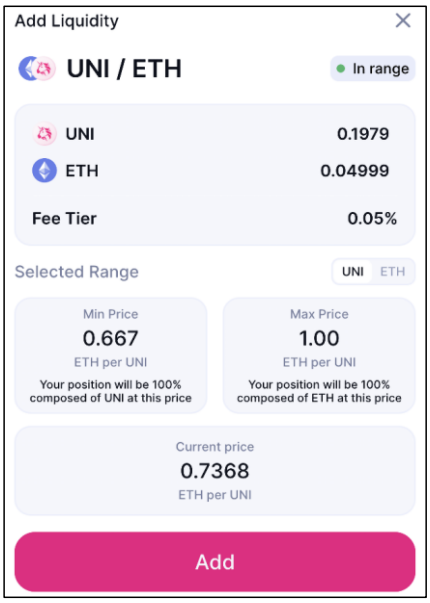

Price oracle
Price feeds
DeFi Loan
smart contract
contains assets
Lender
Deposits tokens and
redeems the principal
loan amount and
interest
Deposits collateral and
borrows tokens
Borrower
Liquidates and earns
discount
Liquidator


Görli
0×64b5...a6A0
0x64...a6A0
ETH Balance
0.0833 görETH
View and sell NFTs
Buy crypto
Transactions
Language
EN
Dark theme

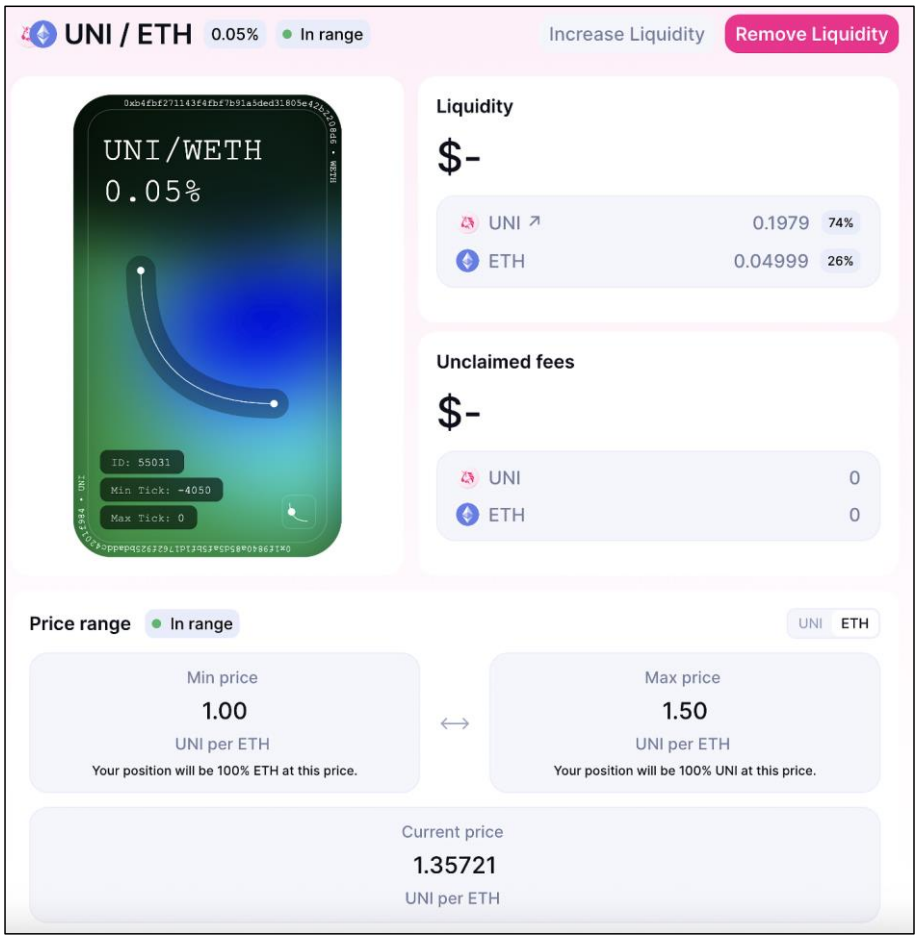

Swap
Tokens
NFTs
Pool
Search tokens and NFT collections
Swap
0.05
ETH
Balance: 0.1521 Max
0.06783
UNI
Balance: 0.285
1 UNI = 0.73715 ETH
Swap
Confirm Swap
0.05
ETH
0.0678274
UNI
1 UNI = 0.73716 ETH
Expected Output
0.0678274 UNI
Price Impact
0.00%
Minimum received after slippage (0.50%)
0.06749 UNI
Price Updated
Accept
Output is estimated. You will receive at least 0.06749 UNI or the transaction will revert.
Confirm Swap


Swap
Tokens
NFTs
Pool
Search tokens and NFT collections
Pools
More
+ New Position
Create a pool
Migrate
V2 liquidity
Learn
Your active V3 liquidity positions will appear here.

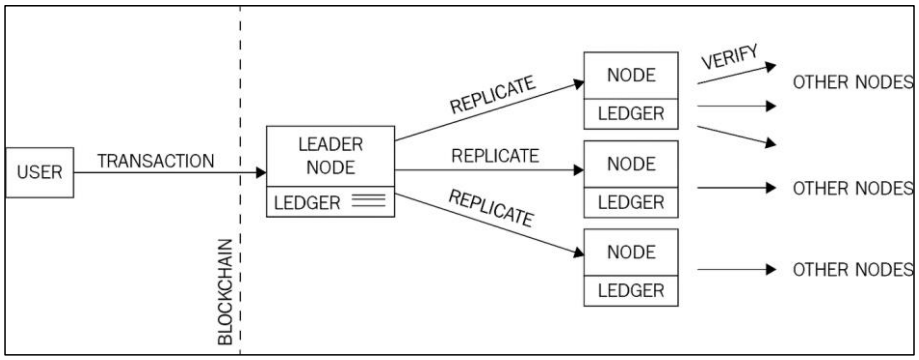

Add Liquidity
Clear All
UNI
ETH
Select Pair
ETH
UNI
0.05% fee tier
0% select
Hide
0.05%
Best for stable pairs.
0% select
0.3%
Best for most pairs.
0% select
1%
Best for exotic pairs.
0% select
Deposit Amounts
0.05
ETH
Balance: 0.1015 MAX
0.197936
UNI
Balance: 0.3528 MAX
Set Price Range
Current Price: 1.35721 UNI per ETH
Your position will appear here.
Min Price
1
UNI per ETH
Max Price
1.4993
UNI per ETH
Full Range
Preview


Add Liquidity
UNI / ETH
In range
UNI
0.1979
ETH
0.04999
Fee Tier
0.05%
Selected Range
UNI
ETH
Min Price
0.667
ETH per UNI
Your position will be 100% composed of UNI at this price
Max Price
1.00
ETH per UNI
Your position will be 100% composed of ETH at this price
Current price
0.7368
ETH per UNI
Add


Pools
More
+ New Position
Your positions (5)
Hide closed positions
RBK / ETH 0.05%
In range
Min: <0.001 RBK per ETH ⟷ Max: 1.00 RBK per ETH
ADI / ETH 0.05%
In range
Min: <0.001 ADI per ETH ⟷ Max: 1.00 ADI per ETH
ETH / RBK 0.3%
In range
Min: 19.96 ETH per RBK ⟷ Max: 100.26 ETH per RBK
UNI / ETH 0.05%
In range
Min: 1.00 UNI per ETH ⟷ Max: 1.50 UNI per ETH
UNI / ETH 0.05%
Closed
Min: 0 UNI per ETH ⟷ Max: ∞ UNI per ETH


UNI / ETH
0.05%
In range
Increase Liquidity
Remove Liquidity
UNI/WETH
0.05%
ID: 55031
Min Tick: -4050
Max Tick: 0
Liquidity
$-
UNI ↗
0.1979
74%
ETH
0.04999
26%
Unclaimed fees
$-
UNI
0
ETH
0
Price range
In range
UNI
ETH
Min price
1.00
UNI per ETH
Your position will be 100% ETH at this price.
↔
Max price
1.50
UNI per ETH
Your position will be 100% UNI at this price.
Current price
1.35721
UNI per ETH

Chapter 22: Blockchain Applications and What's Next

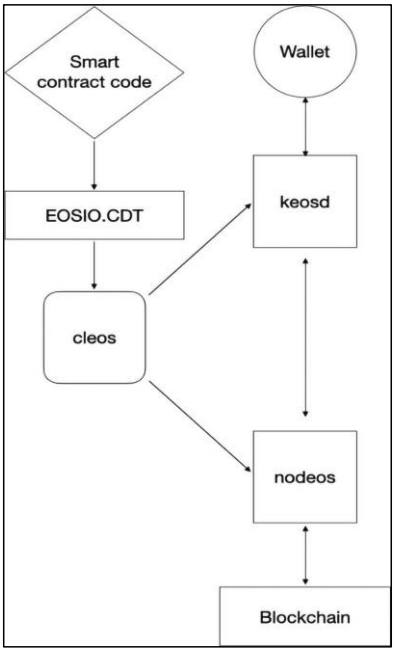


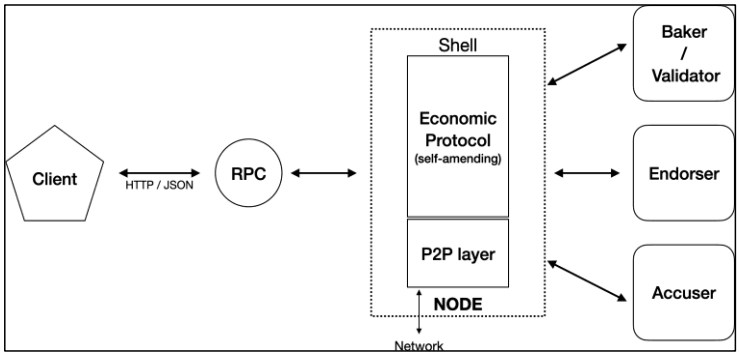

```
drequinox@drequinox-OP7010:-/Downloads$ . /pact
pact> 1234
1234
pact> (+ 1 2)
3
pact> (if (= (± 1 2) 3 "OK" "ERROR")
(interactive) :1:31: error: unexpected
    EOF, expected: ")", ";", "{",
    Boolean false, Boolean true,
    Decimal literal, Integer literal,
    String literal, Symbol literal,
    list literal, pact, sexp, space
(if (= (+ 1 2) 3 "OK" "ERROR")<E0F>

pact> (if (= (+1 2) 3) "OK" "ERROR")
"OK"
pact>
```

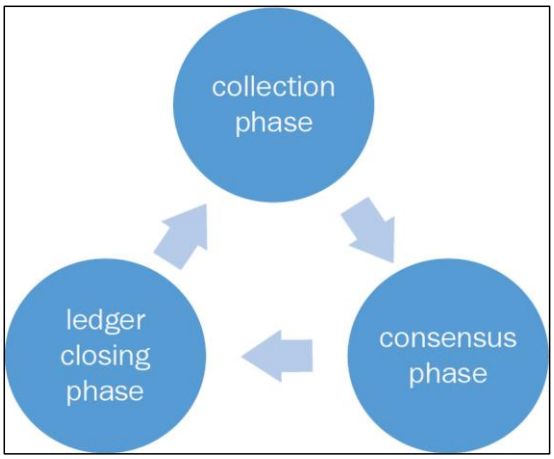
```
"Define keyset"
(define-keyset 'admin-keyset (read-keyset "admin-keyset"))
"Define module"
(module addition 'admin-keyset
"Define function addition that takes 3 argument of type integer"
  (defun addition (x y z : integer)
"Run format"
    (format "Result : {}"  [( + x ( + y z ) )])
)
)
"Run addition function with three arbitrary numbers to add"
(addition 432 4562 87)
```

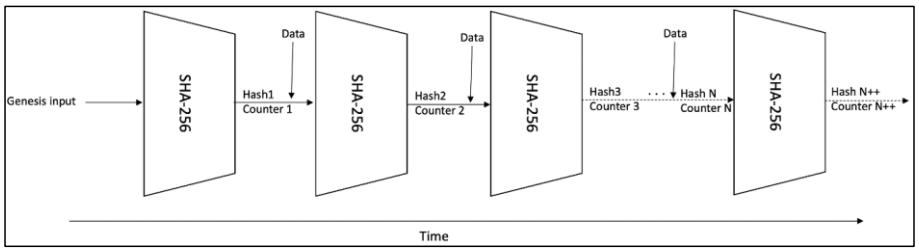
Env | REPL | Messages | Module Explorer

```
;; Welcome to the Pact interactive repl
;; Use 'LOAD into REPL' button to execute editor text
;; then just type at the "pact>" prompt to interact!
;;
;; To reset the REPL type 'reset'!
"Result : 5081"
pact>
```

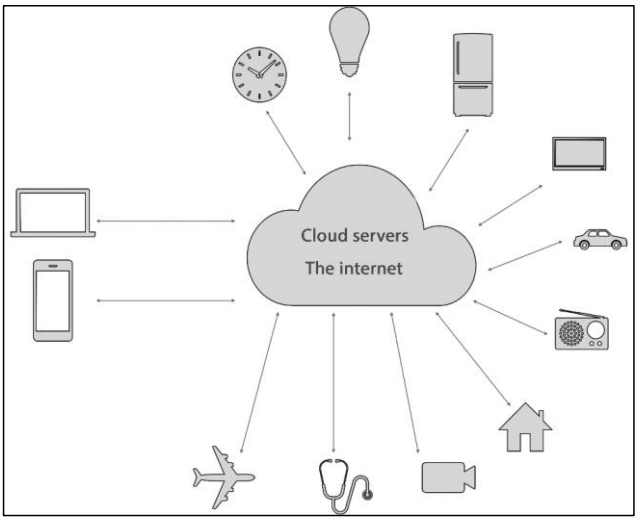


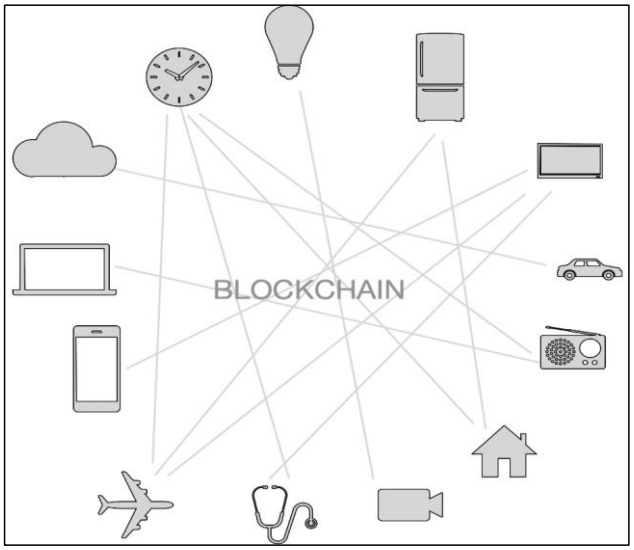


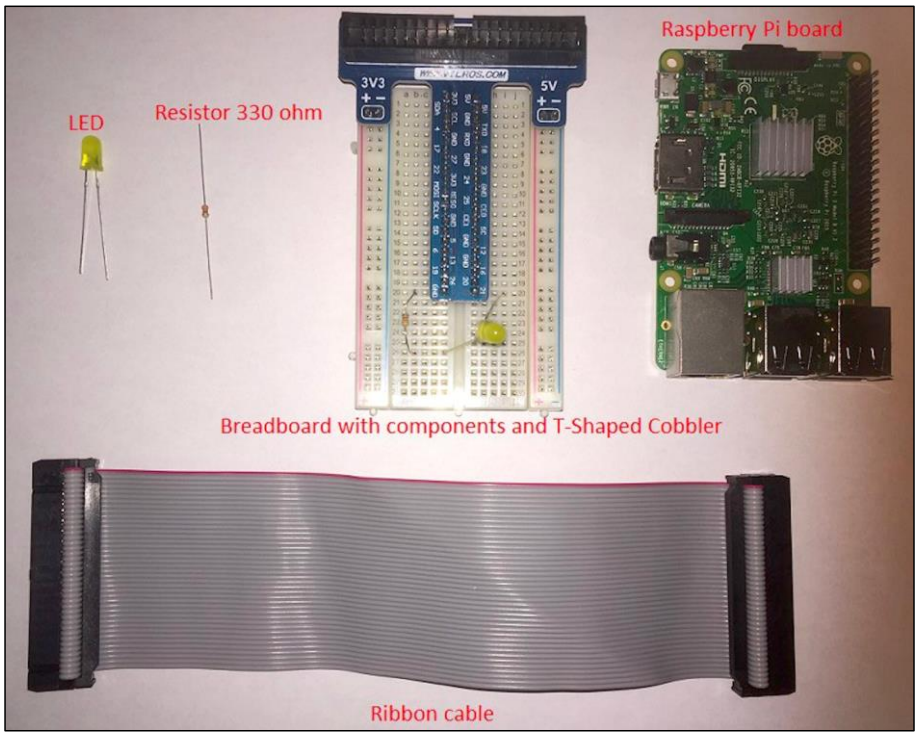

collection phase
consensus phase
ledger closing phase


Genesis input
SHA-256
Hash1
Counter 1
Data
SHA-256
Hash2
Counter 2
Data
SHA-256
Hash3
Counter 3
Hash N
Counter N
Data
SHA-256
Hash N++
Counter N++
Time

Chapter 23: Alternative Blockchains





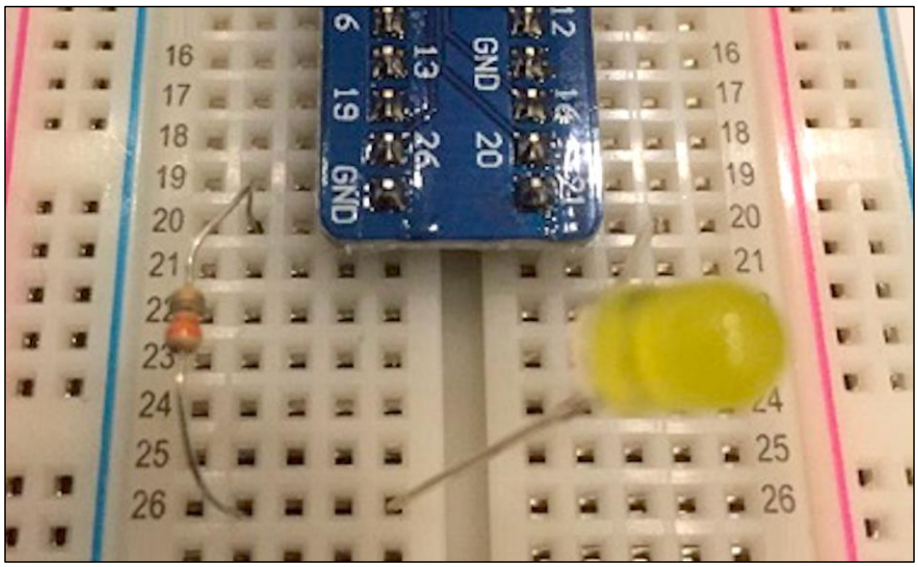


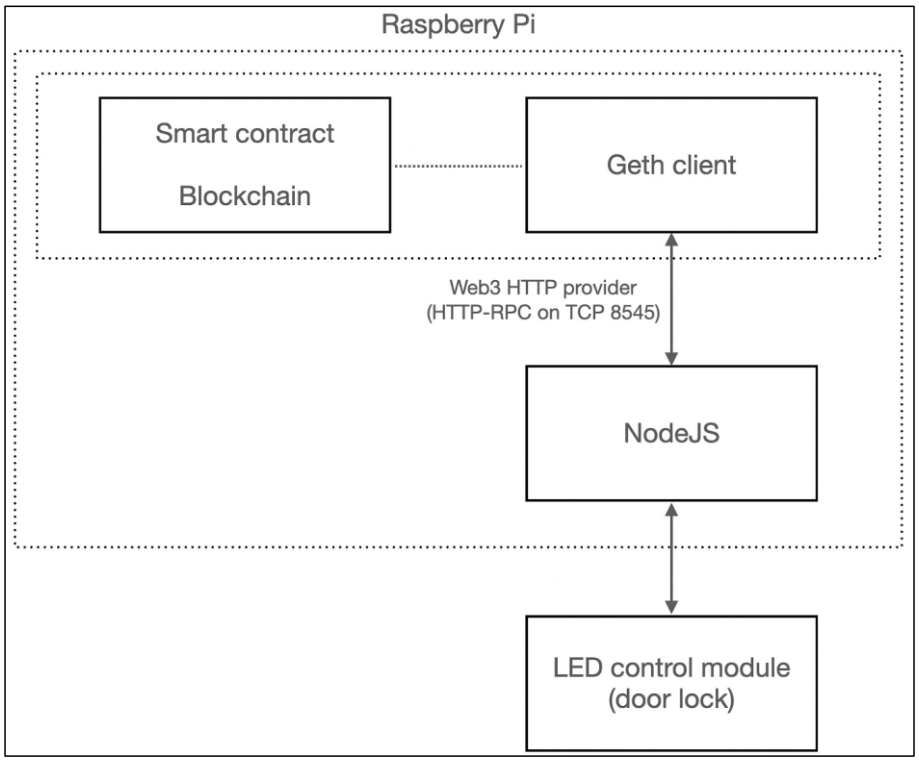

```
> admin.nodeInfo
{
  enode: "enode://44352ede5b9e792e437c1c0431c1578ce3676a87e1f588434aff1299d30325c233c8d426fc57a25380481c8a36fb3
87375e932fb4885885f6452f6efa77f@[::]:30301",
  id: "44352ede5b9e792e437c1c0431c1578ce3676a87e1f588434aff1299d30325c233c8d426fc57a25380481c8a36fbne2787375e9
4885885f6452f6efa77f",
```

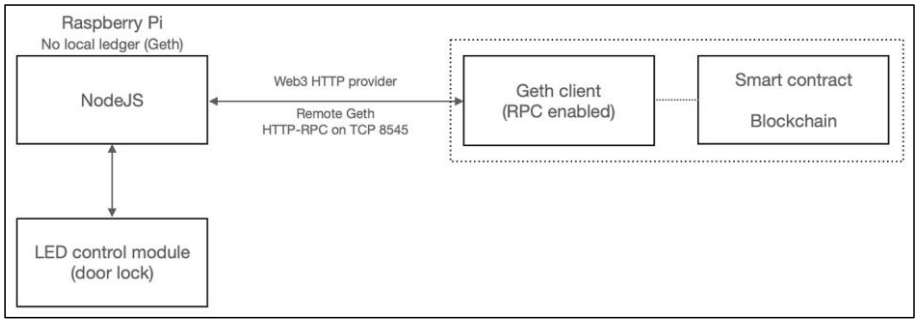
```
pi@raspberrypi:~/.ethereum $ cat static-nodes.json
[
"enode://44352ede5b9e792e437c1c0431c1578ce3676a87e1f588434aff1299d30325c233c8d426fc
57a25380481c8a36fb3be2787375e932fb4885885f6452f6efa77f@92.168.0.19:30301"
]
```

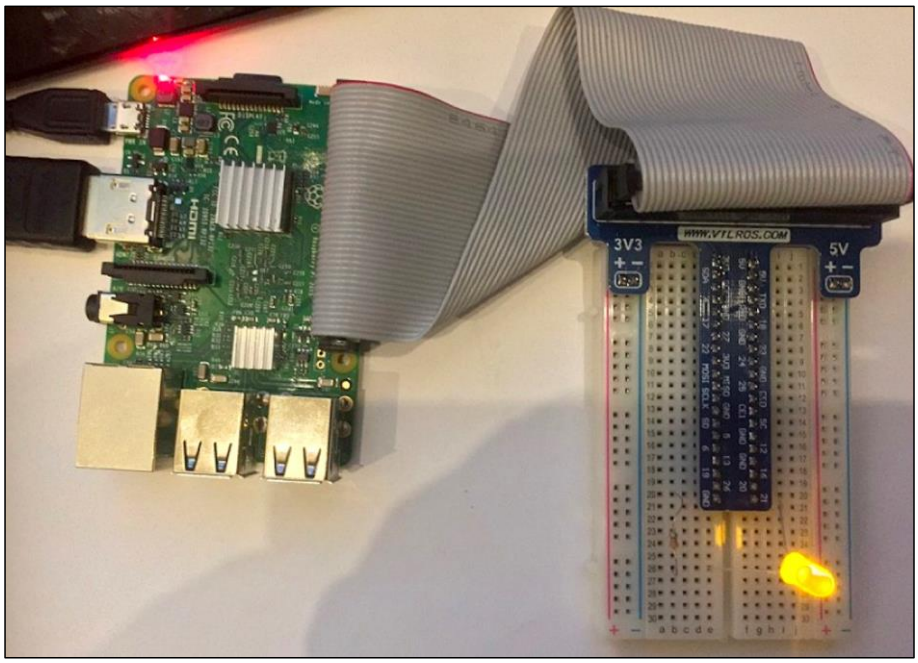
```
> admin.peers
[{
    caps: ["eth/62",  "eth/63"],
    id: "44352ede5b9e792e437c1c0431c1578ce3676a87e1f588434aff1299d30325c233c8d426fc57a25380481c8a36fb3be2787375e932f
b4885885f6452f6efa77f",
    name: "Geth/drequtnox/v1.5.2-stable-c8695209/ltnux/go1.7.3",
    network: {
      localAddress: "192.168.0.21:56550",
      remoteAddress: "192.168.0.19:30301"
    },
    protocols: {
      eth: {
        difficulty: 117194155397,
        head: "0x2d32c90b4c9dacea9a109b0ae52c1ebf511915bb618a2d3c55a80a63852e89f6",
        version: 63
      }
    }
}]
```

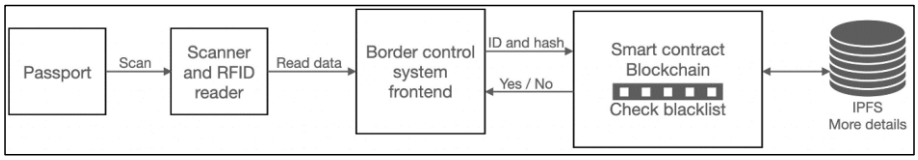
```
> admin.peers
[{
    caps: ["eth/62",  "eth/63"],
    id: "98ba36ecea7ff011803d634da45752abd25101f20a62f23427afc3f280017bc134833dd5ba400bb195ac6ed59c3b01
ca2a3f14638a52697a1bb1bf967fc84274",
    name: "Geth/raspberry/v1.5.6-stable-2a609af5/1inux/go1.7.4",
    network: {
      localAddress: "192.168.0.19:30301",
      remoteAddress: "192.168.0.21:56512"
    },
    protocols: {
      eth: {
        difficulty:  11700366137,
        head: "0x1188f58b4900a1d771d333141ea9400d78400bb8e561494ab436519ae64e1e34",
        version: 63
      }
    }
}]
```




Raspberry Pi
Smart contract
Blockchain
Geth client
Web3 HTTP provider
(HTTP-RPC on TCP 8545)
NodeJS
LED control module
(door lock)


Raspberry Pi
No local ledger (Geth)
NodeJS
Web3 HTTP provider
Remote Geth
HTTP-RPC on TCP 8545
Geth client
(RPC enabled)
Smart contract
Blockchain
LED control module
(door lock)


WWW.VILROS.COM
3V3
5V


Passport
Scan
Scanner and RFID reader
Read data
Border control system frontend
ID and hash
Yes / No
Smart contract Blockchain
Check blacklist
IPFS
More details