Blockchain and Digital Currencies

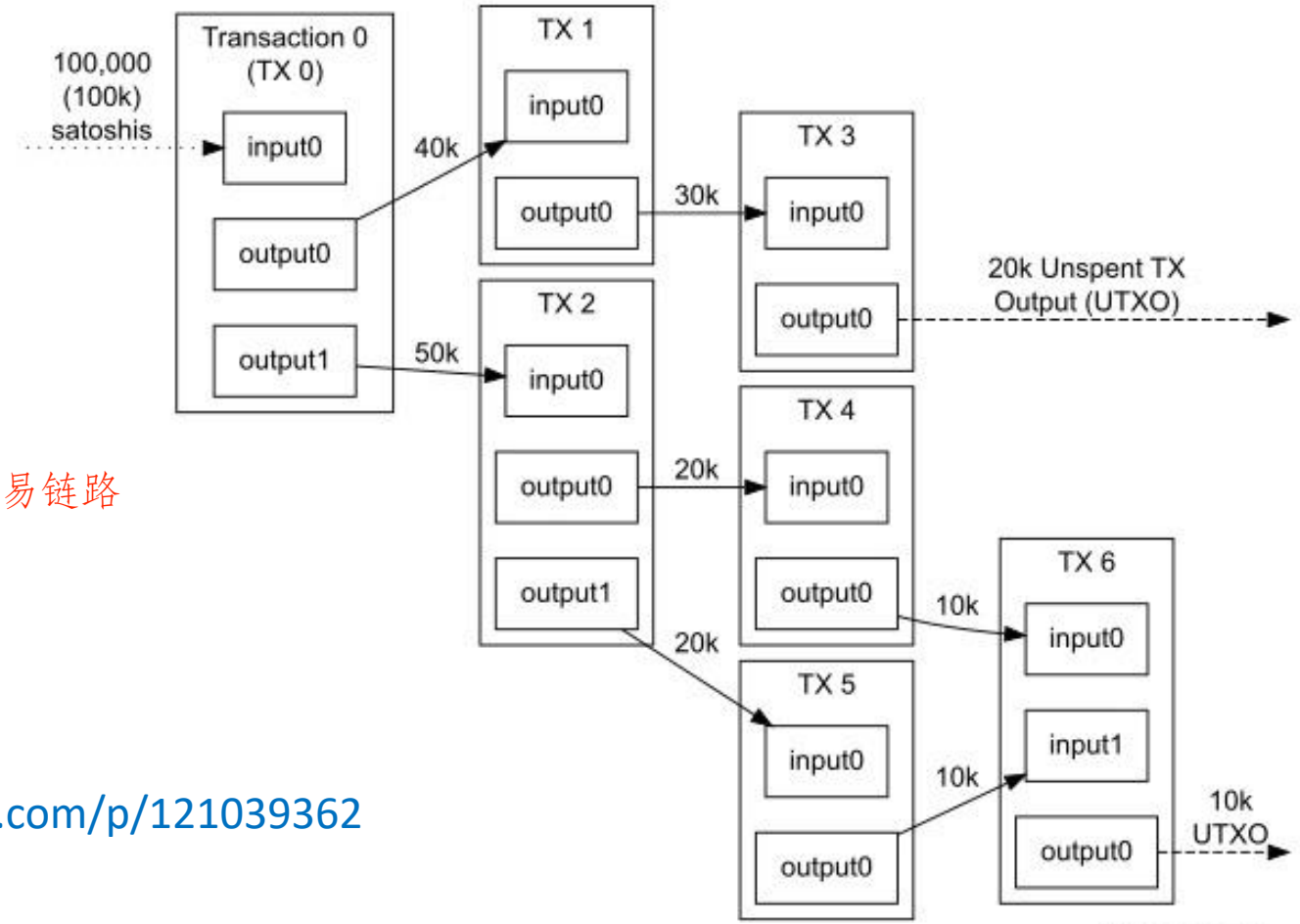
Lecture 5

PHBS 2024 M3

Agenda

- Bitcoin transcript
- Bitcoin transcript applications

Transactions Cause Ownership Transfer

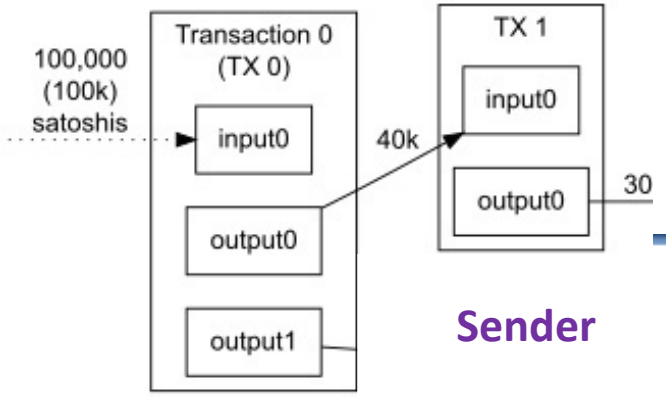


Triple-Entry Bookkeeping (Transaction-To-Transaction Payments) As Used By Bitcoin

一个较为全面的交易链路

https://zhuanlan.zhihu.com/p/121039362

Transactions Need Verification



- Transaction verifications are done by miners
- Verifications usually consist of 2 parts for every single input:
  1. The user who initiates the transaction (sender) has the money
  2. The user who initiates the transaction (sender) can use the money

- The 1st part is done by matching the sender's pubkey to the incoming transaction's destination recipient address
- The 2nd part is done by executing the concatenated signature script (scriptSig) and output script (scriptPubKey)

ScriptSig and ScriptPubKey

交易的输出

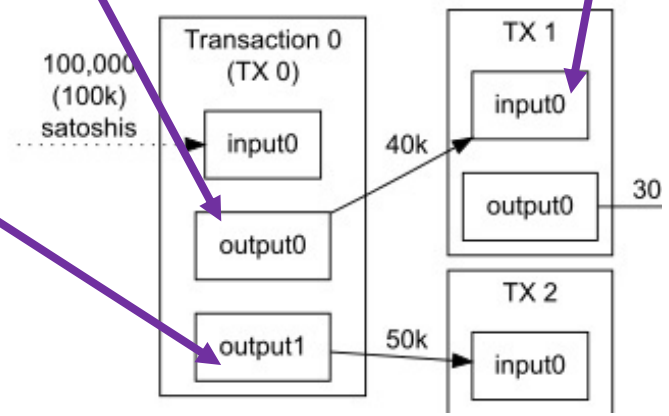
```
"vout": [{
    "value": 0.22684000,
    "n": 0,
    "scriptPubKey": {
        "asm": "DUP HASH160 628e…d743 EQUALVERIFY CHECKSIG",
        "hex": "76a9…88ac",
        "reqSigs": 1,
        "type": "pubkeyhash",
        "addresses": [ "19z8LJkNXLrTv2QK5jgTncJCGUEEfpQvSr"]
    }
  },{
    "value": 0.53756644,
    "n": 1,
    "scriptPubKey": {
        "asm": "DUP HASH160 da7d…2cd2 EQUALVERIFY CHECKSIG",
        "hex": "76a9…88ac",
        "reqSigs": 1,
        "type": "pubkeyhash",
        "addresses": ["1LvGTpdyeVLcLCDK2m9f7Pbh7zwhs7NYhX"]
    }
}],
```


使用 script 进行加密
告诉别人，钱是怎么锁住的

交易的输入

```
"vin": [{
    "txid": "c0cb…c57b",
    "vout": 0,
    "scriptSig": {
        "asm": "3045...0018",
        "hex": "4830...0018"
    },
],
```

这个 n 是和 output 的 n 匹配的



锁定脚本（scriptPubKey）
又称为输出脚本

解锁脚本（scriptSig）
又称为输入脚本

https://zhuanlan.zhihu.com/p/121039362

P2PK (Pay to Public Key)

P2PK (Pay to Public Key)

input script:

```
PUSHDATA(Sig)
```

output script:

```
PUSHDATA(PubKey)
CHECKSIG
```

https://zhuanlan.zhihu.com/p/121039362

P2PKH (Pay to public key hash)

P2PKH (Pay to Public Key Hash)

input script:

```
PUSHDATA(Sig)
PUSHDATA(PubKey)
```

output script:

```
DUP
HASH160
PUSHDATA(PubKeyHash)
EQUALVERIFY
CHECKSIG
```

必须要放出这些东西，很复杂

https://developer.bitcoin.org/devguide/transactions.html

https://zhuanlan.zhihu.com/p/121039362

P2SH(Pay to script hash)

```
...
PUSHDATA(Sig)
...
PUSHDATA(serialized redeemScript)
```

解锁脚本

```
HASH160
PUSHDATA(redeemScriptHash)
EQUAL
```

锁定脚本

用P2SH实现P2PK

redeemScript:
```
PUSHDATA(PubKey)
CHECKSIG
```
赎回脚本，反序列化后内容为P2Pk形式

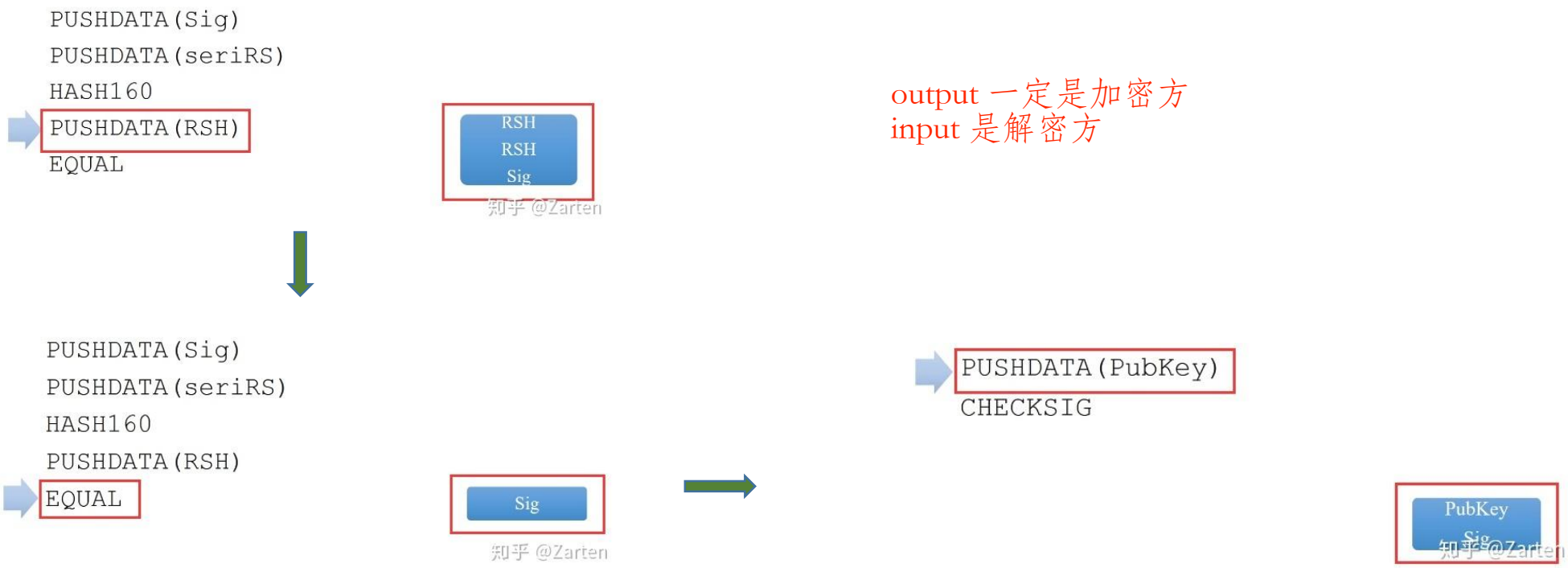
input script:
```
PUSHDATA(Sig)
PUSHDATA(serialized redeemScript)
```
解锁脚本

output script:
```
HASH160
PUSHDATA(redeemScriptHash)
EQUAL
```
锁定脚本

https://zhuanlan.zhihu.com/p/121039362

https://developer.bitcoin.org/devguide/transactions.html

P2SH(Pay to script hash) Key Steps



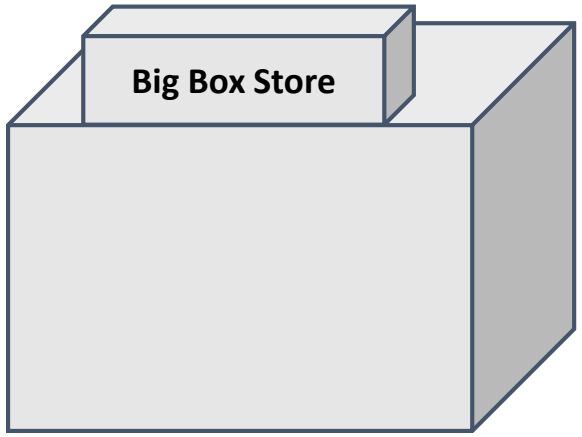
https://zhuanlan.zhihu.com/p/121039362

https://developer.bitcoin.org/devguide/transactions.html

Should Senders Specify Output Scripts?

Alice

Bob

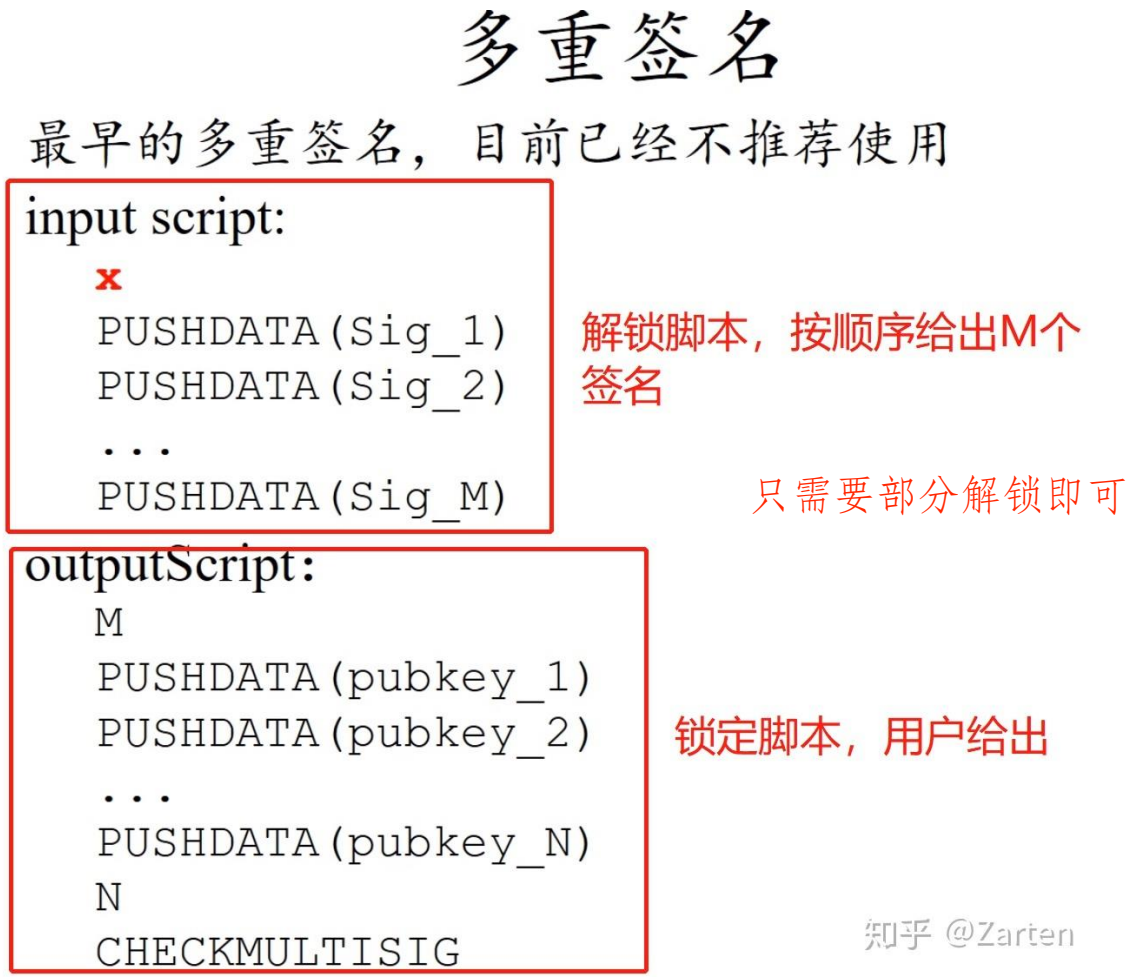


I'm ready to pay for my purchases!

Cool! Well we're using MULTISIG now, so include a script requiring 2 of our 3 account managers to approve. Don't get any of those details wrong. Thanks for shopping at Big Box!

MULTISIG – P2PK, Not Recommended

https://zhuanlan.zhihu.com/p/121039362

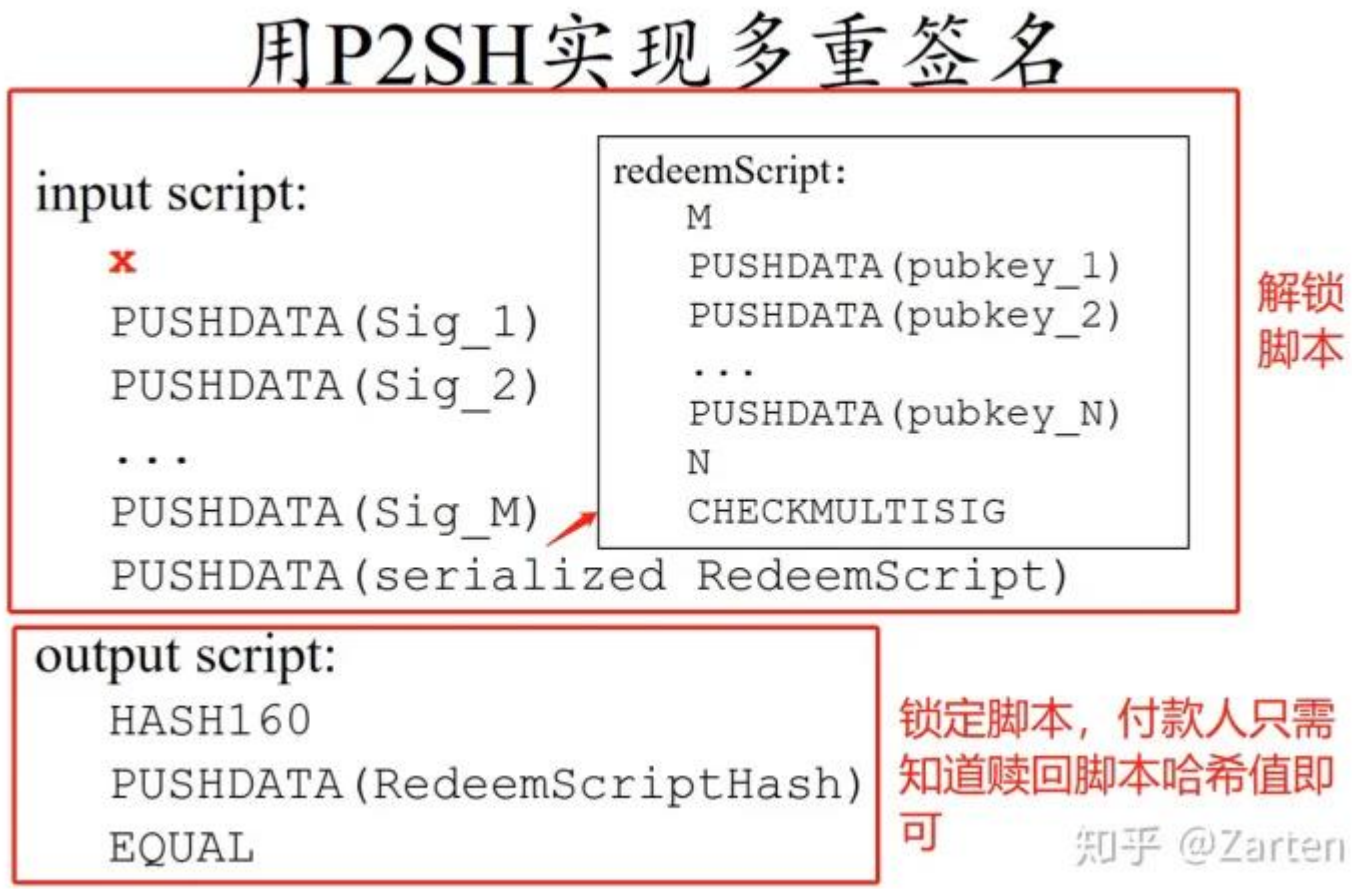


https://developer.bitcoin.org/devguide/transactions.html

MULTISIG – P2SH, Recommended

https://zhuanlan.zhihu.com/p/121039362

这是商户自己提供的
redeemScript



https://developer.bitcoin.org/devguide/transactions.html

Pay-to-Script-Hash (P2SH) Workflow

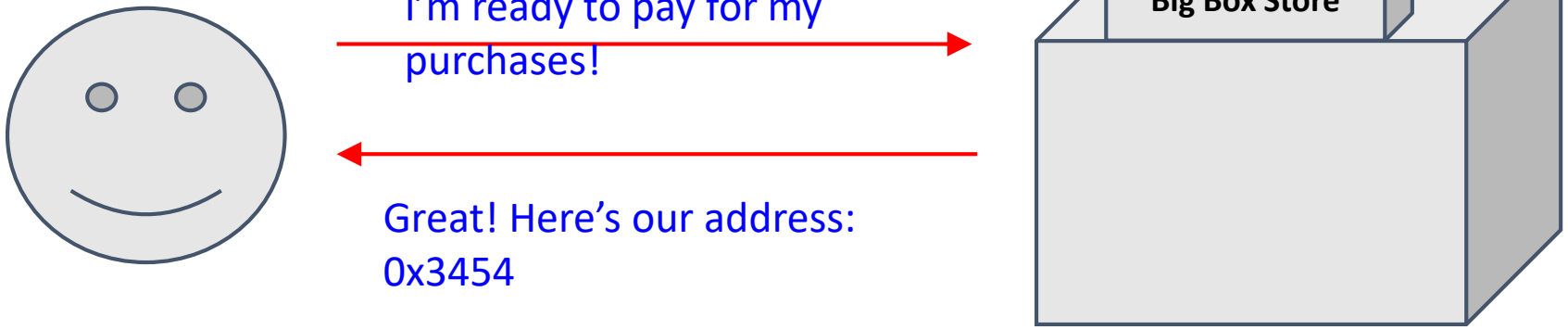
1. Bob
   - creates a redeem script with whatever script he wants
   - hashes the redeem script
   - sends redeem script hash to Alice.
2. Alice
   - creates a P2SH-style output containing Bob's redeem script hash.
3. When Bob wants to spend the output
   - provides his signature along with the redeem script in the signature script.
4. P2P network then
   - ensures the redeem script hashes to the same value as the script hash Alice put in her output;
   - it then processes the redeem script exactly as it would if it were the primary pubkey script,
   - letting Bob spend the output if the redeem script does not return false.

☑ What is this about? What is the usage?

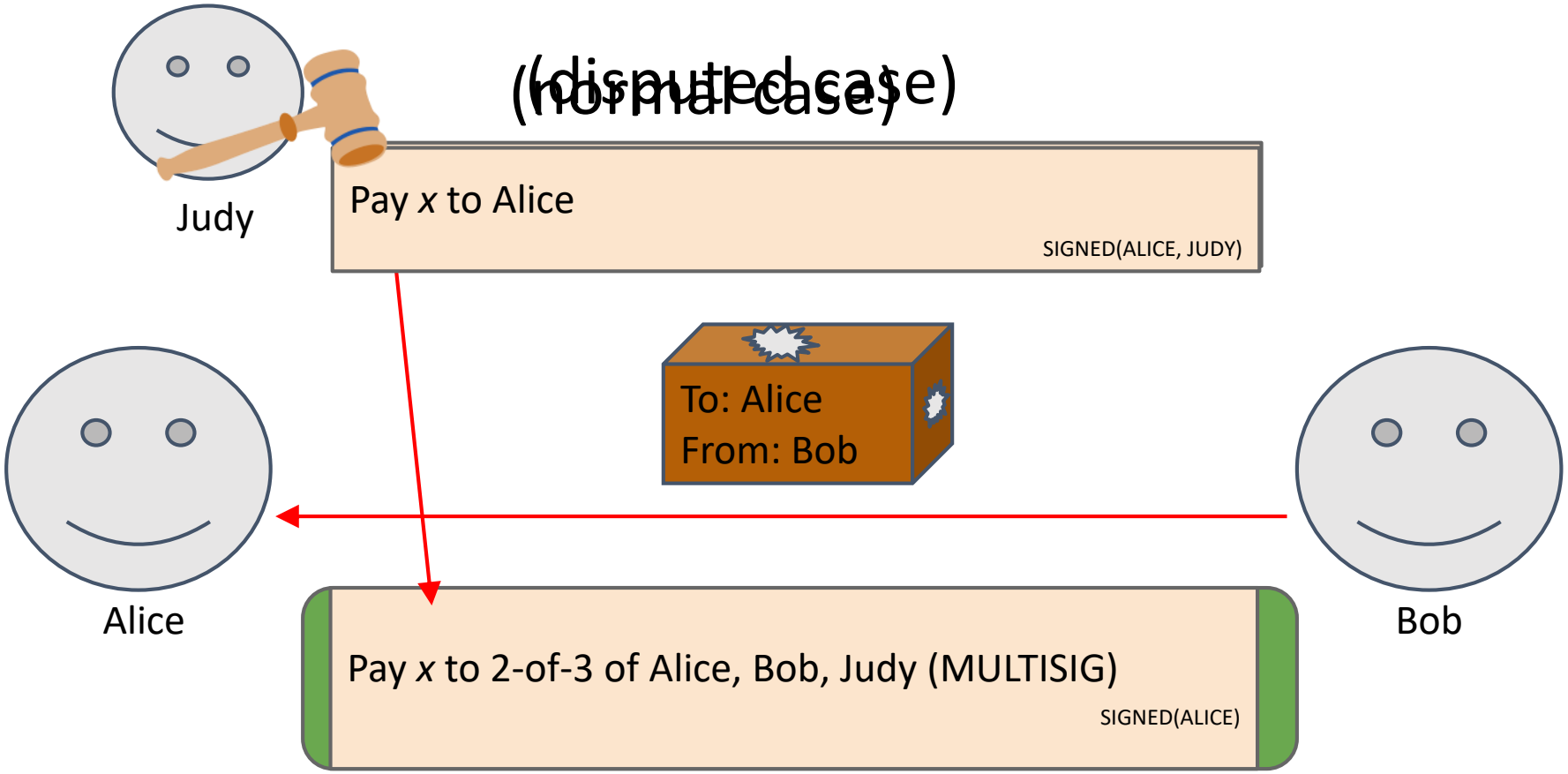
Pay-to-Script-Hash

Alice

Bob

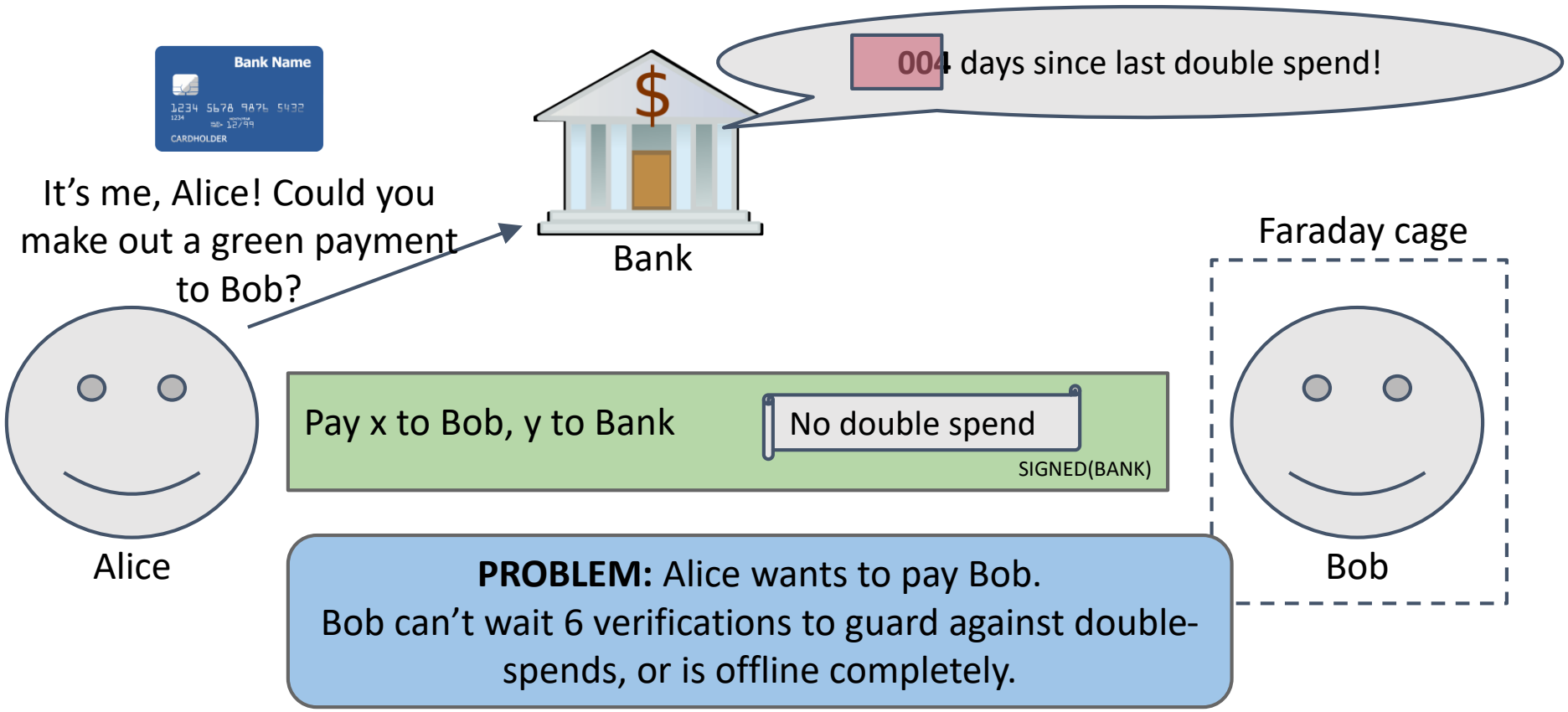


Example 1: Escrow Transactions

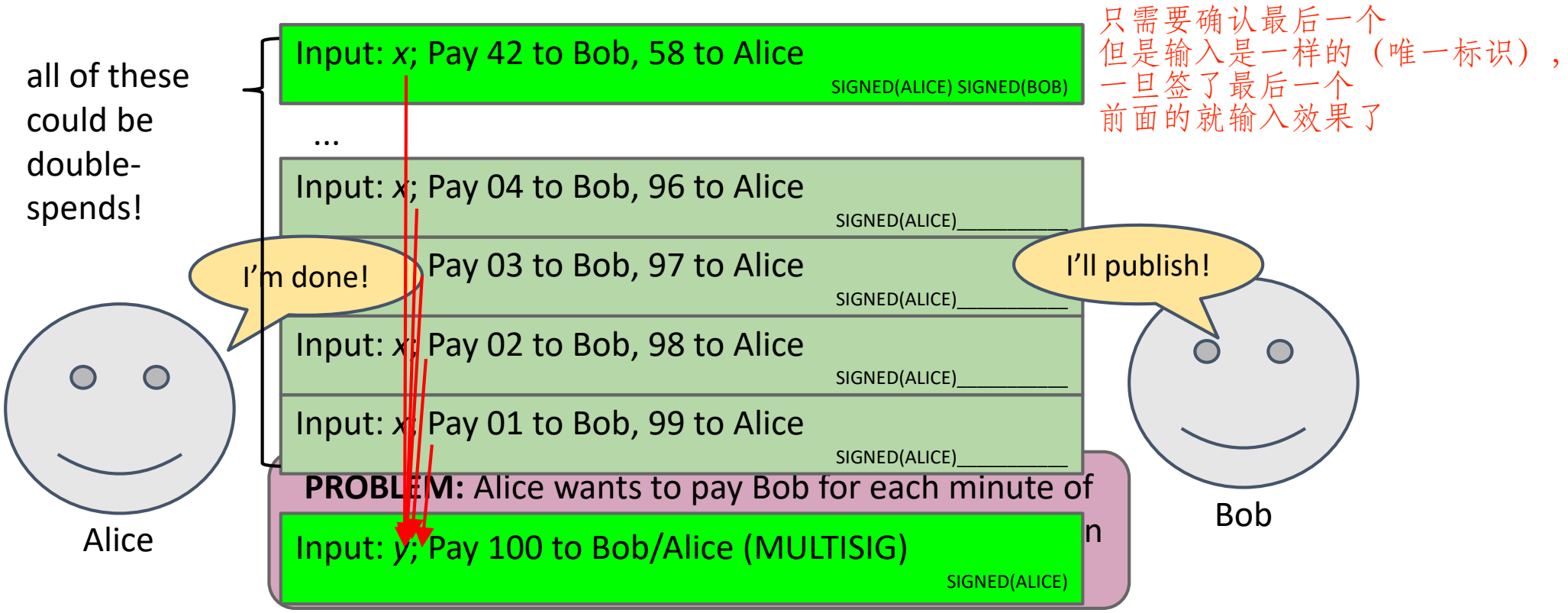


发现 Bob 不诚实给 Alice 了假货
那么 Judy 就会做出裁判。
但关键是怎么找到 Judy 这一个公正的裁判
无非就是构造一个公开的裁决系统，让大众自己判断

Example 2: Green Addresses



Example 3: Efficient Micro-Payments



Example 3: Efficient Micro-Payments

What if Bob never signs??

Input: *x*; Pay 42 to Bob, 58 to Alice

SIGNED(ALICE)_____

Alice demands a timed refund transaction before starting 有限制时间，时间内不完成就作废

Input: *x*; Pay 100 to Alice, LOCK until time *t*

SIGNED(ALICE) SIGNED(BOB)

Alice

Bob

Input: *y*; Pay 100 to Bob/Alice (MULTISIG)

SIGNED(ALICE)

只有共同都签名的时候才生效

lock_time


```
{
        "hash":"5a42590...b8b6b",
         "ver":1,
         "vin_sz":2,
         "vout_sz":1,
         "lock_time":315415,
         "size":404,
...
}
```


Block index or real-world timestamp before which this transaction can't be published

More advanced Scripts

Multiplayer Lotteries

Coin-swapping Protocols

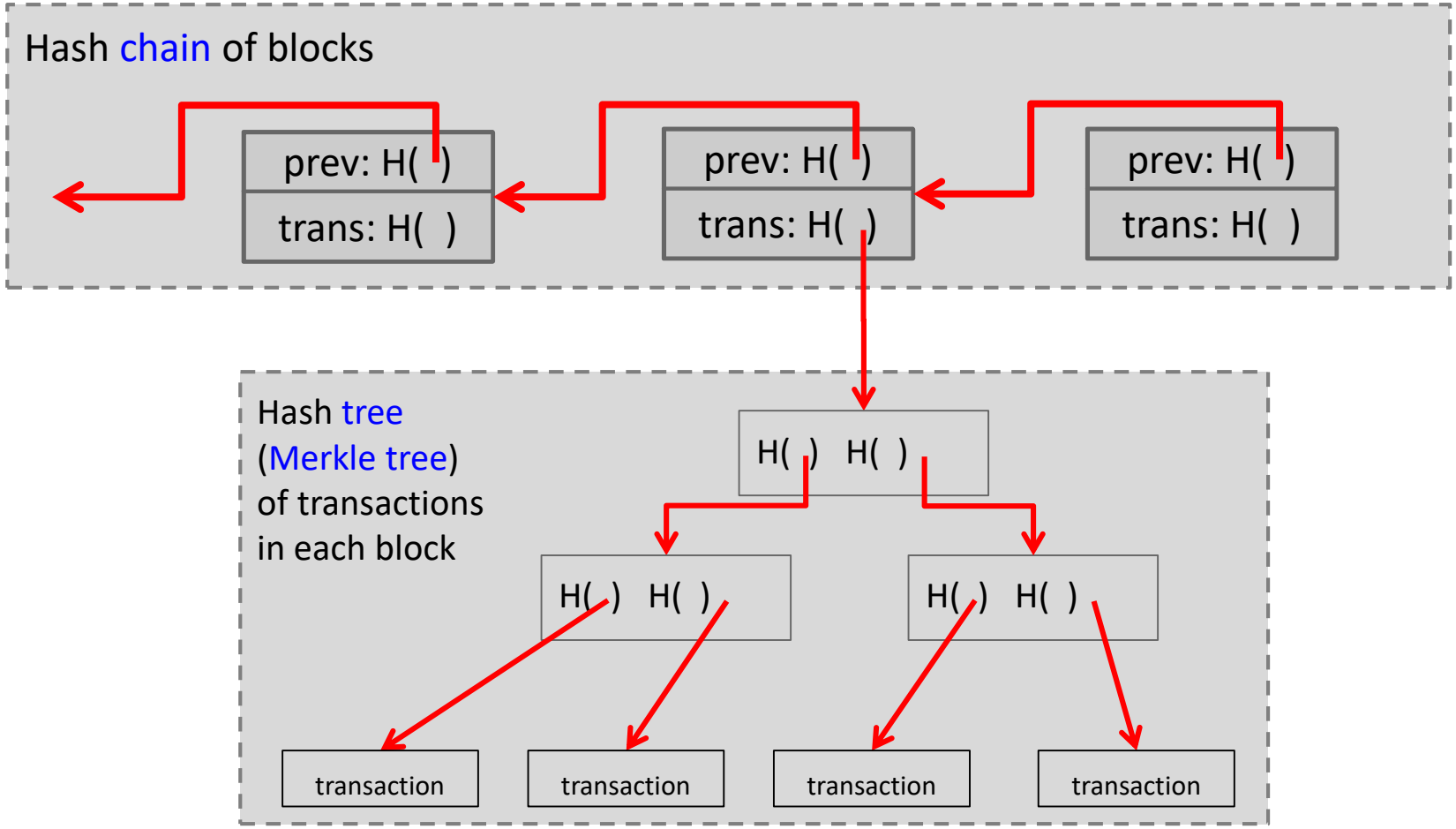
“Smart Contracts”

Bitcoin Blocks

Q: Why bundle transactions together?

1. Requiring consensus for each transaction separately would reduce transaction acceptance rate.
2. Hash-chain of blocks is much shorter.
3. Faster to verify history.

Bitcoin Block Structure



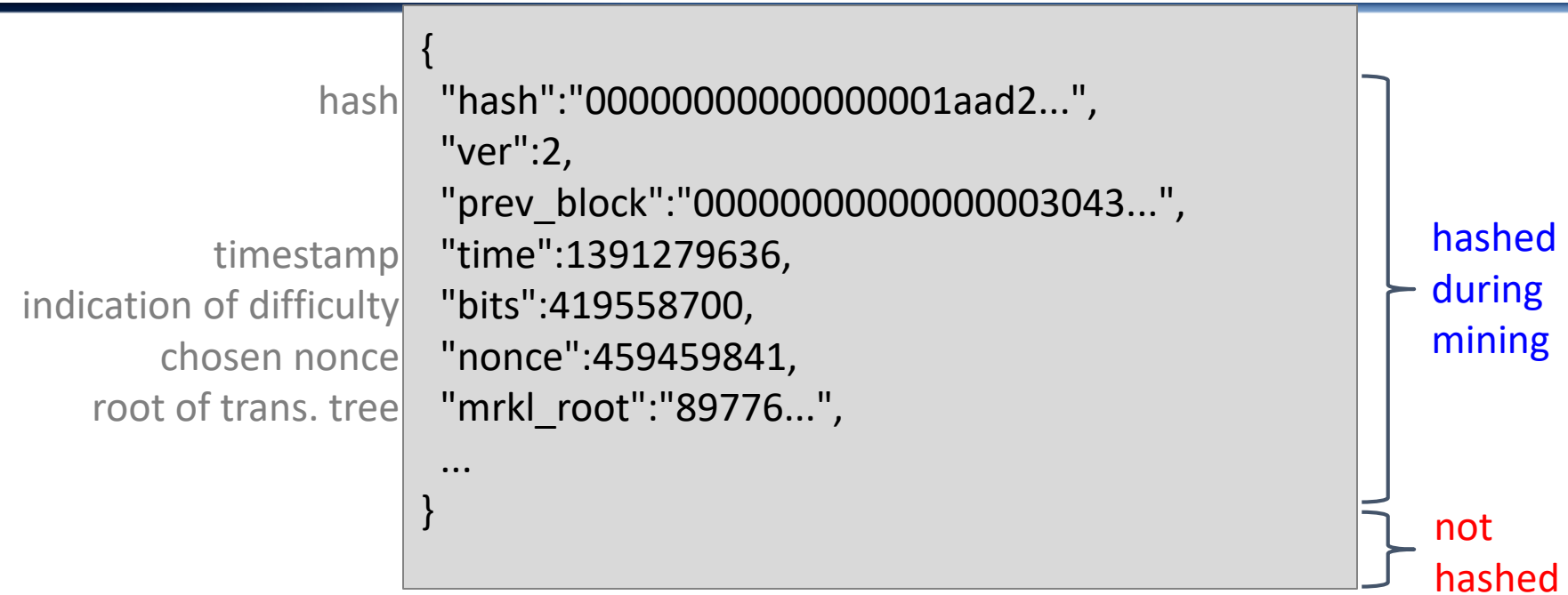
The Real Deal: a Bitcoin Block

block header

transaction data


```
{
 "hash":"00000000000000001aad2...",
 "ver":2,
 "prev_block":"00000000000000003043...",
 "time":1391279636,
 "bits":419558700,
 "nonce":459459841,
 "mrkl_root":"89776...",
 "n_tx":354,
 "size":181520,
 "tx":[
  ...
 ],
 "mrkl_tree":[
  "6bd5eb25...",
  ...
  "89776cdb..."
 ]
}
```

The Real Deal: a Bitcoin Block Header

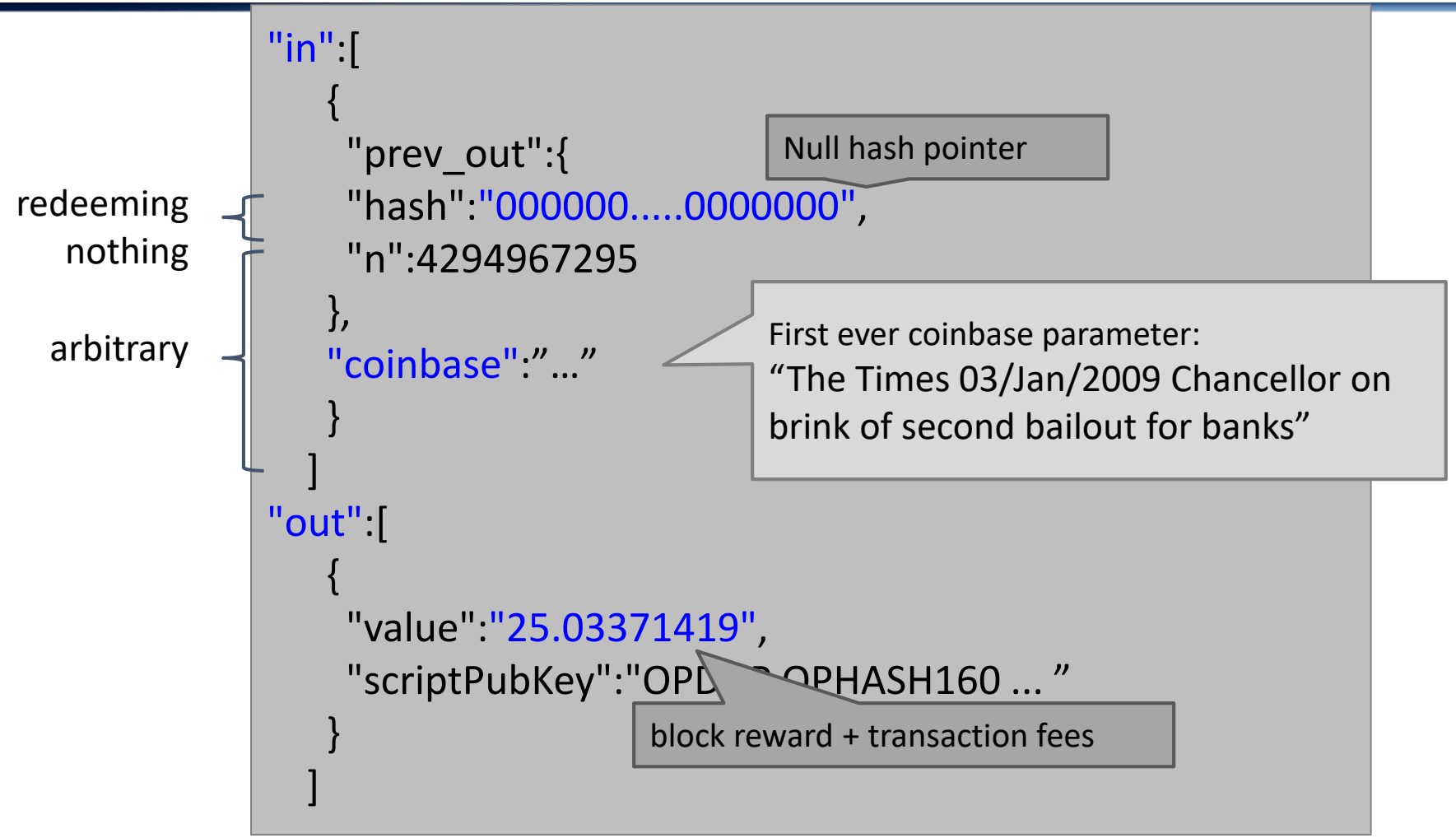


`coinbase` Transaction

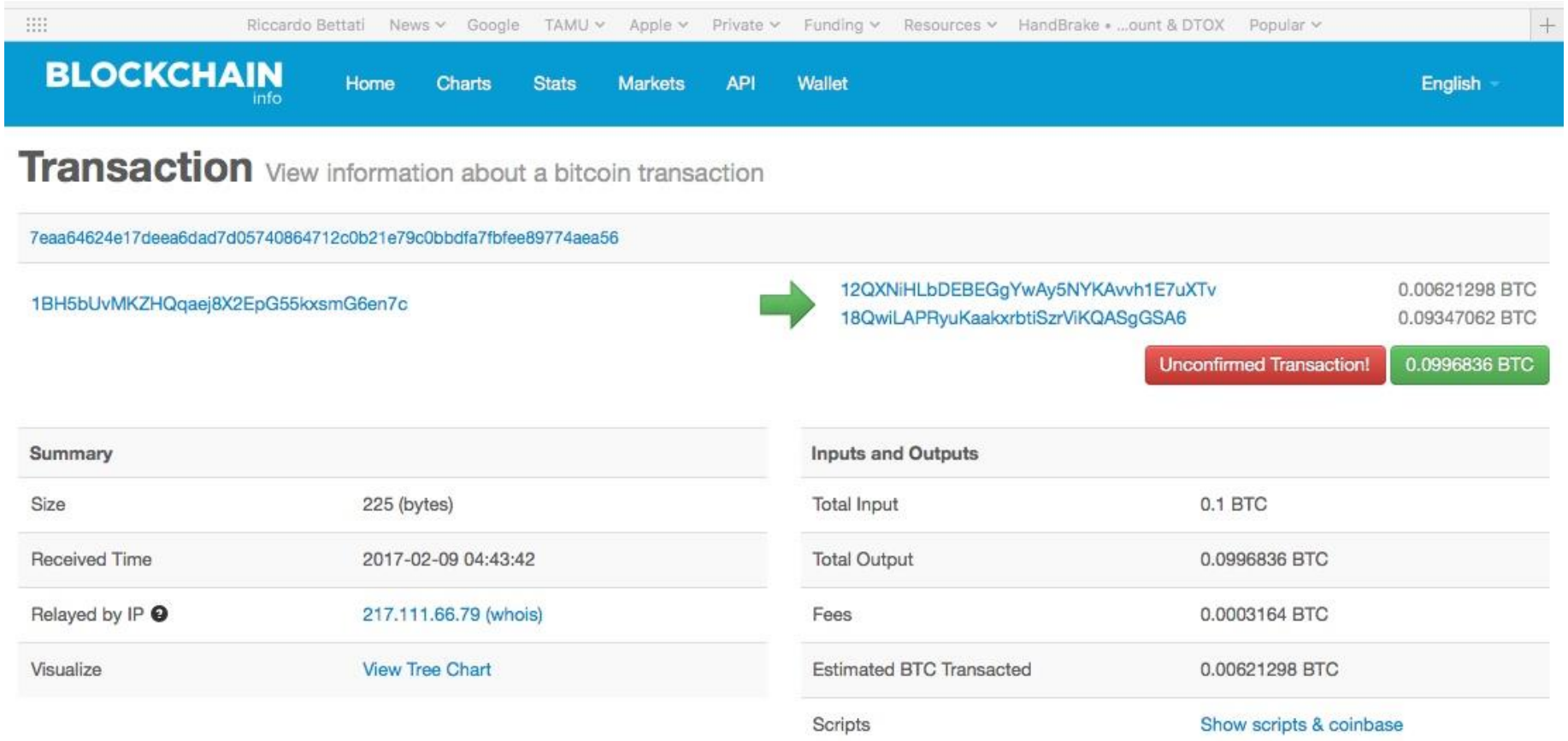
New coins are created with **`coinbase`** transaction:

- Single input field and single output
- Does not redeem previous output
  - Hash pointer is null (indicating actually no inputs at all)
- Output value is miner's revenue from block:
  - *output value = mining reward + transaction fees*
  - transaction fees come from all transactions in block
- Special **`coinbase`** parameter
  - contains arbitrary value

The Real Deal: `coinbase` Transaction



See for yourself!



See for yourself!