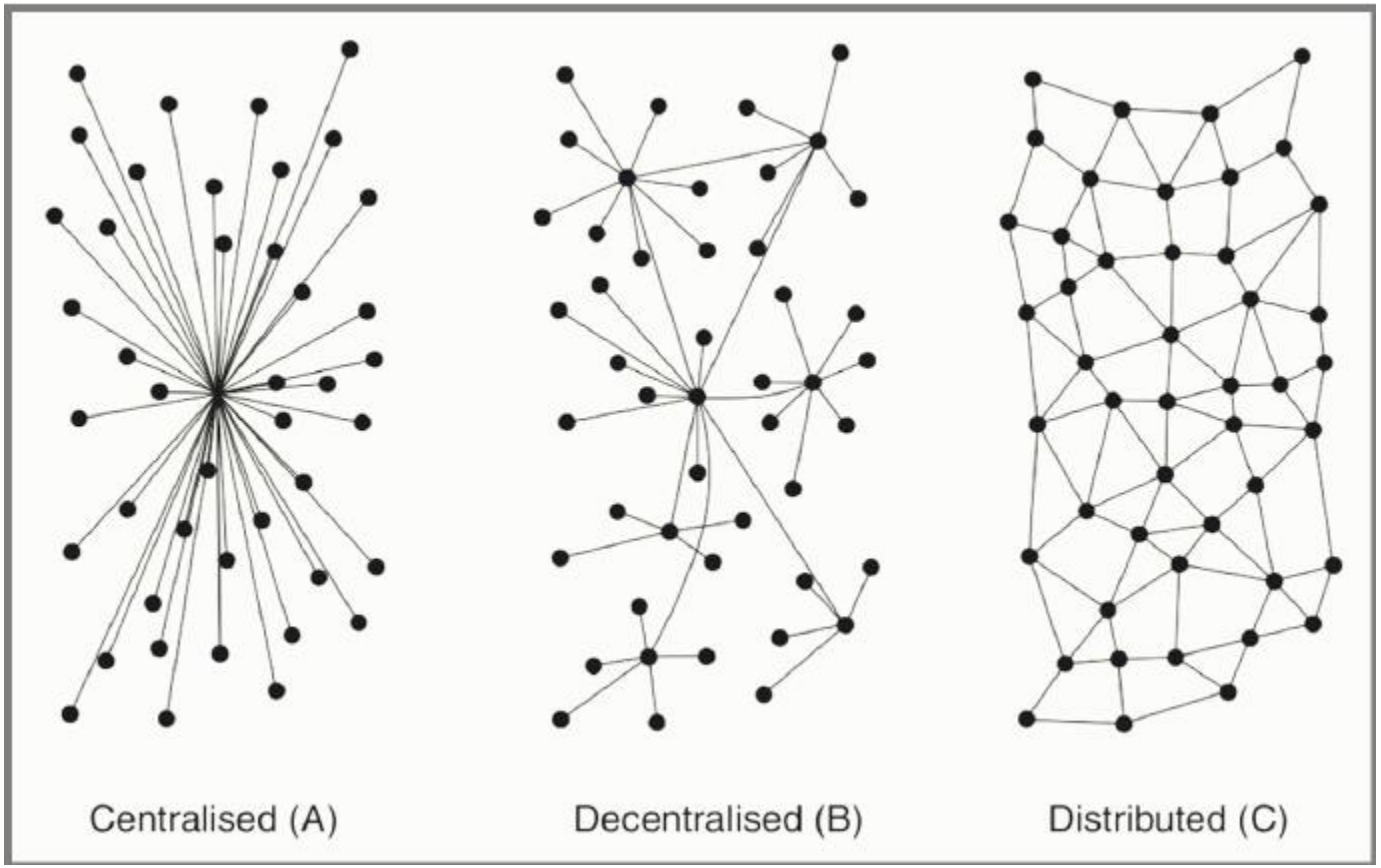
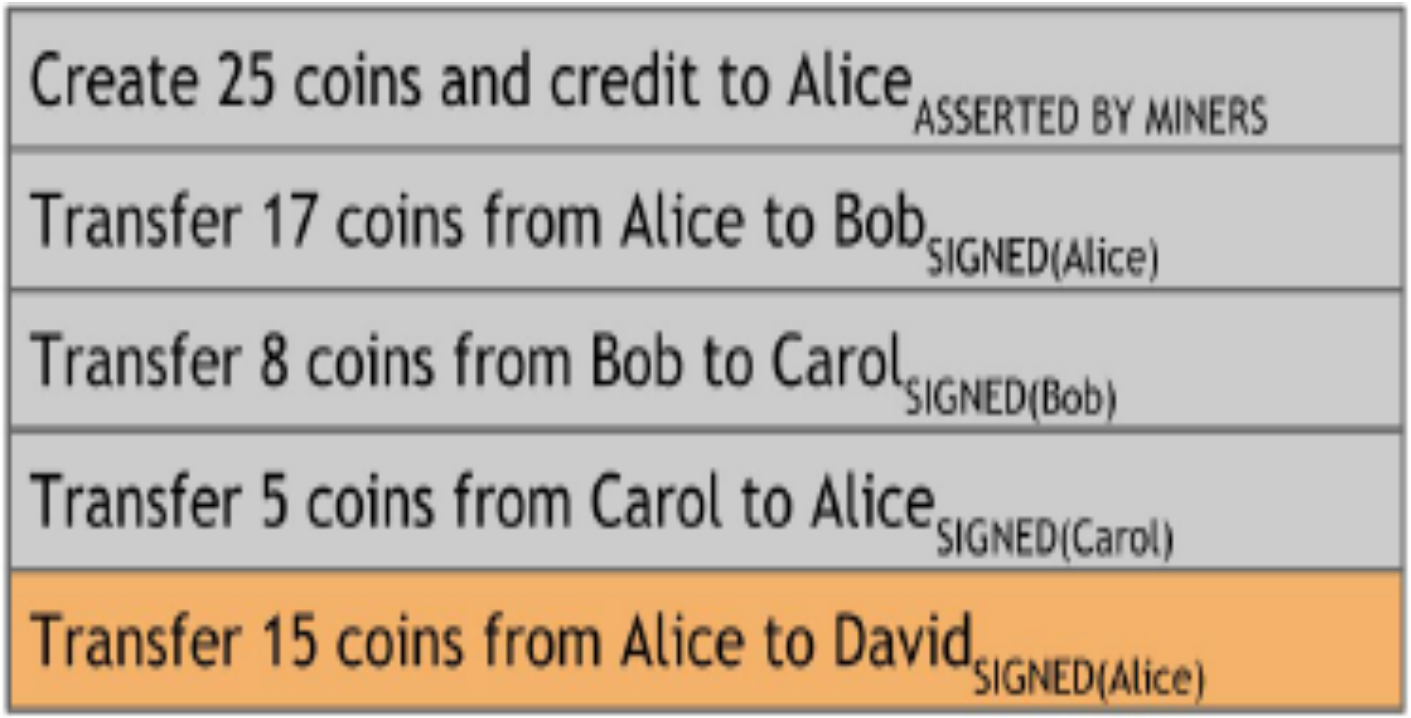
Blockchain – Smart Contracts – 1
(Aug 10, 2019)

Dr. Deven Shah

TCET Mumbai


Centralised (A)
Decentralised (B)
Distributed (C)

Bitcoin Transactions

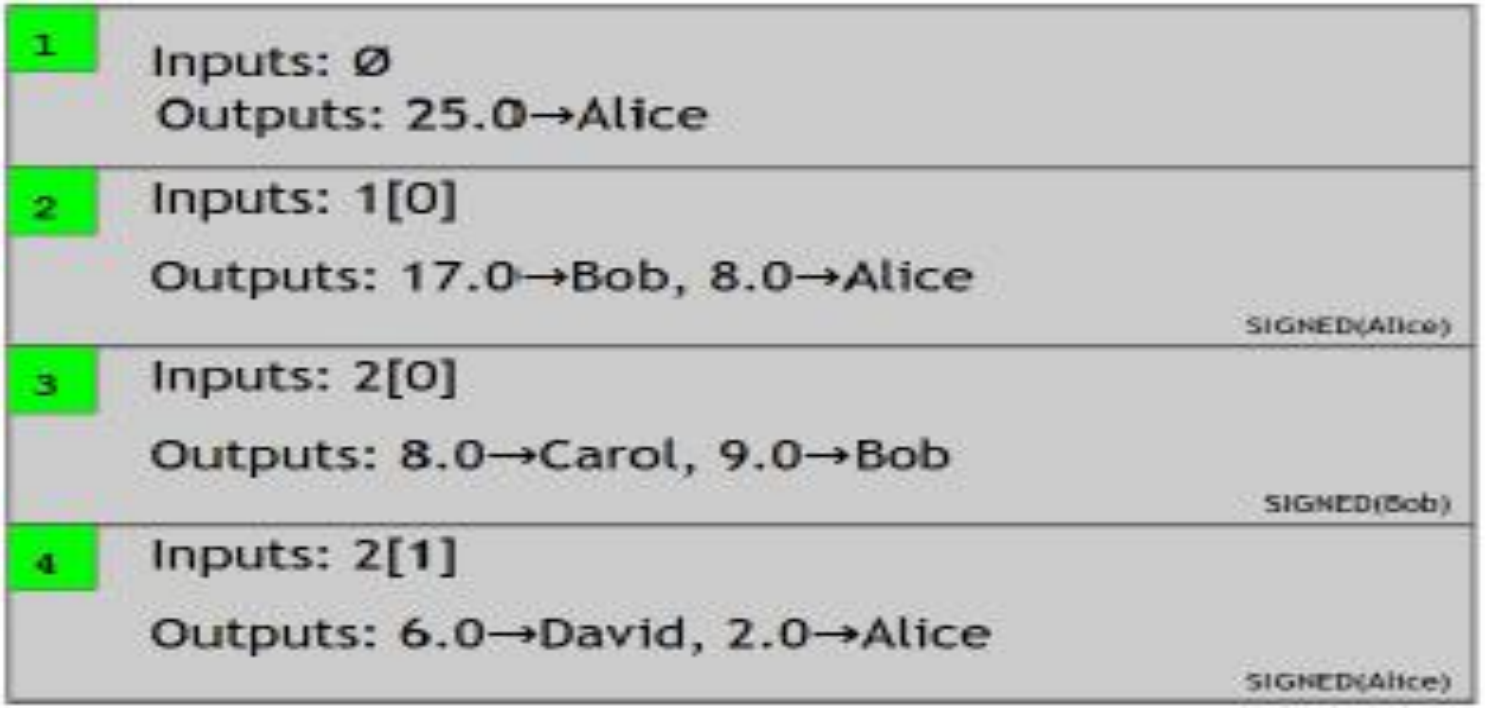


This is what we want to achieved

How to determine that transaction is valid. For that we need to keep track of all users Account balances. For e.g to validate last transaction , we have to look backwards in time Forever to see every transaction affecting Alice.

This is like account –based model and bitcoin not use it due to complexity

Transaction based ledger - Bitcoin



Transactions specify a number of inputs and outputs. Input is like coins being consumed (created in previous transaction) and output is like coin is being created.

For e.g in transaction 2 , input 1[0] refers to output 0 of transaction 1 where alice got bitcoin.

Here output is two , 17 to Bob and 8 to Alice (changed Address)

Efficient Verification though hash pointer. : We need to lookup the transaction output that Alice referenced.

Transaction

- Transaction is core of entire bitcoin system . Every thing else is designed to ensure that transaction can be
  - created
  - propagated
  - Validated
  - added to global ledger

Transaction are data structure that encode the transfer of value between participants in the bitcoin systems

Actual looks : beyond Bitcoin block explorer

- Block explorer show transaction from Jay to Veru
- along with the amount
- But actual transaction look very different : and it is not exist in bitcoin system, which we get through core command line.
  - No Jay address
  - No Viru address
  - No input amount of Jay
  - No sender or recipeint

Definition

An output in a transaction which contains two fields: a value field for transferring zero or more satoshis and a pubkey script for indicating what conditions must be fulfilled for those satoshis to be further spent.

Synonyms

- Output
- TxOut

Transactions output and Input

- TR output is fundamental building block of bitcoin
- Tr outputs are indivisible chunks of bitcoin recorded on the blockchain in entire network.
- Bitcoin full node client track all available and spendable outputs known as UTXO (Unpsent transaction output)
- Currently UTXO is in millions
- You can decide user UTXO need to use entirely along with concept of chain address

Transaction Output

- Transaction output consist of two parts

  - An amount of Bitcoin

  - A cryptographic puzzles that determine the conditions required to spend the output (called as locking script) ScriptPubKey

Transactions Inputs

- Transaction inputs identify (by Reference) which UTXO will be consumed and provide proof of ownership though unlocking script.
- Unlocking script construct to satsify the spending conditions set in the UTXO

- Unlocking script is digital signature and public key to providing ownership of bitcoin

Transaction Syntax

metadata

input(s)

output(s)

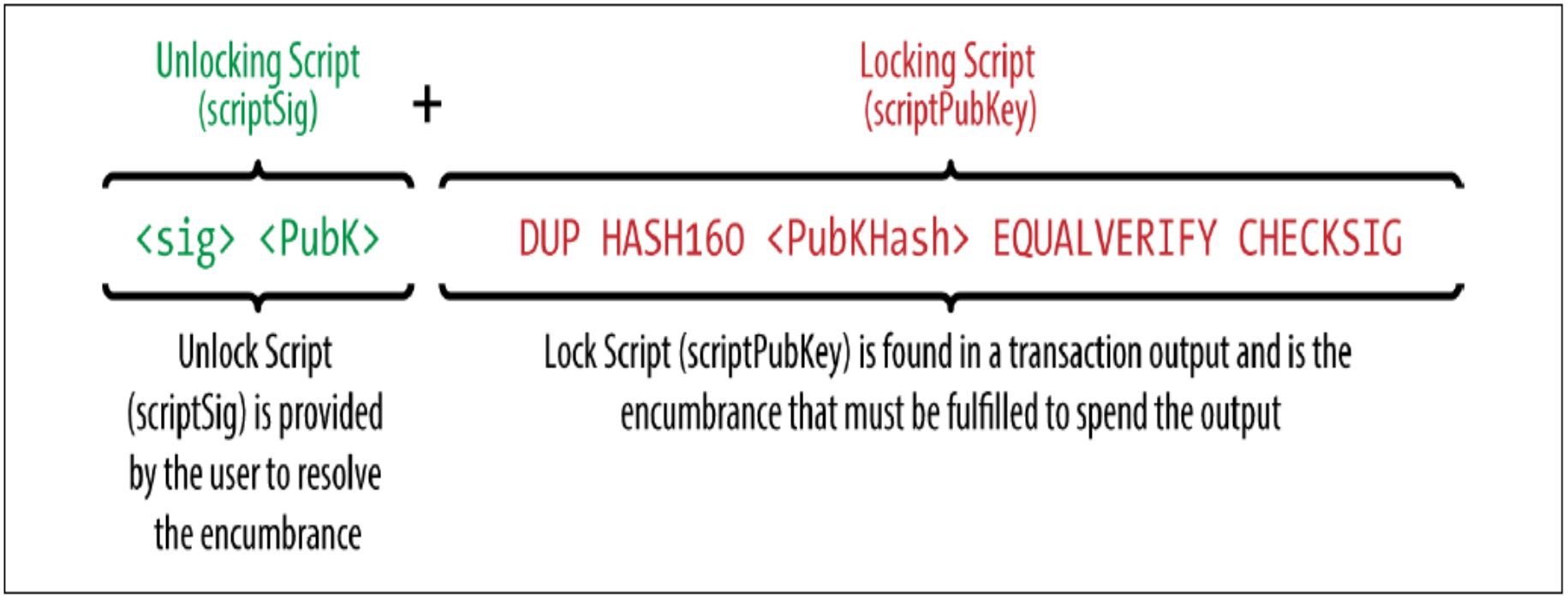

```
{
 "hash":"5a42590fbe0a90ee8e8747244d6c84f0db1a3a24e8f1b95b10c9e050990b8b6b",
 "ver":1,
 "vin_sz":2,
 "vout_sz":1,
 "lock_time":0,
 "size":404,
 "in":[
  {
   "prev_out":{
    "hash":"3be4ac9728a0823cf5e2deb2e86fc0bd2aa503a91d307b42ba76117d79280260",
    "n":0
   },

    "scriptSig":"30440..."
  },
  {
   "prev_out":{
    "hash":"7508e6ab259b4df0fd5147bab0c949d81473db4518f81afc5c3f52f91ff6b34e",
    "n":0
   },
   "scriptSig":"3f3a4ce81...."
  }
 ],
 "out":[
  {
   "value":"10.12287097",
   "scriptPubKey":"OP_DUP OP_HASH160 69e02e18b5705a05dd6b28ed517716c894b3d42e OP_EQUALVERIFY OP_CHECKSIG"
  }
 ]
}
```

Bitcoin Scripts

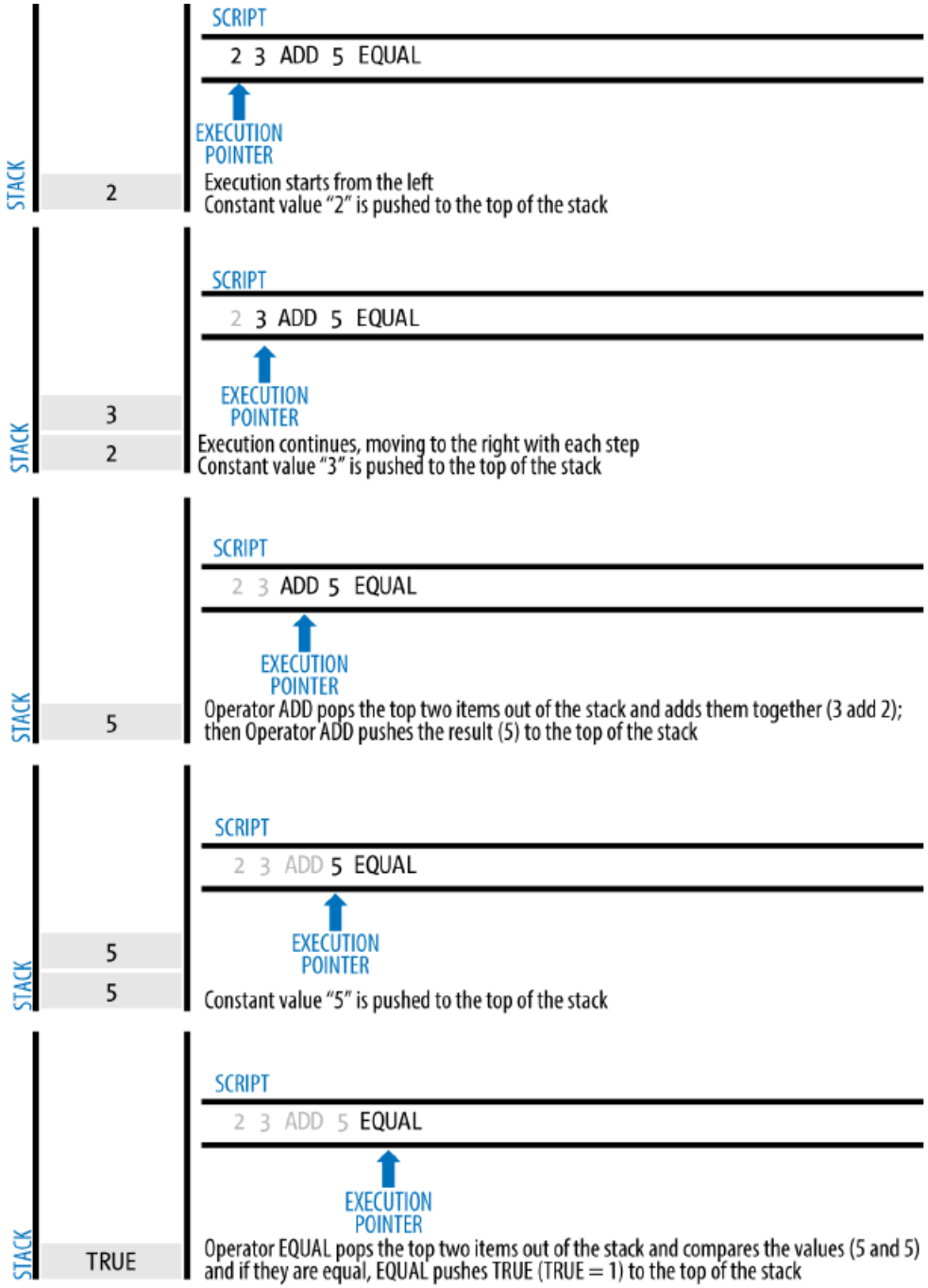
- The most common type of transactions in Bitcoin is to redeem a previous transaction output by signing with the correct key.
- i.e “ this can be redeemed by a public key that hashes to X, along with a signature from the owner of that public key”
- So To validate that a transaction redeems a previous transaction output correctly, we need to combine the new transactions input script and earlier transaction’s output script
- Scriptpubkey : output script specifies public key (hashes)
- ScriptSig : input script specifies a signature to public key.

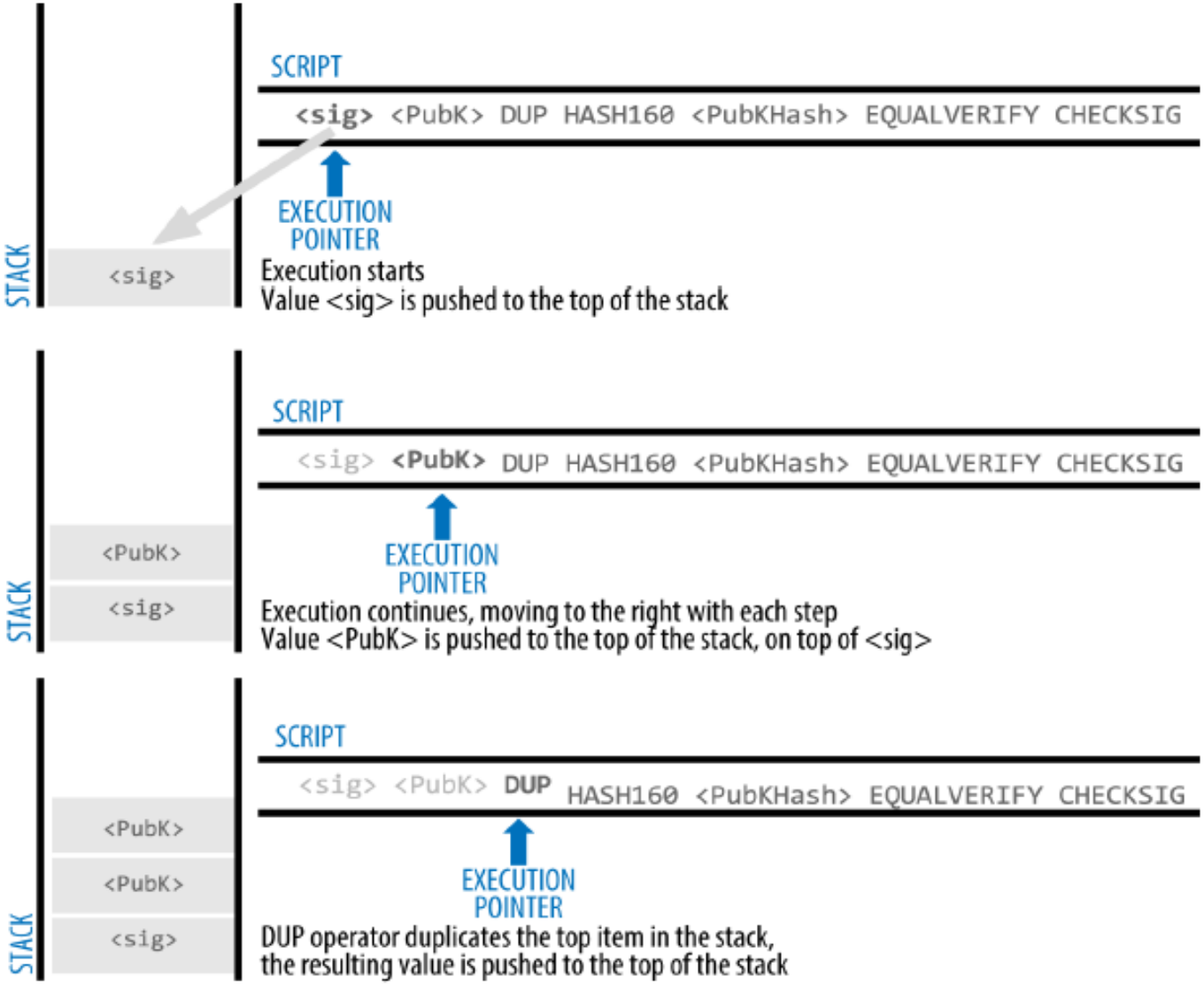
Script Construction (Lock + Unlock)

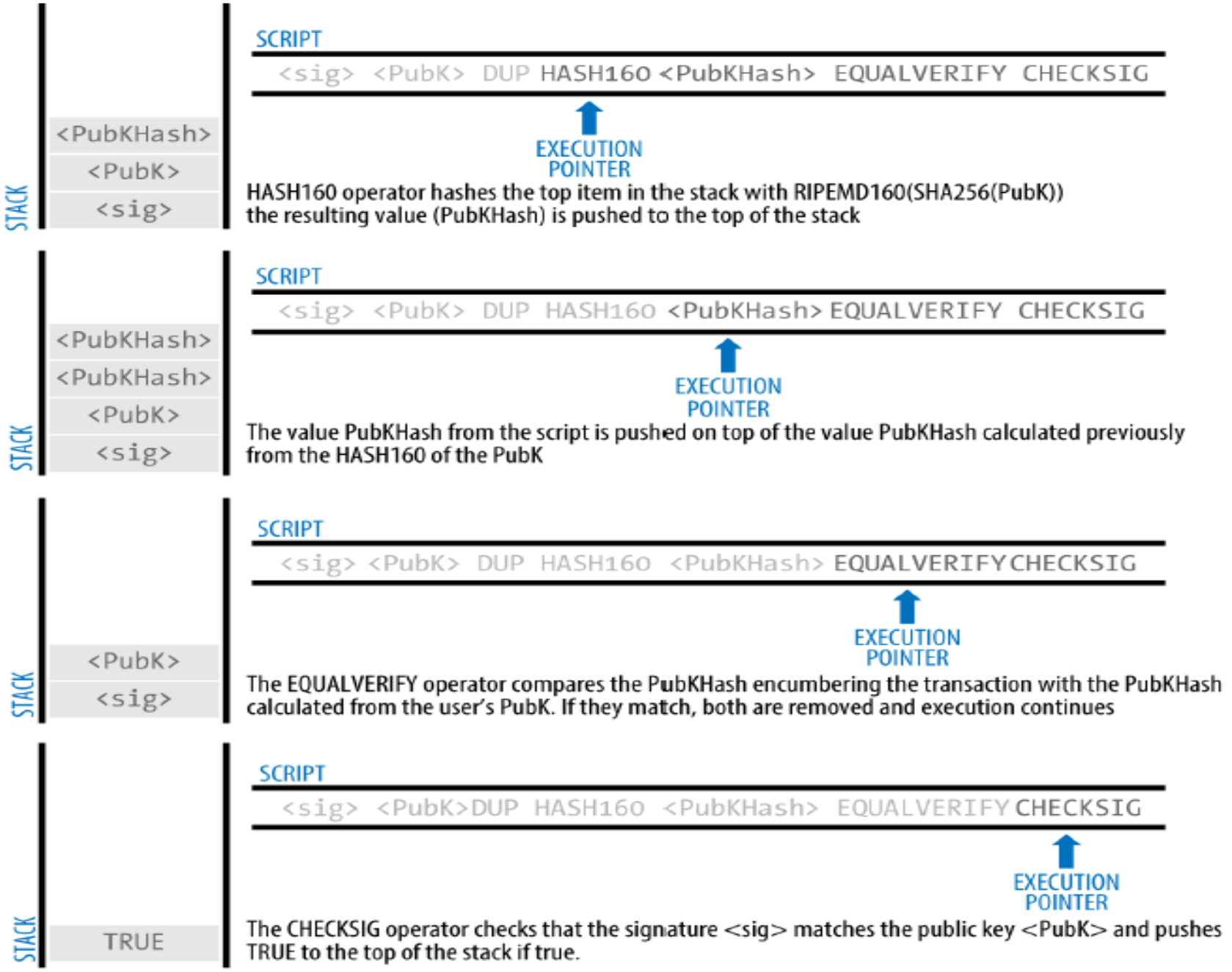
- Bitcoin transaction Validation engine relies on two type of scripts to validate transactions .
- A locking script (Output) and an unlocking script (input)
- Locking Script: “scriptPubkey”
- Unlocking script: “ scriptseg” contain digital signature



Forth like script language, which is stack based


SCRIPT
2 3 ADD 5 EQUAL
EXECUTION POINTER
STACK
2
Execution starts from the left
Constant value "2" is pushed to the top of the stack
SCRIPT
2 3 ADD 5 EQUAL
EXECUTION POINTER
STACK
3
2
Execution continues, moving to the right with each step
Constant value "3" is pushed to the top of the stack
SCRIPT
2 3 ADD 5 EQUAL
EXECUTION POINTER
STACK
5
Operator ADD pops the top two items out of the stack and adds them together (3 add 2); then Operator ADD pushes the result (5) to the top of the stack
SCRIPT
2 3 ADD 5 EQUAL
EXECUTION POINTER
STACK
5
5
Constant value "5" is pushed to the top of the stack
SCRIPT
2 3 ADD 5 EQUAL
EXECUTION POINTER
STACK
TRUE
Operator EQUAL pops the top two items out of the stack and compares the values (5 and 5) and if they are equal, EQUAL pushes TRUE (TRUE = 1) to the top of the stack

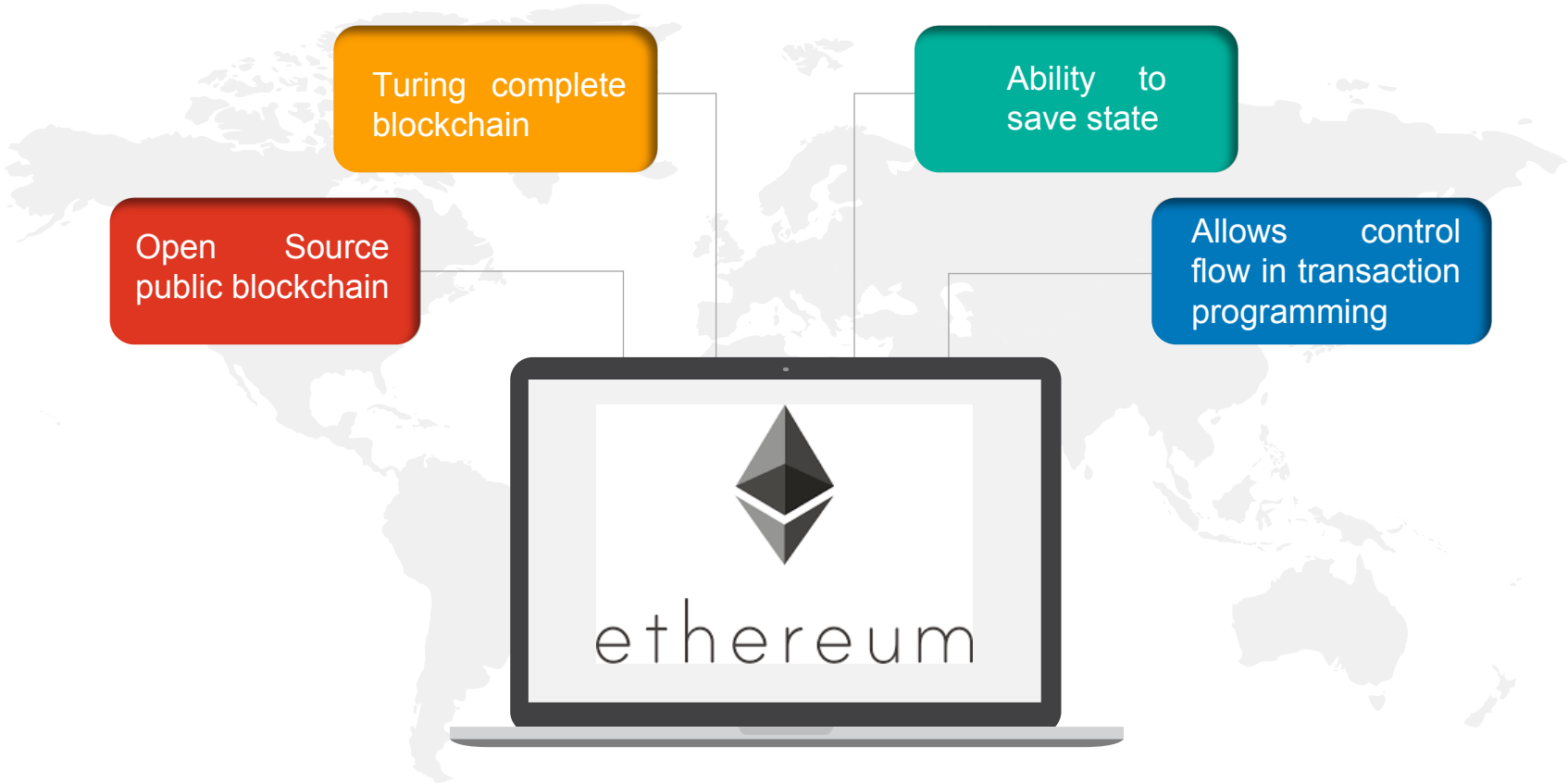

SCRIPT
<sig> <PubK> DUP HASH160 <PubKHash> EQUALVERIFY CHECKSIG
EXECUTION POINTER
STACK
<sig>
Execution starts
Value <sig> is pushed to the top of the stack
SCRIPT
<sig> <PubK> DUP HASH160 <PubKHash> EQUALVERIFY CHECKSIG
EXECUTION POINTER
STACK
<PubK>
<sig>
Execution continues, moving to the right with each step
Value <PubK> is pushed to the top of the stack, on top of <sig>
SCRIPT
<sig> <PubK> DUP HASH160 <PubKHash> EQUALVERIFY CHECKSIG
EXECUTION POINTER
STACK
<PubK>
<PubK>
<sig>
DUP operator duplicates the top item in the stack,
the resulting value is pushed to the top of the stack


STACK
<PubKHash>
<PubK>
<sig>
SCRIPT
<sig> <PubK> DUP HASH160 <PubKHash> EQUALVERIFY CHECKSIG
EXECUTION POINTER
HASH160 operator hashes the top item in the stack with RIPEMD160(SHA256(PubK)) the resulting value (PubKHash) is pushed to the top of the stack
STACK
<PubKHash>
<PubKHash>
<PubK>
<sig>
SCRIPT
<sig> <PubK> DUP HASH160 <PubKHash> EQUALVERIFY CHECKSIG
EXECUTION POINTER
The value PubKHash from the script is pushed on top of the value PubKHash calculated previously from the HASH160 of the PubK
STACK
<PubK>
<sig>
SCRIPT
<sig> <PubK> DUP HASH160 <PubKHash> EQUALVERIFY CHECKSIG
EXECUTION POINTER
The EQUALVERIFY operator compares the PubKHash encumbering the transaction with the PubKHash calculated from the user's PubK. If they match, both are removed and execution continues
STACK
TRUE
SCRIPT
<sig> <PubK>DUP HASH160 <PubKHash> EQUALVERIFY CHECKSIG
EXECUTION POINTER
The CHECKSIG operator checks that the signature <sig> matches the public key <PubK> and pushes TRUE to the top of the stack if true.

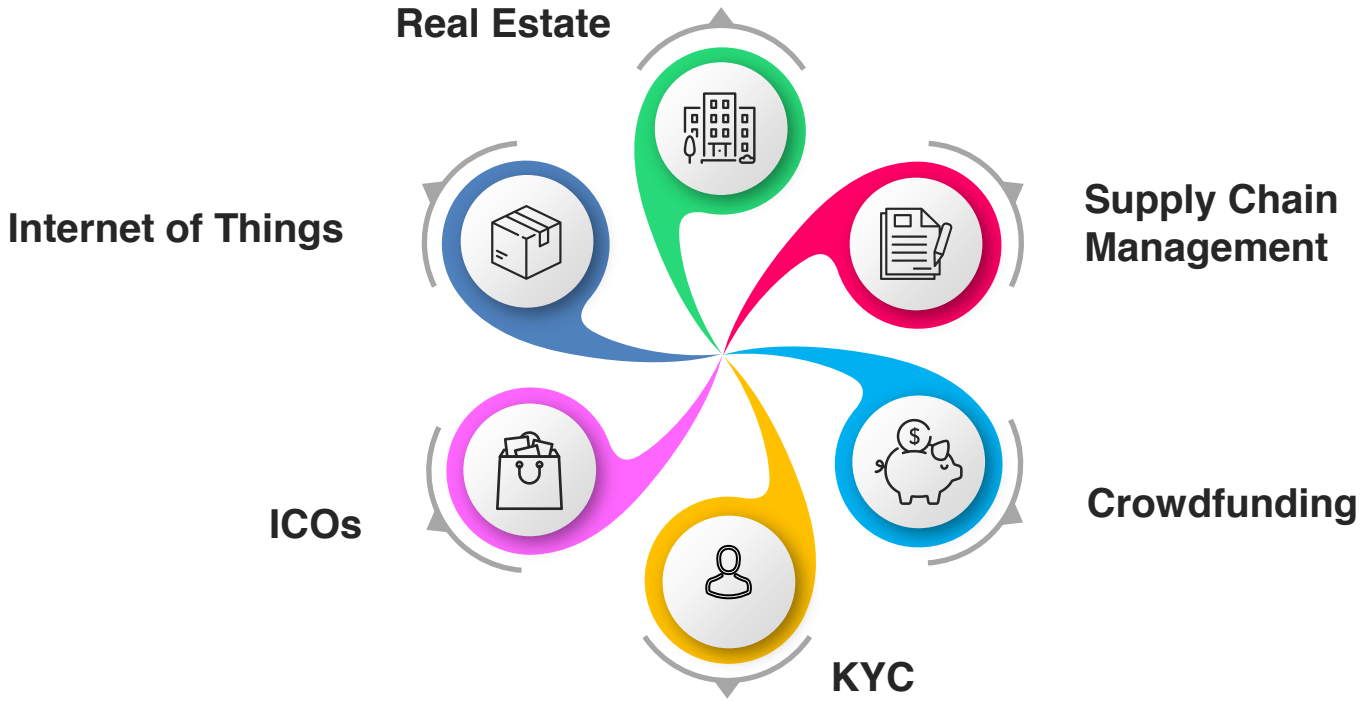
OP_CHECKSIG is used directly without first hashing the public key. This was used by early versions of Bitcoin where people paid directly to IP addresses, before Bitcoin addresses were introduced. scriptPubKeys of this transaction form are still recognized as payments to user by Bitcoin Core. The disadvantage of this transaction form is that the whole public key needs to be known in advance, implying longer payment addresses, and that it provides less protection in the event of a break in the ECDSA signature algorithm.

scriptPubKey: <pubKey> OP_CHECKSIG
scriptSig: <sig>

BEYOND CRYPTOCURRENCY


Turing complete blockchain
Ability to save state
Open Source public blockchain
Allows control flow in transaction programming
ethereum

Applications of Ethereum



BITCOIN AND ETHEREUM: **SIMILARITIES**

Features	Bitcoin	Ethereum
Has a Blockchain	Yes	Yes
Main net is Public and Permission less	Yes	Yes
Proof of Work Mining	Yes	Yes
Inbuilt Cryptocurrency	BTC	ETH

BITCOIN AND ETHEREUM: **DIFFERENCES**

Features	Bitcoin	Ethereum
Block Time	~10 minutes	14 seconds
Blocks generated per hour	6 blocks	250 blocks
Maximum Block Size	1 MB	15,00,000 Gas => 2KB
Transactions per Block	1,500-2,000	~70
Token issuance	BTC generation halves every four years	ETH generation is constant every year

BITCOIN AND ETHEREUM: **DIFFERENCES**

Features	Bitcoin	Ethereum
Mining Rewards	12.5 BTC	5 ETH
Blocks mined a little late and don't form part of blockchain	Referred as Orphans and are discarded	Referred as Unlces and can be referenced by later blocks
Built-in Programming Language	No	Solidity
Allows Control Flow	No	Yes
Smart Contract	No	Yes

SMART CONTRACTS

What is Smart Contract?

01 Pre-written Logic i.e. computer code

02 Stored and replicated on a distributed storage platform

04 Results in ledger updates

03 Executed by a network of computers

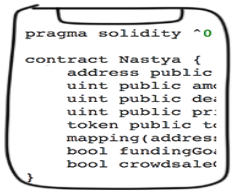
Traditional Contract vs Smart Contract

SILLY CONTRACT

- BORING OFFICIAL PAPER
- NO GUARANTEE
- KILL THE TREES
- NEEDS 50 LAWYERS

NOT YOUR BUDDY

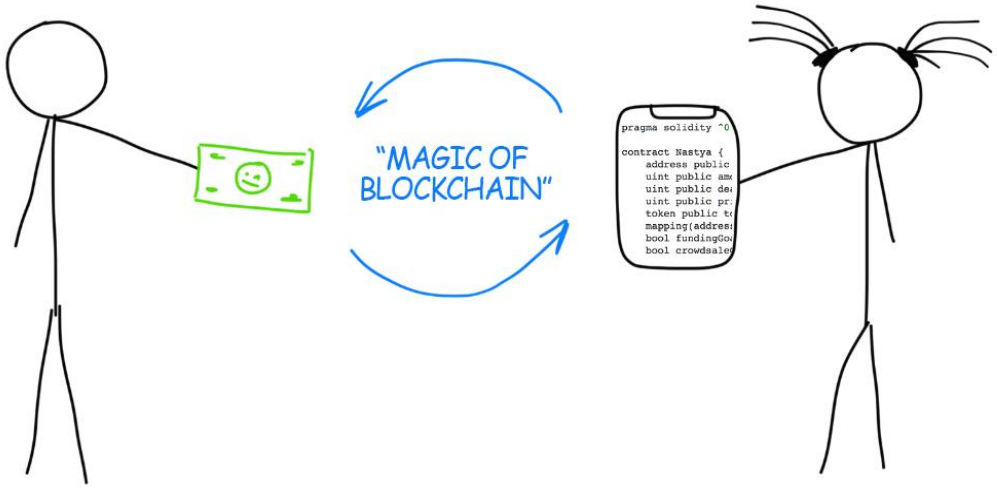
SMART CONTRACT



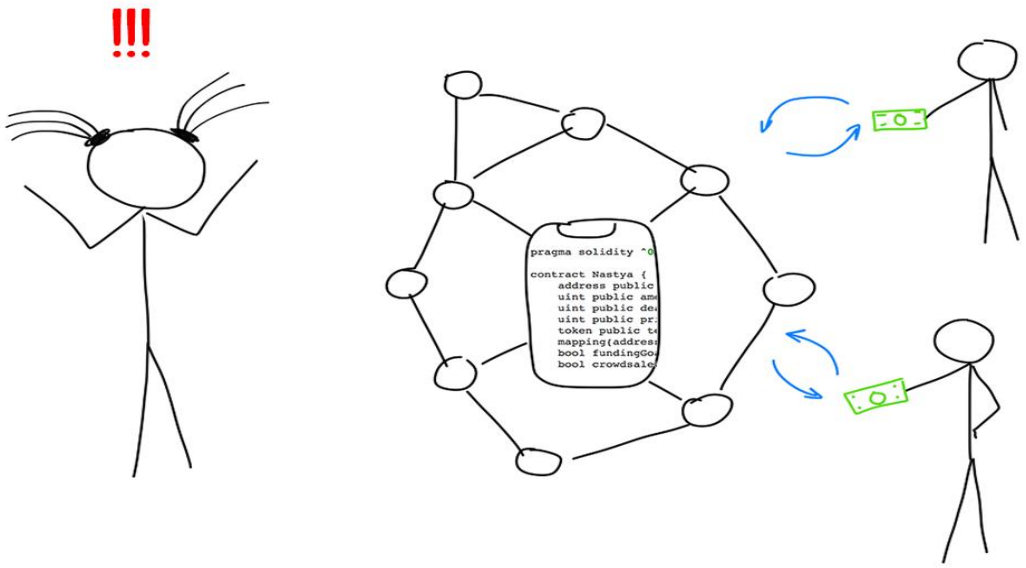
- PERFECT CODE
- VERIFIED BY MATHS
- I'M A PROGRAMMER, YOU CAN'T FOOL ME
- ANYONE CAN WRITE HIS OWN

YOUR BUDDY

Transaction through Smart Contracts



Any Business Logic on Smart Contract



Smart Contract executed on all nodes