区块链智能合约开发

中山大学
数据科学与计算机学院

目录

智能合约及平台简介

智能合约概念

“一个智能合约是一套以数字形式定义的承诺（promises），包括合约参与方可以在上面执行这些承诺的协议。”

——尼克·萨博，1993

- 事件驱动
- 自动执行
- 价值转移
- 中心化，出错难以追溯

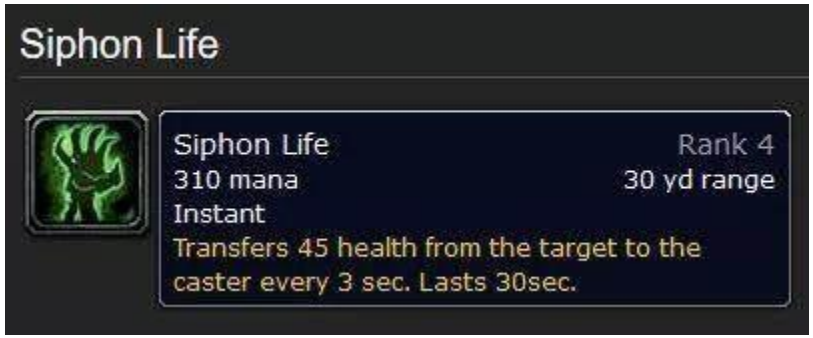
  大额交易不可靠

智能合约及平台简介

中心化程序

智能合约及平台简介

2010年，16岁的V神愤怒的删除了《魔兽世界》客户端，因为暴雪在3.10补丁中移除了术士的技能“生命虹吸”。



对此， V神在暴雪官方论坛提出抗议，没有收获任何官方答复。他意识到了网络游戏“中心化管理”的弊端：游戏的拥有者是暴雪，他们可以不问玩家意见，随意修改游戏内容。

他决定放弃这款游戏，尽管他已经在他的术士身上花费了3年心血

智能合约及平台简介

中心化程序

Vitalik Buterin： I happily played World of Warcraft during 2007-2010, but one day Blizzard removed the damage component from my beloved warlock's Siphon Life spell. I cried myself to sleep, and on that day I realized what horrors centralized services can bring. I soon decided to quit.

——https://about.me/vitalik_buterin

智能合约及平台简介

维塔利克·布特林出生在俄罗斯的一个IT家庭，他的父母都在莫斯科的一所大学中从事计算机科研工作，

4岁那年，父亲送给他了人生中的第一台计算机，从此以后，他最喜欢的玩具就变成了Excel

智能合约及平台简介

升上三年级之后，被学校分到了专为天才儿童准备的尖子班

开始试着自己编写游戏程序了，成品包括一个模拟弹球在房间中运动轨迹的小程序，改造版的《太空侵略者》，以及一款中世纪奇幻主题的战棋游戏。

2007年的时候，他和身边的很多同龄人一样，沉迷于暴雪出品的网络游戏《魔兽世界》，一玩就是一整天。

智能合约及平台简介

读高中的维塔利克开始给一个以比特币为主题的网站写博文日志赚稿费，每篇稿酬是5比特币，在当时的价值是不到4美元。

站上署名“维塔利克·布特林”的文章越来越多。他开始在文章的末尾加上自己的比特币钱包地址，并声称一旦收到足够的捐款，他就会继续写下一篇文章。这让他收获大量的比特币，虽然当时的比特币还不怎么值钱。

智能合约及平台简介

创办了行业杂志《Bitcoin Magazine》（比特币杂志）

智能合约及平台简介

到滑铁卢大学读书，但过了半年多，他发现自己没办法兼顾学业与比特币的研究。维塔利克选择了一条很多IT领袖走过的老路：辍学

财富自由：2013年的比特币的价值，已经从当时的不到1美元猛增到了近1000美元一枚

意识到了比特币在先天的设计上就有一些无法突破的局限性。他曾在官方论坛与比特币高层管理人员提出改进建议，但和《魔兽世界》经历过的事情一样，他的建议没有收到任何答复

智能合约及平台简介

2013年公布了他的宏伟计划，并发表初版《以太坊白皮书》，这时候的他年仅19岁。

在白皮书中，他肯定了比特币网络“分布式数据库”的伟大理念，但也指出它的缺点：扩展性不足，只有比特币一种符号。而以太坊则可以视为一台分布式的电脑，任何人都可以在以太坊上传和执行应用程序，矿工们就像是负责计算的CPU，共同组成一个去中心化的世界计算机。

智能合约及平台简介

2017年3月开始，以太币价值忽然一路高歌猛进，从不到10美元的价格一路飙升至6月的250美元。

智能合约及平台简介

2017年6月，俄罗斯总统普京本人在圣彼得堡的国际经济论坛上亲自接见了这位出生在俄罗斯的年轻人。

智能合约及平台简介

比特币中的脚本（非图灵完备）

Input Scripts

ScriptSig: PUSHDATA(22)[0014a333007260cfa6a8225d6e706b1a43b3524aa355]
Witness:
02483045022100e56760947227c0465bdc545bff1ed496d759774b15d529f5bdeb40c631225e1c0220701b1df1b7bdec2f29a1a76cdaa5ff863f01e045d7f058562840248d94a6bec601210

ScriptSig: PUSHDATA(22)[0014133a03f26c34abc4efbc54002257bf5e669a3834]
Witness:
02483045022100a1de6d5746f4b4043fbff1eefb4d19c173c242611b937b9e1e38f29063768882022019e4605b0b20b87babf0773c44a2d10c4ad9810a96451b0b2658a0cf80f4d3070121

Output Scripts

DUP HASH160 PUSHDATA(20)[2d657da999468d123f770b736cbe080e7d070551] EQUALVERIFY CHECKSIG

HASH160 PUSHDATA(20)[09232271e313d3308e5056b329f54719e8580c72] EQUAL

非图灵完备　　**对比特币进行改良？**

智能合约及平台简介

以太坊（Ethereum）

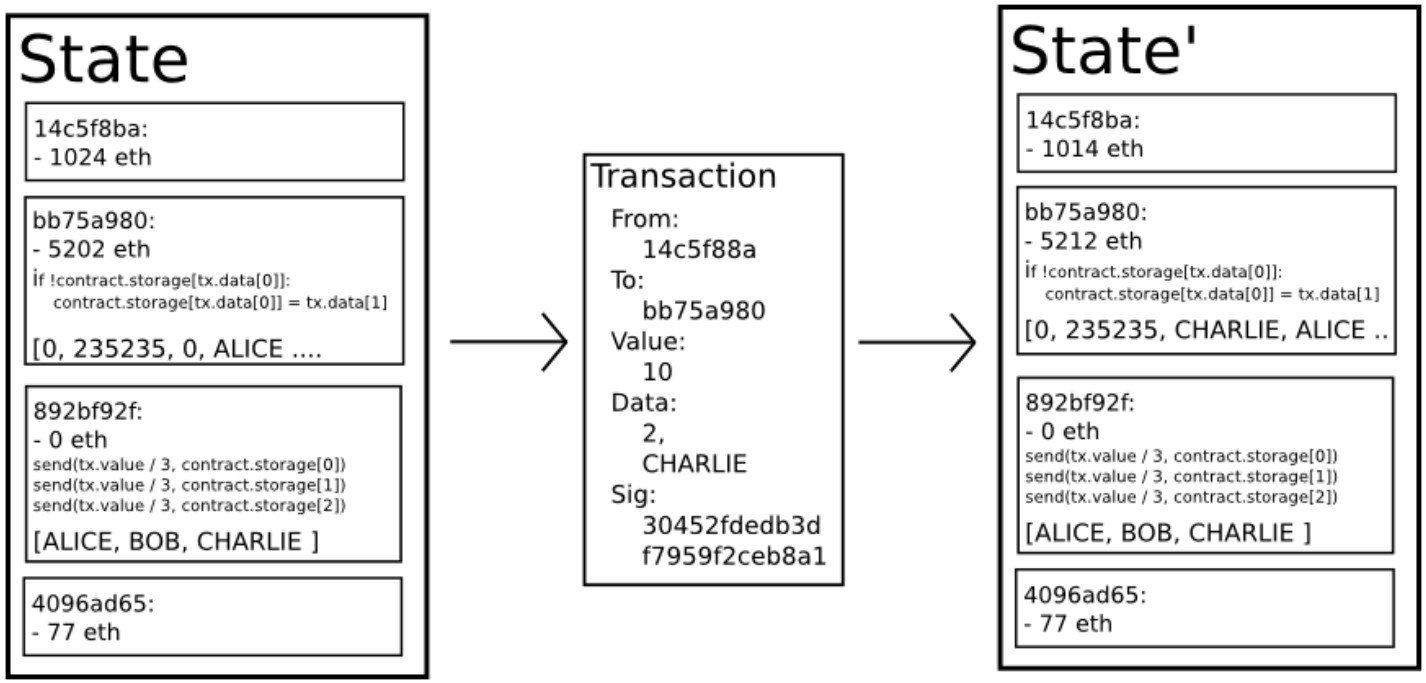
- 开源的
- 分布式的
- 基于区块链的智能合约平台
- 无需审查及第三方干扰
- 智能合约引擎EVM

  Ethereum Virtual Machine

智能合约及平台简介

以太坊（Ethereum）

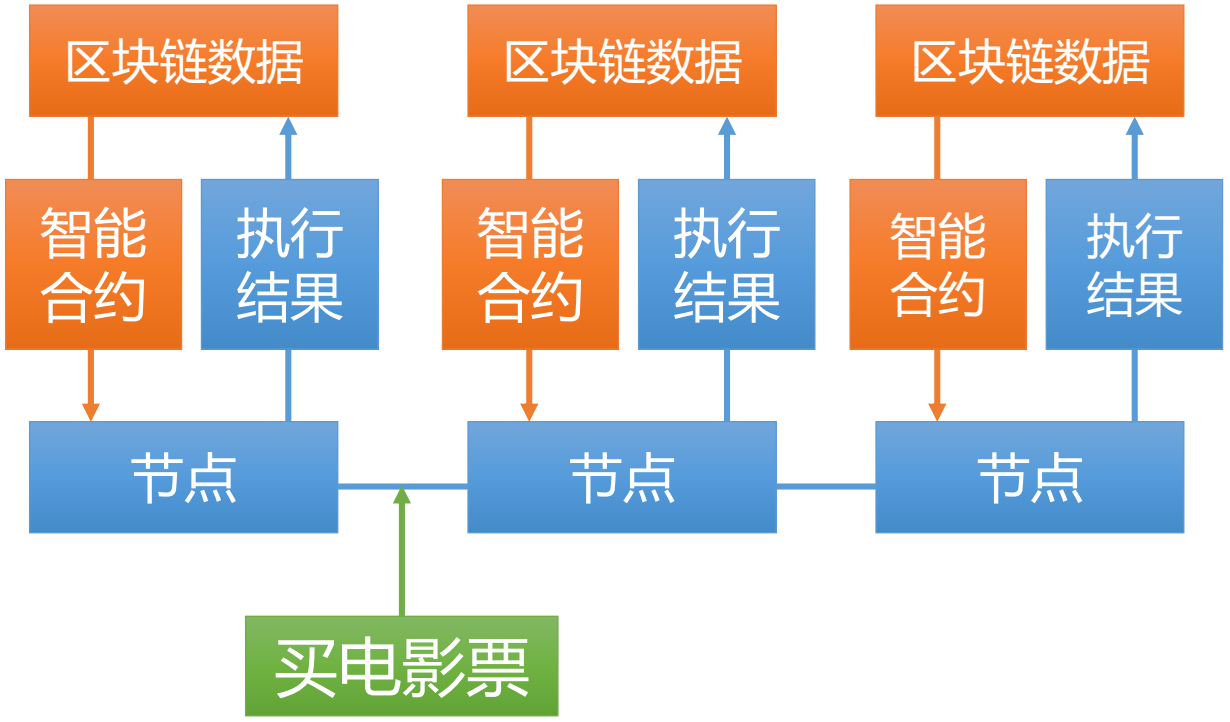
- 重要思路：通过区块链交易进行系统的状态转换。



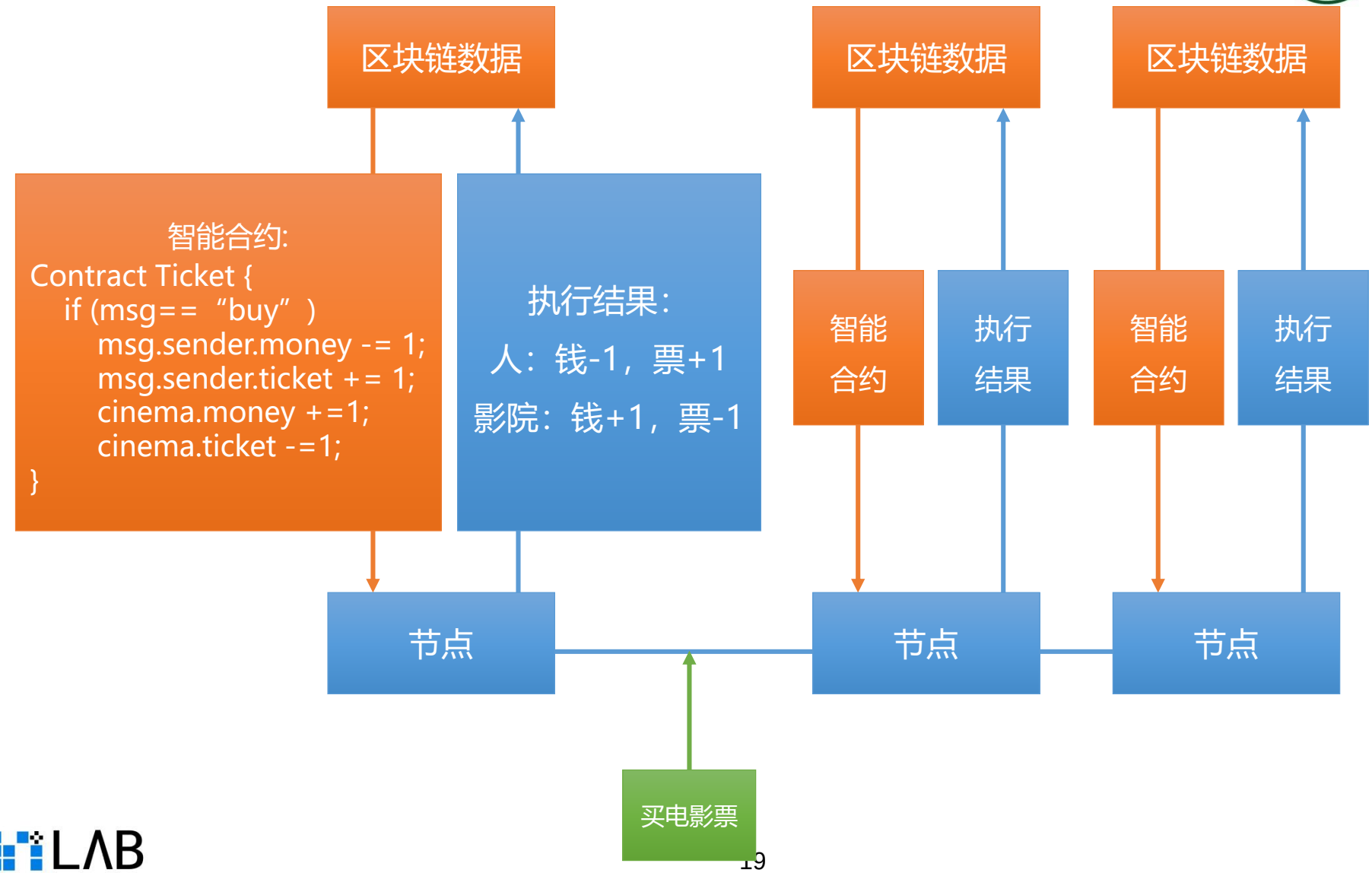
智能合约及平台简介

区块链智能合约

- 去中心化执行
- 不可篡改
- 可回溯



智能合约及平台简介


区块链数据
区块链数据
区块链数据
智能合约:
Contract Ticket {
if (msg== “buy”)
msg.sender.money -= 1;
msg.sender.ticket += 1;
cinema.money +=1;
cinema.ticket -=1;
}
执行结果：
人：钱-1，票+1
影院：钱+1，票-1
智能合约
执行结果
智能合约
执行结果
节点
节点
节点
买电影票

智能合约及平台简介

区块链智能合约平台

	Application	Smart contract execution	Smart contract language	Data model	Consensus
Hyperledger	Smart contract	Dockers	Golang, Java	Account-based	PBFT
Ethereum	Smart contract, Crypto-currency	EVM	Solidity, Serpent, LLL	Account-based	Ethash (PoW)
Eris-DB	Smart contract	EVM	Solidity	Account-based	Tendermint (BFT)
Ripple	Crypto-currency	-	-	UTXO-based	Ripple Consensus Ledger (PoS)
ScalableBFT	Smart contract	Haskell Execution	Pact	Account-based	ScalableBFT
Stellar	Smart contract	Dockers	JavaScript, Golang, Java, Ruby, Python, C#	Account-based	Stellar Consensus Protocol
Dfinity	Smart contract	EVM	Solidity, Serpent, LLL	Account-based	Blockchain Nervous System
Parity	Smart contract	EVM	Solidity, Serpent, LLL	Account-based	Proof of Authority
Tezos	Smart contract, Crypto-currency	Dockers	Tezos Contract Script Language	Account-based	Proof of Stake
Corda	Smart contract	JVM	Kotlin, Java	UTXO-based	Raft
Sawtooth Lake	Smart contract	TEE	Python	Account-based	Proof of Elapsed Time

(Tien Tuan Anh Dinh, Ji Wang, Gang Chen, Rui Liu, Beng Chin Ooi, and Kian-Lee Tan. 2017. BLOCKBENCH: A Framework for Analyzing Private Blockchains. In *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM, 1085–1100)

智能合约及平台简介

区块链智能合约平台

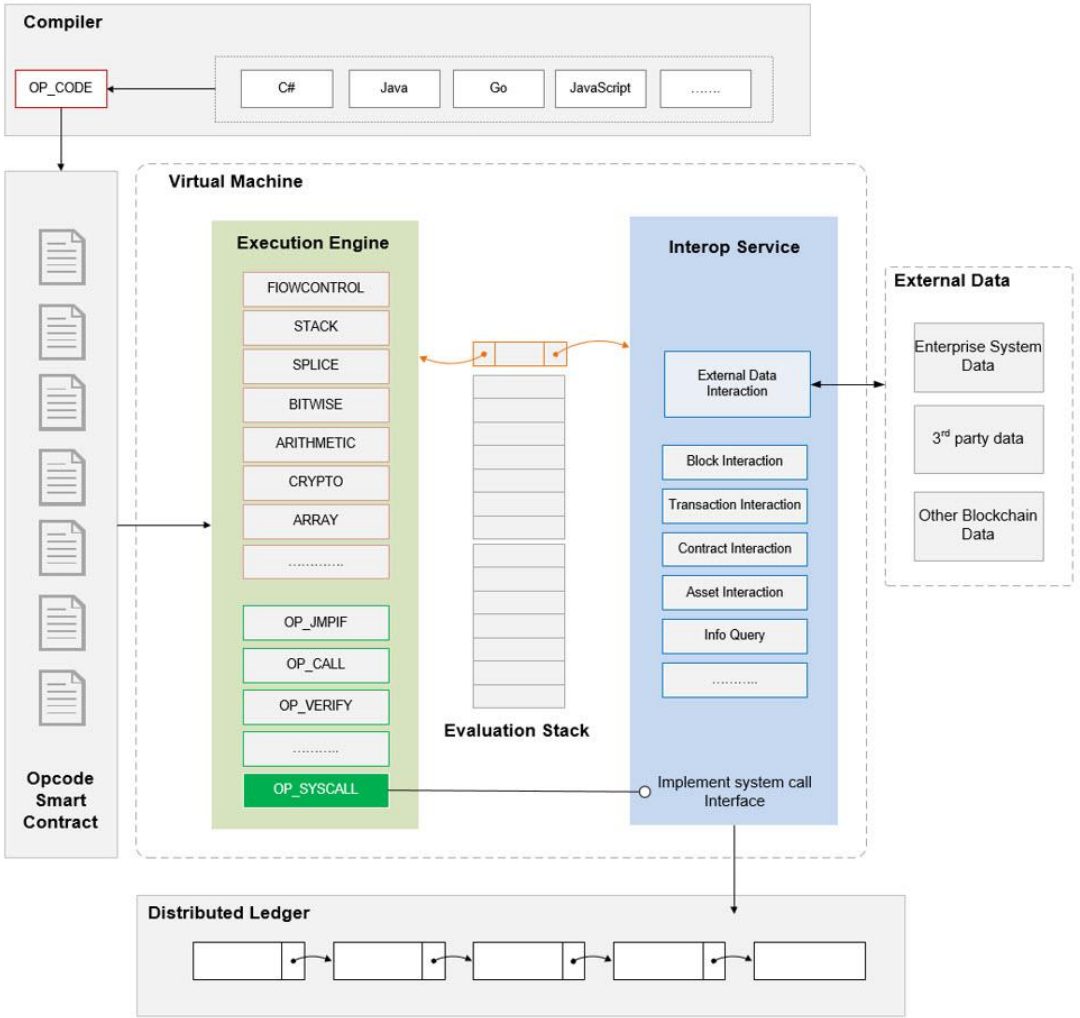
- 比较出名的

Platform	Language	Execution
Ethereum [7]	Solidity, Serpent	EVM
NEO [28]	C#, Java, Python	NeoVM
EOS [21]	C/C++	WASM
Fabric [17]	Java, Go	Docker
Corda [23]	Java, Kotlin	JVM
Hyperchain [25]	Solidity,Java	HyperVM

智能合约及平台简介

NEO

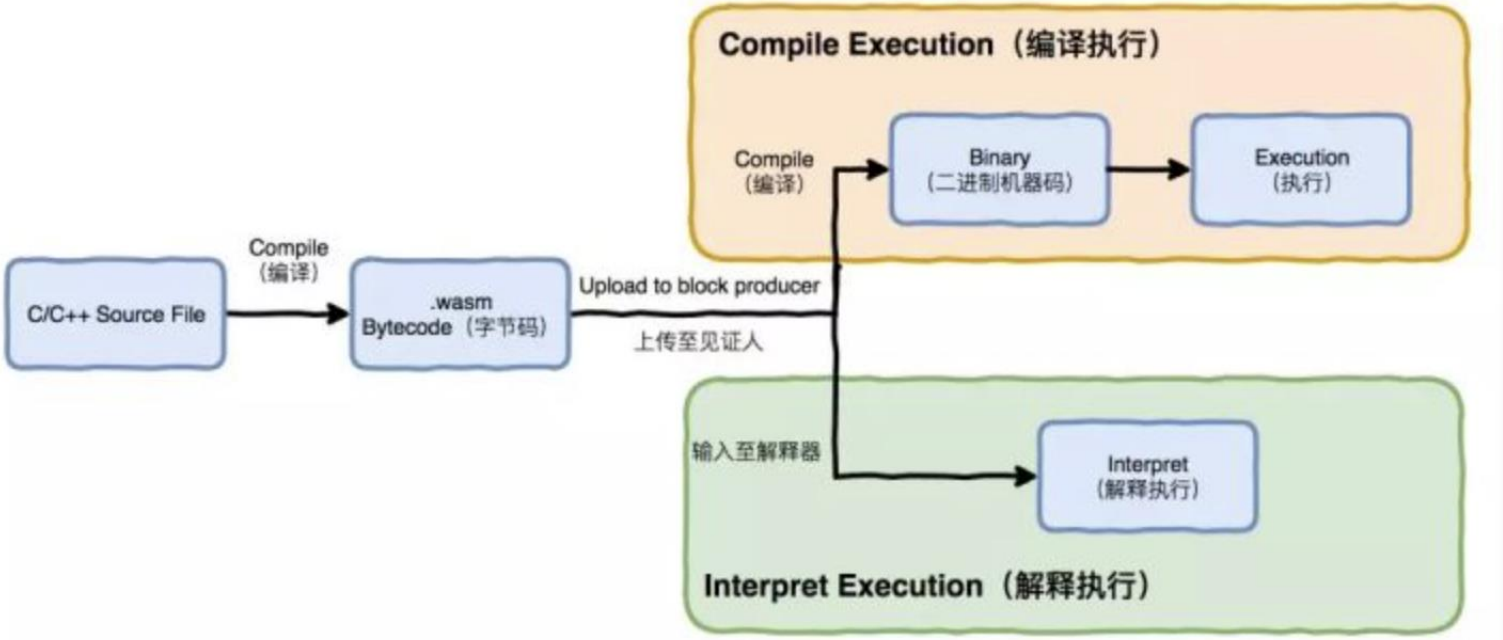
- “中国以太坊”
- dBFT共识机制
- 合约运行在NEO VM
- 智能合约编写支持 C#、VB.Net、F#、Java、Kotlin、Python



智能合约及平台简介

EOS

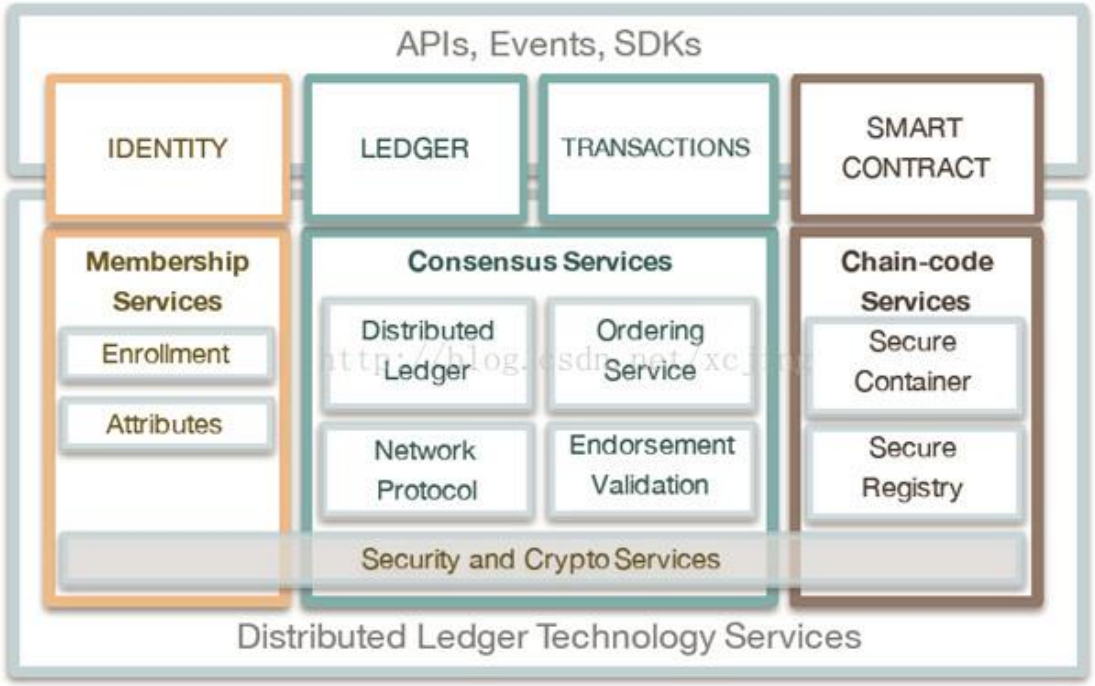
- DPoS共识机制
- 开发者抵押代币获取合约运行资源，用户免费
- 合约通过C++等编写，支持运行在WASM或EVM



智能合约及平台简介

Hyperledger Fabric

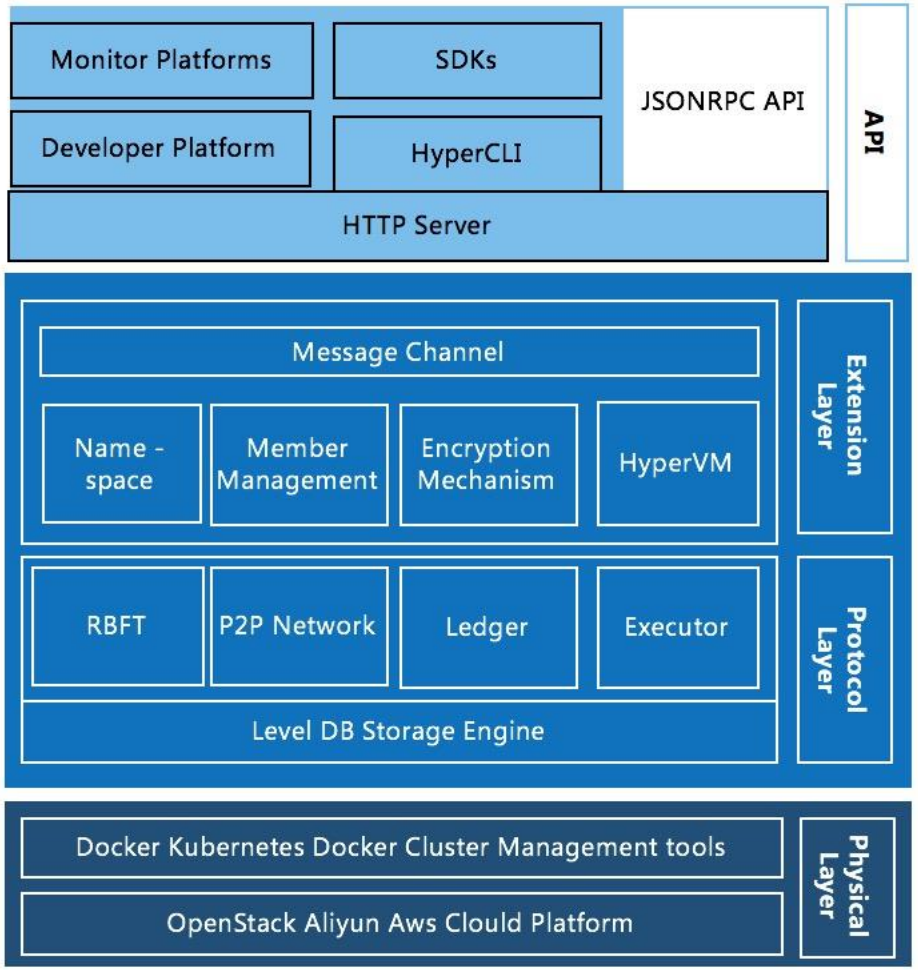
- 联盟链
- 可进行成员管理
- 智能合约称为chaincode
- 合约运行在docker中
- 可采用Go、Java、Node.js语言编写



智能合约及平台简介

Hyperchain

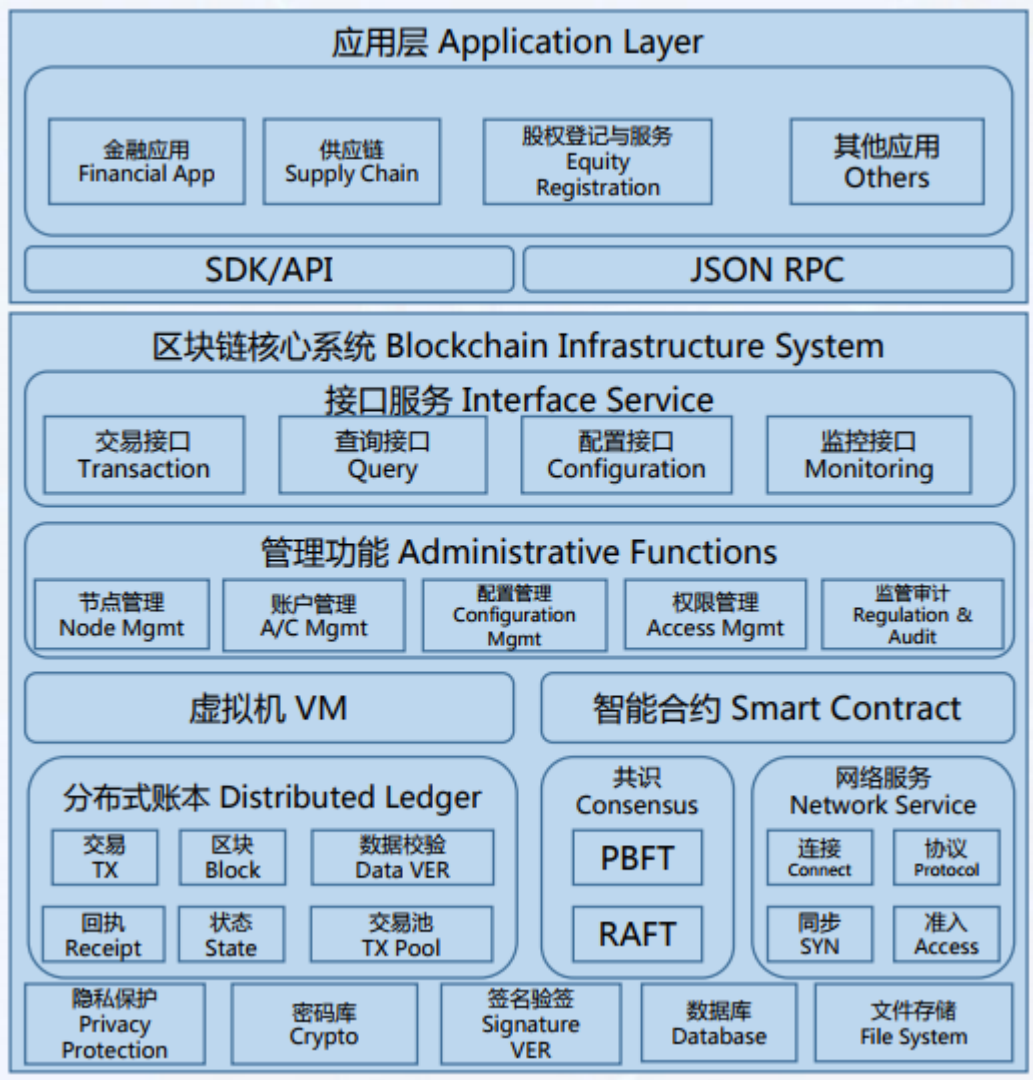
- 联盟链
- 可进行成员管理
- 共识机制rBFT
- 合约运行在HyperVM
- 支持Solidity Java语言



智能合约及平台简介

FISCO-BCOS

- 联盟链
- 微众银行背景
- 可进行成员管理
- 共识机制PBFT RAFT
- 合约运行在VM
- 支持Solidity语言



智能合约及平台简介

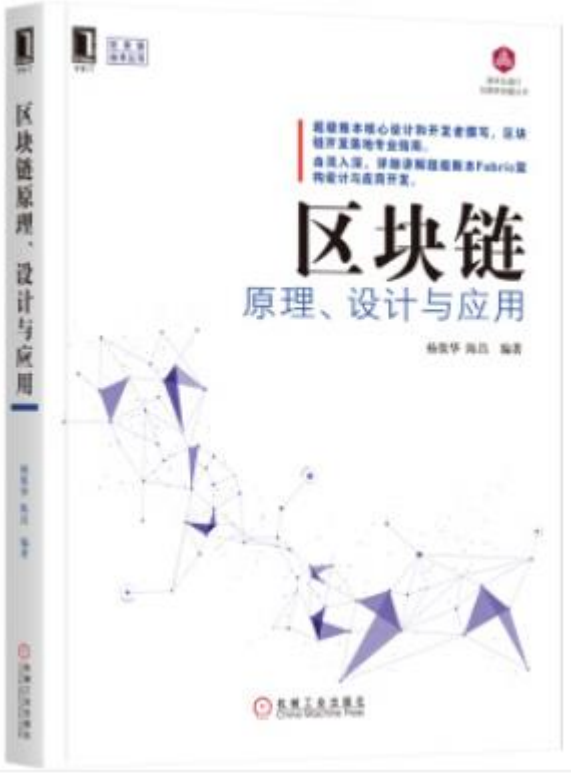
参考网站

- 杨保华：https://github.com/yeasy/blockchain_guide 多人共同维护的入门手册
- 谈国鹏：http://www.8btc.com/author/16692 在实战中附加许多基础知识的讲解
- 李赫：http://blog.csdn.net/sportshark/article/list/3 既有搭私链实战，又有代码分析
- Ethereum (ETH) Blockchain Explorer https://etherscan.io/
- EthFans | 以太坊爱好者 https://ethfans.org/
- The ethereum node explorer https://ethernodes.org/network/1
- State of the Đapps https://www.stateofthedapps.com/

智能合约及平台简介

参考书籍

公有链为主



联盟链为主

智能合约及平台简介

推荐视频

https://pan.baidu.com/s/1o8qIk10#list/path=%2F

HyperLedger Fabric 0.6：

谈谈区块链（03）：hyperledger入门 --〉难度：**

谈谈区块链（04）：hyperledger初级开发指南 --〉难度：***

以太坊：

谈谈区块链（05）：深入浅出以太坊 --〉难度：***

谈谈区块链（08）：以太坊分叉和以太币买卖 --〉难度：**

谈谈区块链（09）：以太坊智能合约入门 --〉难度：***

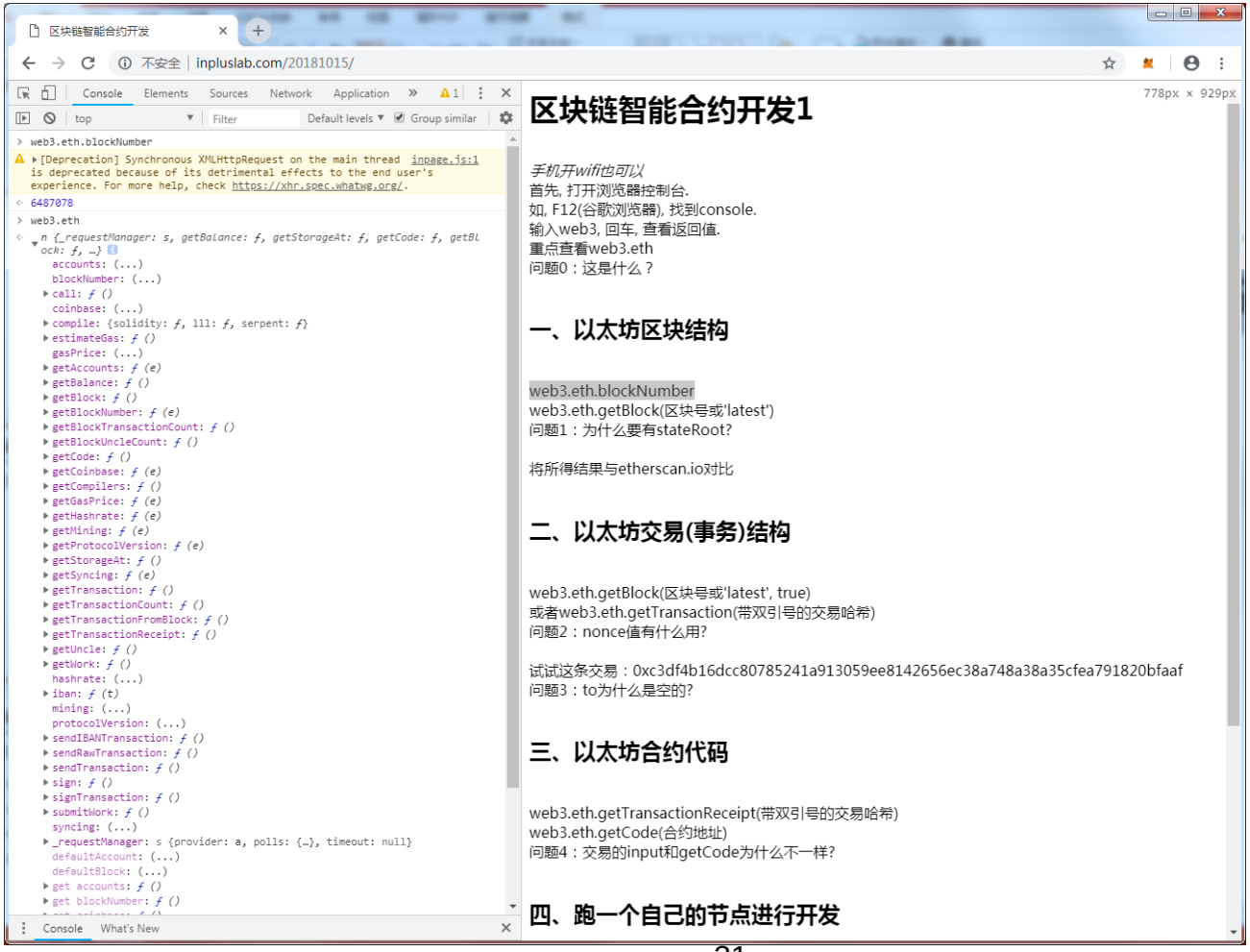
谈谈区块链（10）：以太坊中的Events和Logs --〉难度：****

谈谈区块链（11）：以太坊web3.js详解 --〉难度：***

目录

以太坊基本操作及原理

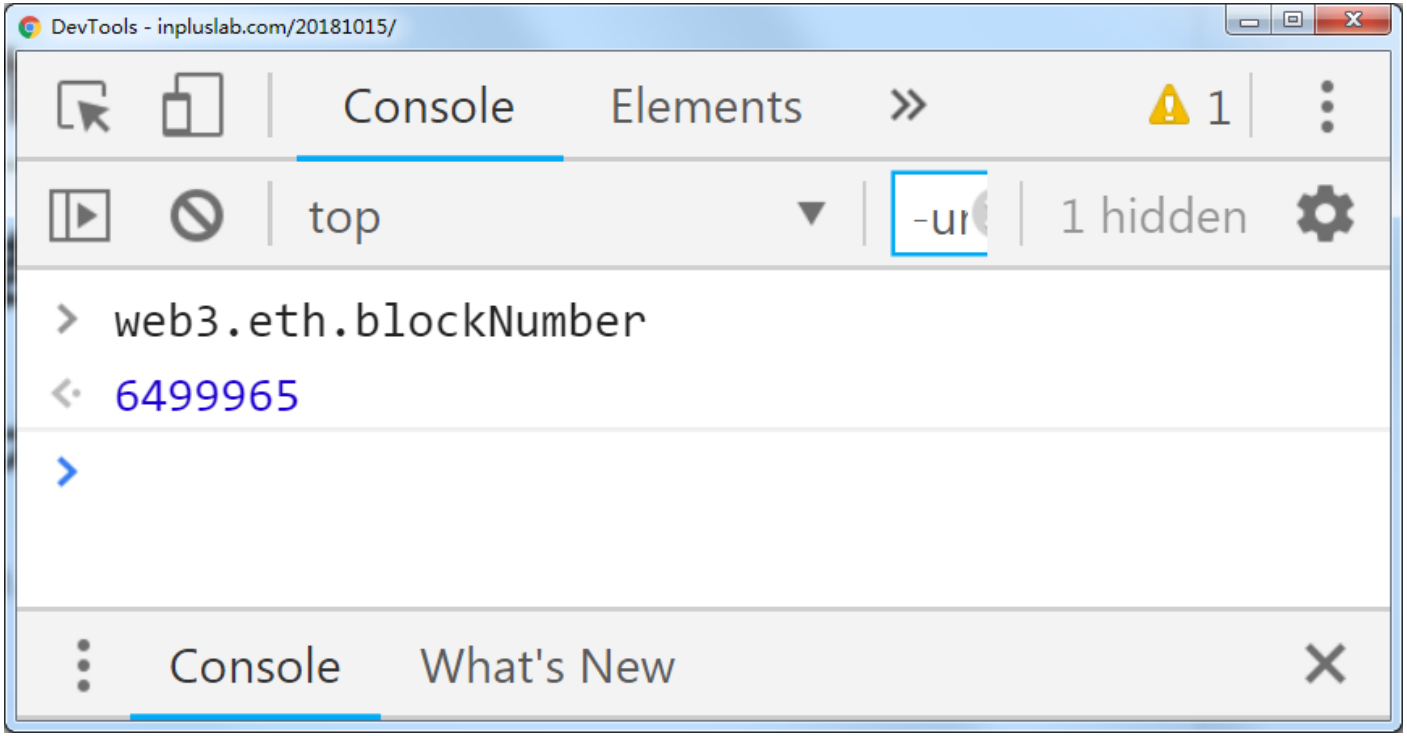
- 打开 inpluslab.com/20181015/



以太坊基本操作及原理

以太坊区块结构

获取当前以太坊区块号



以太坊基本操作及原理

以太坊区块结构

获取最新区块

```
> web3.eth.getBlock(6499965)  > web3.eth.getBlock('latest')
{difficulty: r, extraData: "0x7777772e4257506f6f6c2e6e6574", gasLimit: 8011572, gasUsed: 685210, hash: "
54c6c8a79e49cbf196224502acd71dba5648822326a5341638664825534", …}
  difficulty: r {s: 1, e: 15, c: Array(2)}
  extraData: "0x7777772e4257506f6f6c2e6e6574"
  gasLimit: 8011572
  gasUsed: 685210
  hash: "0xf32c054c6c8a79e49cbf196224502acd71dba5648822326a5341638664825534"
  logsBloom: "0x00000000000400000000004000000000008000000000000010000400002600000000000000000000000000
  miner: "0x52e44f279f4203dcf680395379e5f9990a69f13c"
  mixHash: "0xe9a24f985bb4dce71dd5f8aec4f5713151ea141480b767576a048a5c69ec9ef0"
  nonce: "0x6a77a5b80954a13f"
  number: 6499965
  parentHash: "0x99666773916ece3fc0a1798865263b116a3a05906c882fae94ce2af63ef4f75f"
  receiptsRoot: "0xeb19a5c42be2c748abdb4c94b876657465a847bf8149d80bcd30ca0825959fff"
  sha3Uncles: "0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347"
  size: 2775
  stateRoot: "0x1b70fa0a38fd488cb758769d5c4f0b4280e01fd80eb0aed3dd9891a9b177c673"
  timestamp: 1539329692
  totalDifficulty: r {s: 1, e: 21, c: Array(2)}
  transactions: (17) ["0x8979af95848237a6daa4e27a5f2ff8f78f72a337ce0a56d35976c6079b513d53", "0x7f6c3dd5b
  transactionsRoot: "0x1bd337f0e01be4084a4bffb6e90333afc929d4a7d22f0770e55c670b48db14d6"
  uncles: []
  __proto__: Object
```

以太坊基本操作及原理

以太坊区块结构

获取最新区块

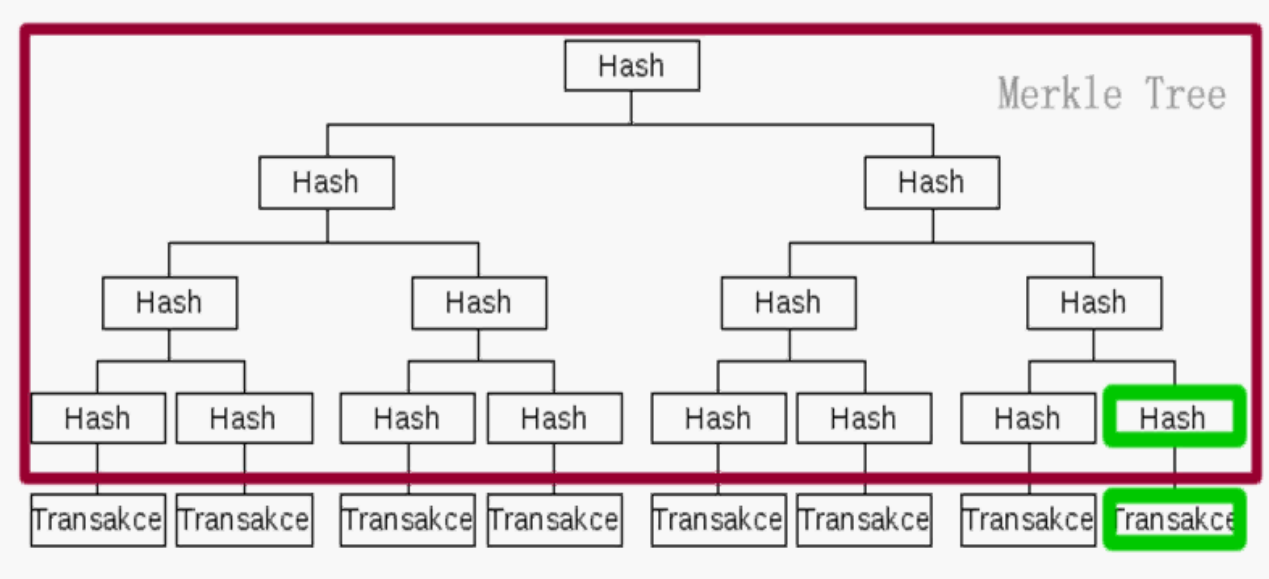
问题：为什么要有stateRoot?

```
> web3.eth.getBlock(6499965) > web3.eth.getBlock('latest')
{difficulty: r, extraData: "0x7777772e4257506f6f6c2e6e6574", gasLimit: 8011572, gasUsed: 685210, hash: "
54c6c8a79e49cbf196224502acd71dba5648822326a5341638664825534", …}
  difficulty: r {s: 1, e: 15, c: Array(2)}
  extraData: "0x7777772e4257506f6f6c2e6e6574"
  gasLimit: 8011572
  gasUsed: 685210
  hash: "0xf32c054c6c8a79e49cbf196224502acd71dba5648822326a5341638664825534"
  logsBloom: "0x00000000000400000000004000000000000000000000000000000100004000026000000000000000000000000
  miner: "0x52e44f279f4203dcf680395379e5f9990a69f13c"
  mixHash: "0xe9a24f985bb4dce71dd5f8aec4f5713151a141480b767576a048a5c69ec9ef0"
  nonce: "0x6a77a5b80954a13f"
  number: 6499965
  parentHash: "0x99666773916ece3fc0a1798865267b116a3a05906c882fae94ce2af63ef4f75f"
  receiptsRoot: "0xeb19a5c42be2c748abdb4c94b876657465a847bf8149d80bcd30ca0825959fff"
  sha3Uncles: "0x1dcc4de8dec75d7aab85b567b6ccd41ad312451b948a7413f0a142fd40d49347"
  size: 2775
  stateRoot: "0x1b70fa0a38fd488cb758769d5c4f0b4280e01fd80eb0aed3dd9891a9b177c673"
  timestamp: 1539329692
  totalDifficulty: r {s: 1, e: 21, c: Array(2)}
  transactions: (17) ["0x8979af95848237a6daa4e27a5f2ff8f78f72a337ce0a56d35976c6079b513d53", "0x7f6c3dd5b
  transactionsRoot: "0x1bd337f0e01be4084a4bffb6e90333afc929d4a7d22f0770e55c670b48db14d6"
  uncles: []
  __proto__: Object
```

以太坊基本操作及原理

以太坊账户结构

比特币中采用Merkle Tree对交易不断哈希获得根部



以太坊基本操作及原理

以太坊账户结构

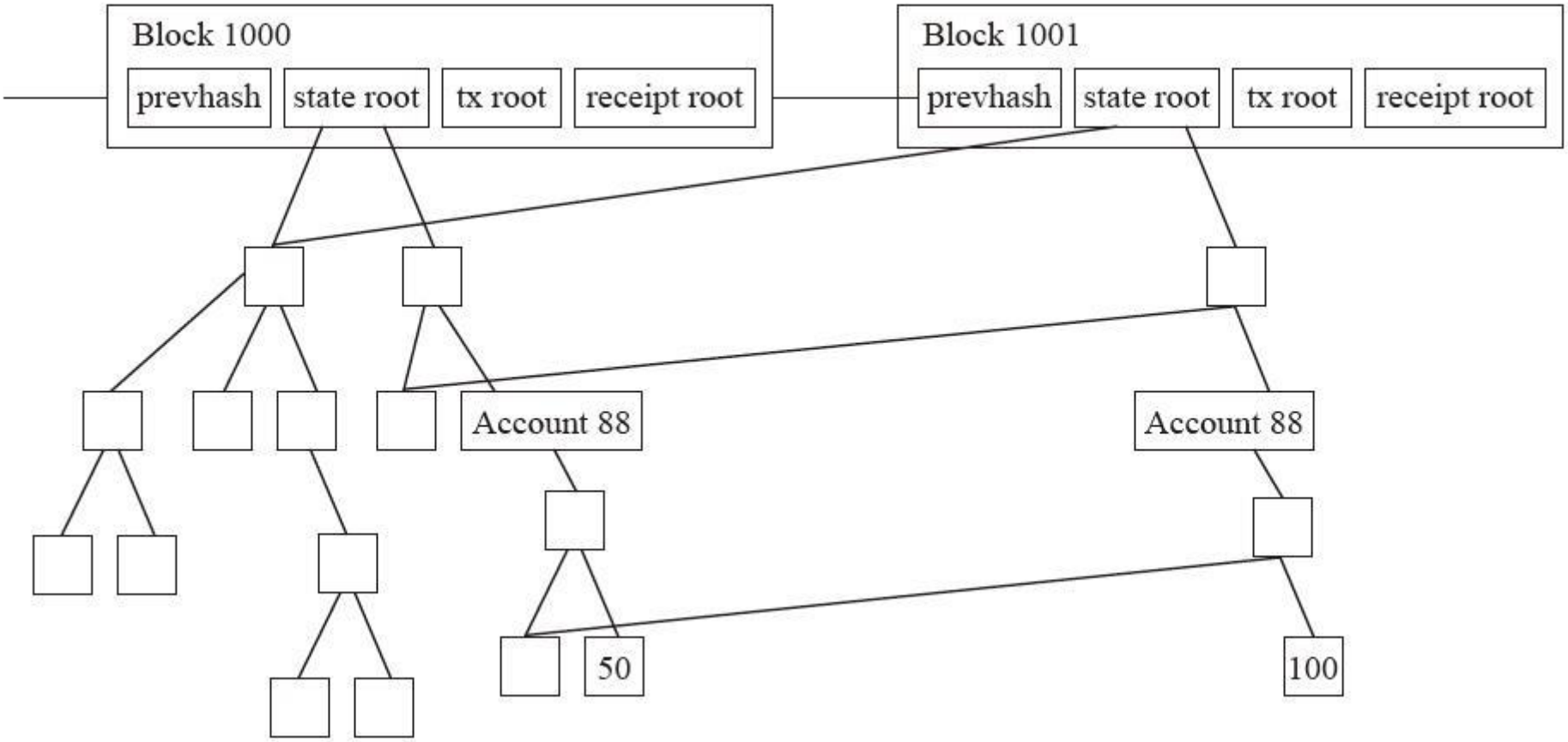
1. **外部账户：**由私钥来控制，是由用户实际控制的账户。每个外部账户拥有一对公私钥，这对密钥用于签署交易，它的地址由公钥决定。外部账户不能包含智能合约代码。
2. **合约账户：**包含合约代码的账户。合约账户不由用户控制，而是由合约代码控制。通过区块链上的交易进行创建、调用、销毁。因此“智能”。

以太坊基本操作及原理

- 以太坊账户结构

以太坊基本操作及原理

以太坊账户结构



以太坊基本操作及原理

以太坊账户结构

- 以太坊采用Merkle Patricia Tree对账户信息哈希
- 人+合约的存储信息->Worldstate
- 延伸概念：Radix Tree、RLP编码、Merkle Patricia Tree

问题：为什么要有stateRoot?

以太坊基本操作及原理

以太坊账户结构

- 以太坊采用Merkle Patricia Tree对账户信息哈希
- 人+合约的存储信息->Worldstate
- 延伸概念：Radix Tree、RLP编码、Merkle Patricia Tree

问题：为什么要有stateRoot?

方便节点间状态的互相验证，保证在交易的每个区块（每时每刻），所有节点的状态是一致的。

以太坊基本操作及原理

以太坊交易(事务)结构

获取最新交易

问题：nonce值有什么用?

```
> web3.eth.getTransaction("0xdb3be044880d3e9f1c04c37c164191ca6d283479677ab0ff82f2
20282269a09d")
<- {blockHash: "0xf671fe1554e45a76a7fe2af8825f3702a486594c58c1a5d01fa5190c4485b4
▼ 61", blockNumber: 6502125, from: "0x5e032243d507c743b061ef021e2ec7fcc6d3ab89"
  , gas: 45000, gasPrice: r, …}
    blockHash: "0xf671fe1554e45a76a7fe2af8825f3702a486594c58c1a5d01fa5190c4485b461"
    blockNumber: 6502125
    from: "0x5e032243d507c743b061ef021e2ec7fcc6d3ab89"
    gas: 45000
  ▶ gasPrice: r {s: 1, e: 10, c: Array(1)}
    hash: "0xdb3be044880d3e9f1c04c37c164191ca6d283479677ab0ff82f220282269a09d"
    input: "0x"
    nonce: 98125
    r: "0x43ab0a587d5bc3bfa3f766a19227d63691a88a57718a8aa08968273d7636c145"
    s: "0x493fe4166f3438fc4feadc2c84693ff09447612a37f3170190269e20c6358206"
    to: "0x904d669c797988433d28c642af6da74e5b10a94c"
    transactionIndex: 0
    v: "0x26"
  ▶ value: r {s: 1, e: 17, c: Array(1)}
  ▶ __proto__ : Object
```

以太坊基本操作及原理

以太坊交易(事务)结构

- from：交易发送者的地址；
- to：交易接收者的地址；
- value：发送者要转移给接收者的以太币数量；
- data（input）：存在的数据字段，如果存在，则是表明该交易是一个创建或者调用智能合约交易；
- Gas Limit（Gas）：表示这个交易允许消耗的最大Gas数量；
- GasPrice：表示发送者愿意支付给矿工的Gas价格；
- nonce：用来区别同一用户发出的不同交易的标记；
- hash：由以上信息生成的散列值（哈希值），作为交易的ID；
- r、s、v：交易签名的三个部分，由发送者的私钥对交易hash进行签名生成。

以太坊基本操作及原理

Gas

Contract Source Code </>

```
14      // assert(b > 0); // Solidi
15      uint c = a / b;
16      // assert(a == b * c + a %
17      return c;
18    }
19
20    function sub(uint a, uint b)
21      assert(b <= a);
22      return a - b;
23    }
24
25    function add(uint a, uint b)
26      uint c = a + b;
27      assert(c >= a);
28      return c;
29    }
30
31    function max64(uint64 a, uint
32      return a >= b ? a : b;
33    }
```

=>

Contract Creation Code

```
PUSH1 0x60
PUSH1 0x40
MSTORE
CALLDATASIZE
ISZERO
PUSH2 0x00f6
JUMPI
PUSH4 0xffffffff
PUSH1 0xe0
PUSH1 0x02
```

以太坊基本操作及原理

Gas

Contract Creation Code

```
PUSH1 0x60
PUSH1 0x40
MSTORE
CALLDATASIZE
ISZERO
PUSH2 0x00f6
JUMPI
PUSH4 0xffffffff
PUSH1 0xe0
PUSH1 0x02
```

=>

Operation	Gas	Description
ADD/SUB	3	Arithmetic operation
MUL/DIV	5	
ADDMOD/MULMOD	8	
AND/OR/XOR	3	Bitwise logic operation
LT/GT/SLT/SGT/EQ	3	Comparison operation
POP	2	Stack operation
PUSH/DUP/SWAP	3	
MLOAD/MSTORE	3	Memory operation
JUMP	8	Unconditional jump
JUMPI	10	Conditional jump
SLOAD	200	Storage operation
SSTORE	5,000/ 20,000	
BALANCE	400	Get balance of an account
CREATE	32,000	Create a new account using CREATE
CALL	25,000	Create a new account using CALL

以太坊基本操作及原理

Gas

Contract Creation Code

```
PUSH1 0x60
PUSH1 0x40
MSTORE
CALLDATASIZE
ISZERO
PUSH2 0x00f6
JUMPI
PUSH4 0xffffffff
PUSH1 0xe0
PUSH1 0x02
```

=>

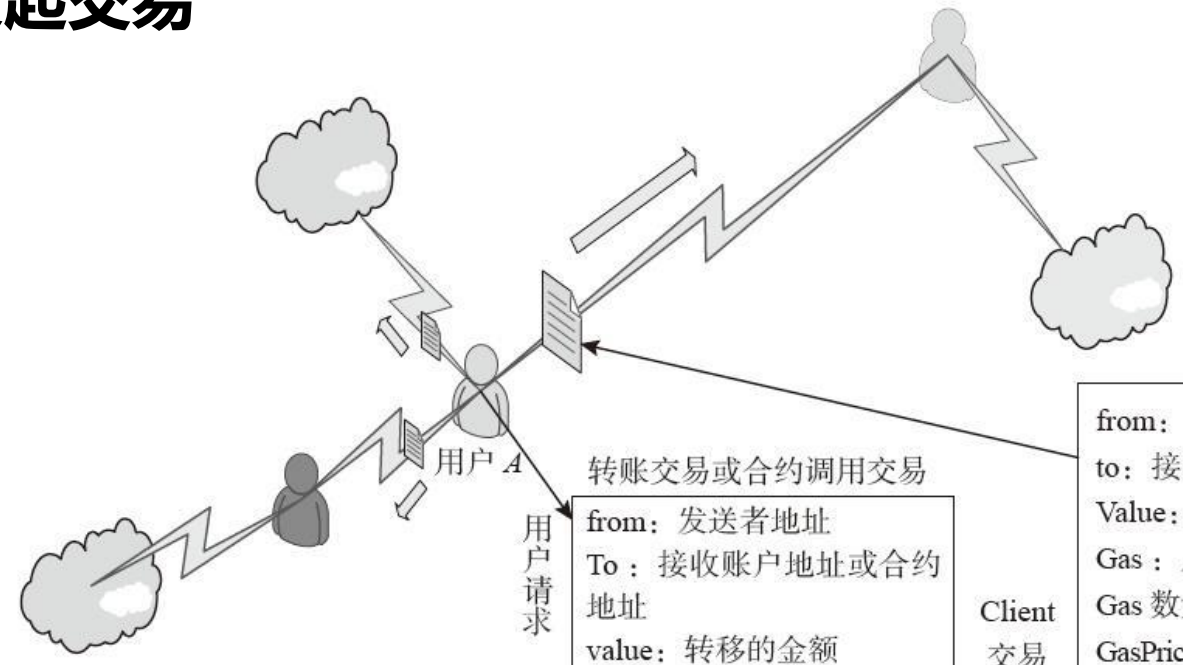
Operation	Gas	Description
ADD/SUB	3	Arithmetic operation
MUL/DIV	5	
ADDMOD/MULMOD	8	
AND/OR/XOR	3	Bitwise logic operation
LT/GT/SLT/SGT/EQ	3	Comparison operation
POP	2	Stack operation
PUSH/DUP/SWAP	3	
MLOAD/MSTORE	3	Memory operation
JUMP	8	Unconditional jump
JUMPI	10	Conditional jump
SLOAD	200	Storage operation
SSTORE	5,000/ 20,000	
BALANCE	400	Get balance of an account
CREATE	32,000	Create a new account using CREATE
CALL	25,000	Create a new account using CALL

问题：Gas，GasPrice，GasLimit，GasUsed的区别?

以太坊基本操作及原理

以太坊交易(事务)结构

发起交易

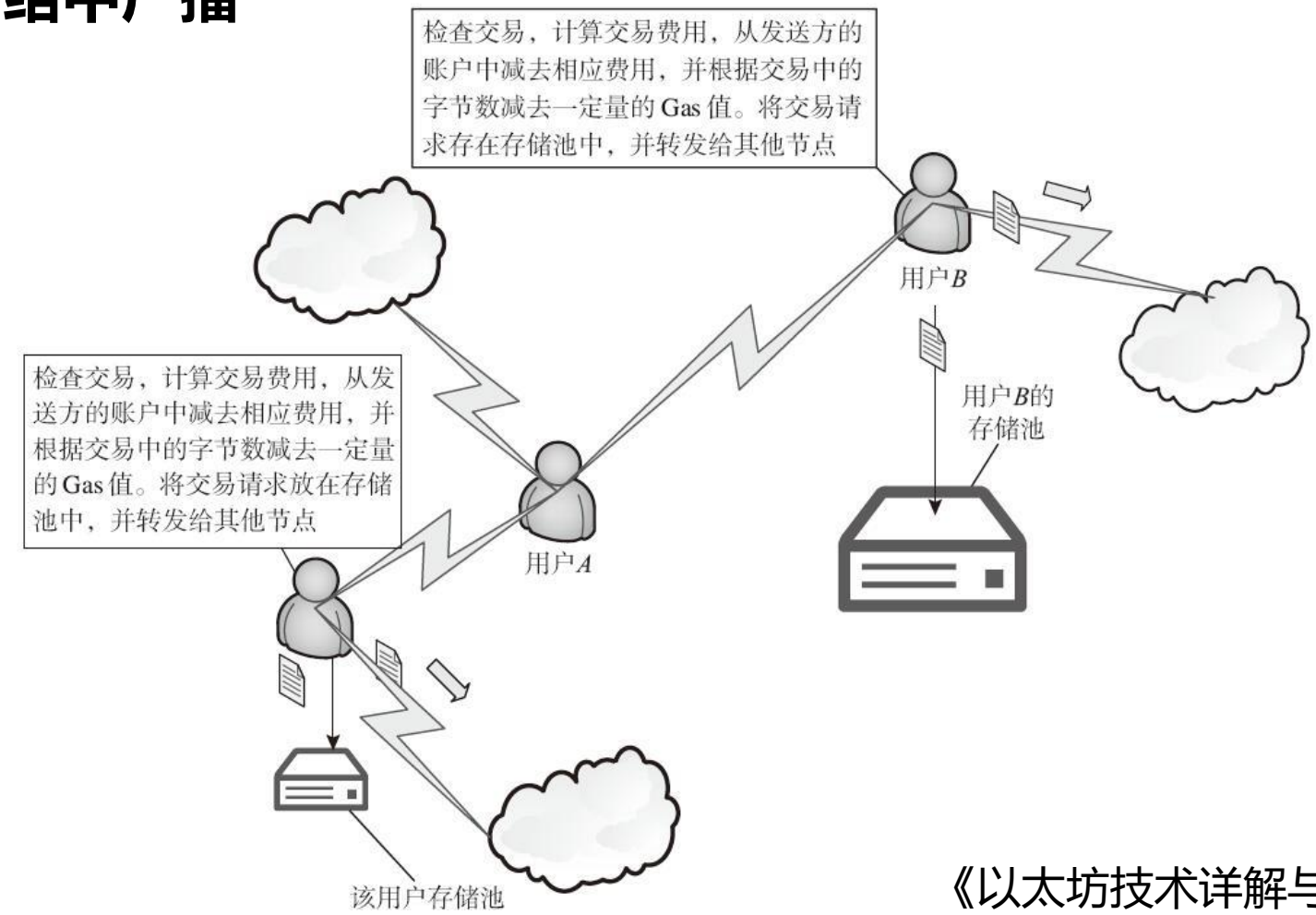


《以太坊技术详解与实战》

以太坊基本操作及原理

以太坊交易(事务)结构

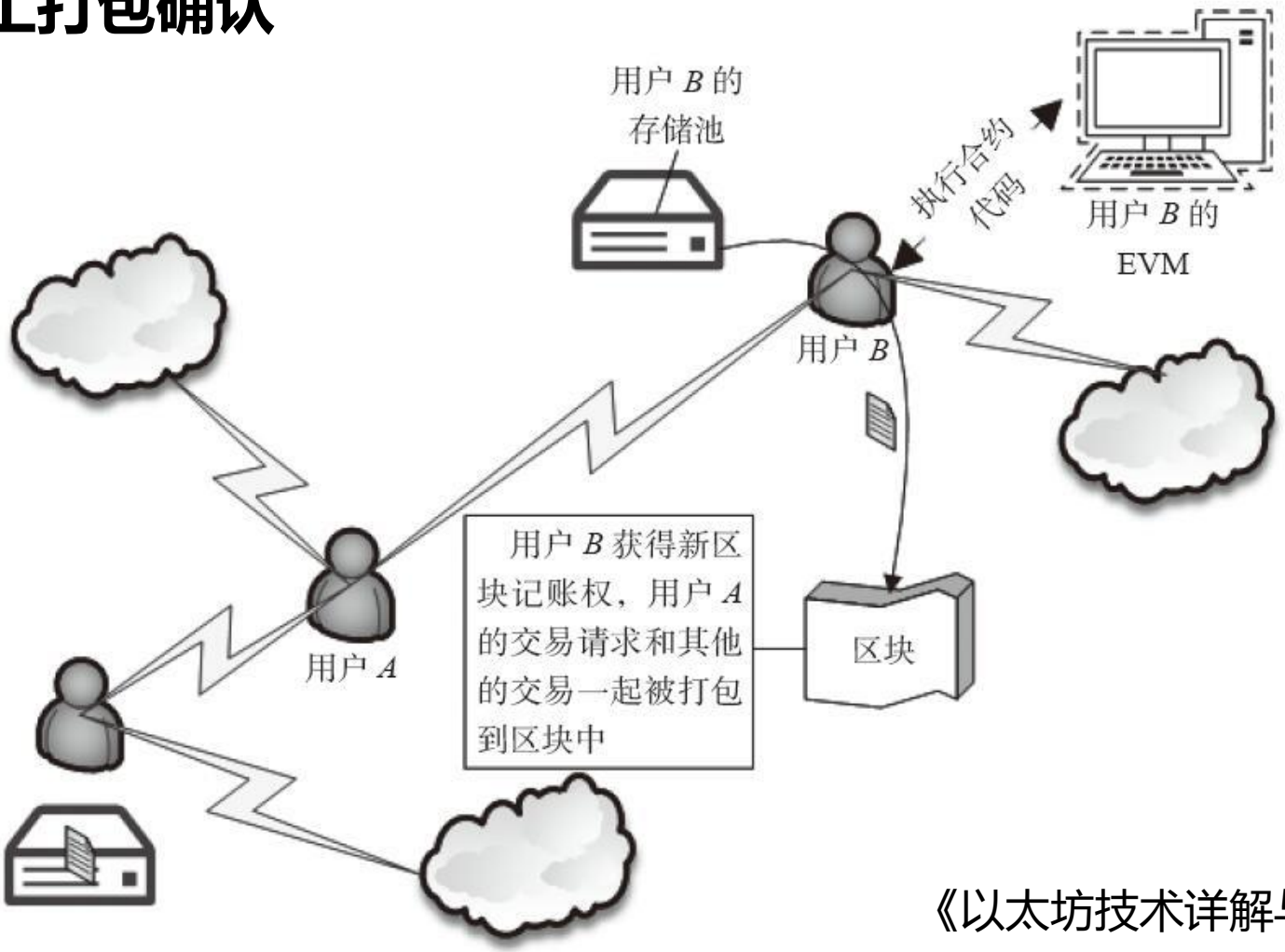
网络中广播



以太坊基本操作及原理

以太坊交易(事务)结构

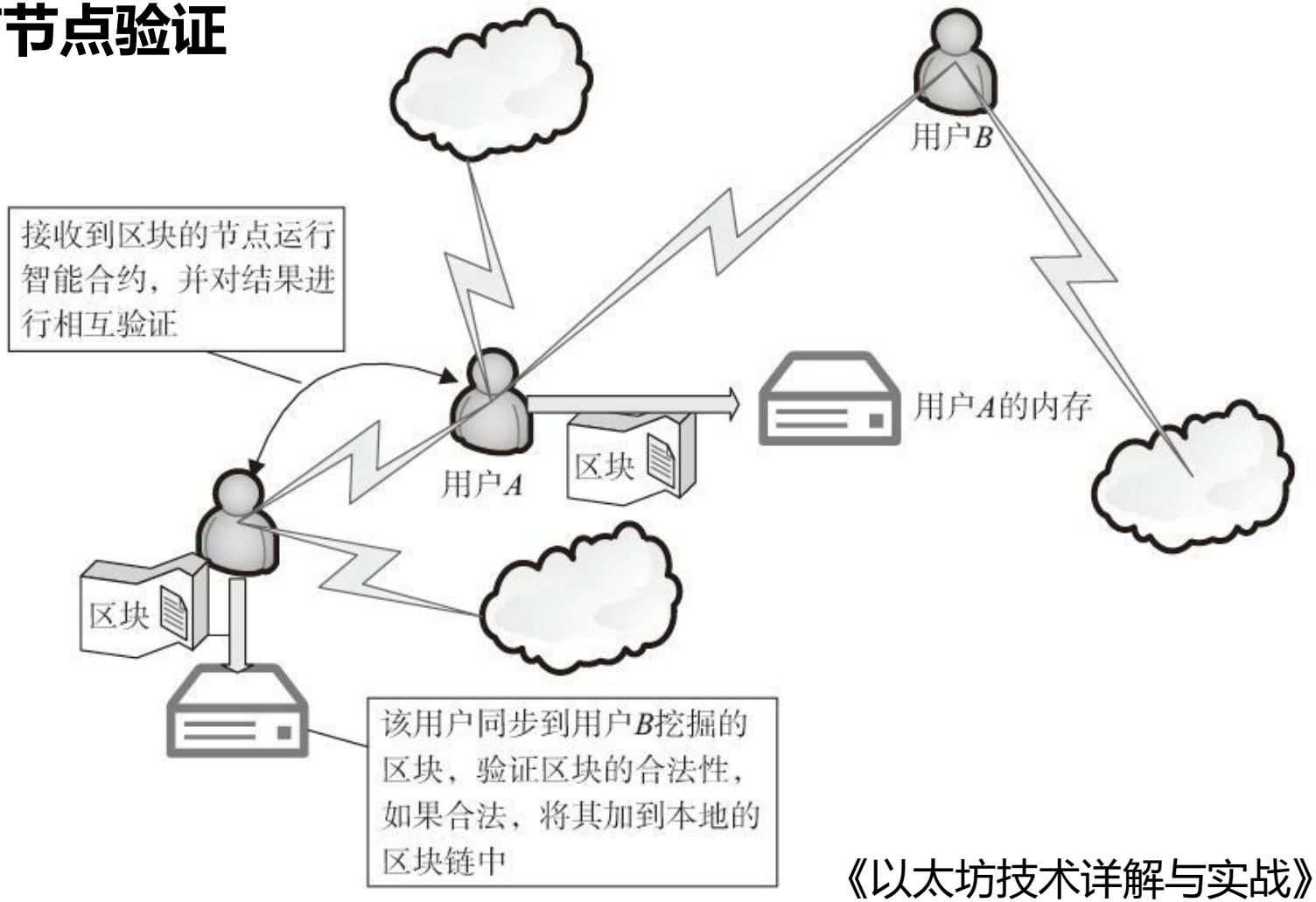
矿工打包确认



以太坊基本操作及原理

以太坊交易(事务)结构

所有节点验证



《以太坊技术详解与实战》

以太坊基本操作及原理

- **以太坊交易(事务)结构**
  - **交易中的Nonce值**

    from账户发出交易的次数, 同一账户的交易会被依次确认

    问题：nonce值有什么用?

以太坊基本操作及原理

以太坊交易(事务)结构

交易中的Nonce值

from账户发出交易的次数, 同一账户的交易会被依次确认

问题：nonce值有什么用?

- 区块中的nonce值：挖矿
- 交易中的nonce值：
  1. 确认交易顺序
  2. 防止双花
  3. 撤销pending中的交易
  4. 确定生成的合约地址

以太坊基本操作及原理

以太坊交易(事务)结构

试试这条交易 0xc3df4b16dcc80785241a913059ee8142656ec38a748a38a35cfea791820bfaaf

问题：to为什么是空的?

```
> web3.eth.getTransaction("0xc3df4b16dcc80785241a913059ee8142656ec38a748a38a35cfe
a791820bfaaf")
<- {blockHash: "0x2f89bbebdc680a37ce8d4f71594793dc01b66a6e3ba29ad99225e7fd9a23f9
▼ 7a", blockNumber: 5240655, from: "0xa450fcdb1079cdacfeff221087c00536e97e365a"
  , gas: 1222666, gasPrice: r, …}
    blockHash: "0x2f89bbebdc680a37ce8d4f71594793dc01b66a6e3ba29ad99225e7fd9a23f97a"
    blockNumber: 5240655
    from: "0xa450fcdb1079cdacfeff221087c00536e97e365a"
    gas: 1222666
  ▶ gasPrice: r {s: 1, e: 9, c: Array(1)}
    hash: "0xc3df4b16dcc80785241a913059ee8142656ec38a748a38a35cfea791820bfaaf"
    input: "0x606060405234156200001057600080fd5b6200009c6040805190810160405280600c8
    nonce: 0
    r: "0x9bd0a592f7e45a97cae551358e167d42d00c85453290bf7876c264fb3daf4329"
    s: "0x1d969e043b487fb9f396a06cc12910979288630a3aa2bef68320f0900fcc465c"
    to: null
    transactionIndex: 65
    v: "0x26"
  ▶ value: r {s: 1, e: 0, c: Array(1)}
  ▶ __proto__: Object
```

以太坊基本操作及原理

- 以太坊交易(事务)结构

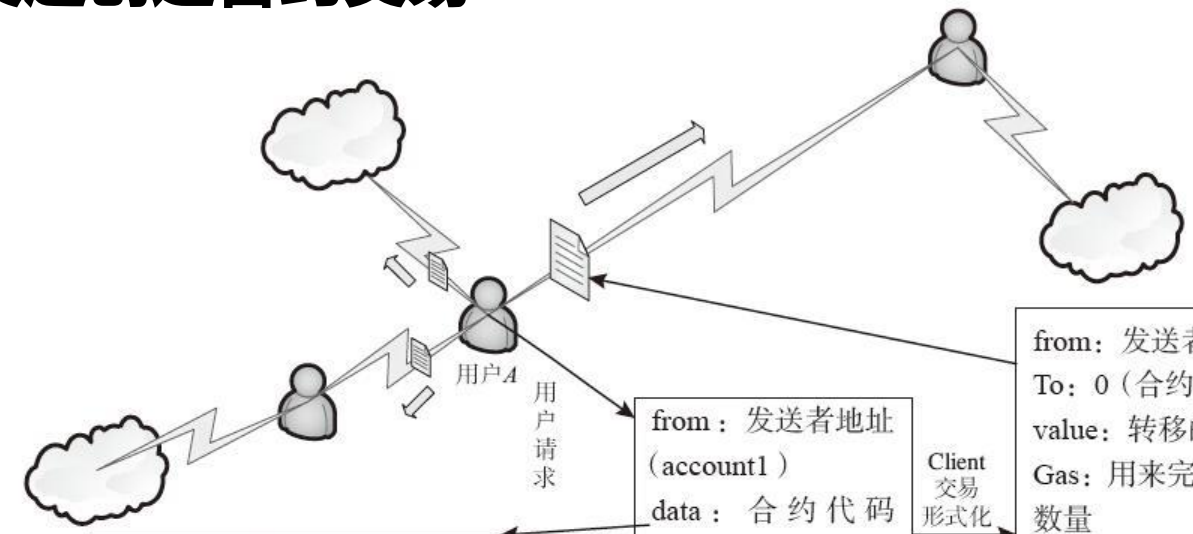
  问题：to为什么是空的?

  创建以太坊智能合约时，to的地址是空的。，如果为空则意味这是一个创建智能合约的交易

以太坊基本操作及原理

以太坊交易(事务)结构

发起创建合约交易



from：发送者地址（H（account1））
To：0（合约创建交易地址为空）
value：转移的金额
Gas：用来完成交易（执行合约）的 Gas 数量
GasPrice：交易中愿意付出的 Gas 的单价
data：编译成字节码的智能合约
nonce：整数，用户 *A* 发起的每笔交易都有唯一的编号，表示该交易是用户发起的第几笔交易

Client 交易形式化

from：发送者地址（account1）
data：合约代码（用户编写的代码）
value：转移的金额
Gas：用来完成交易的 Gas 数量
GasPrice：交易中愿意付出的 Gas 的单价

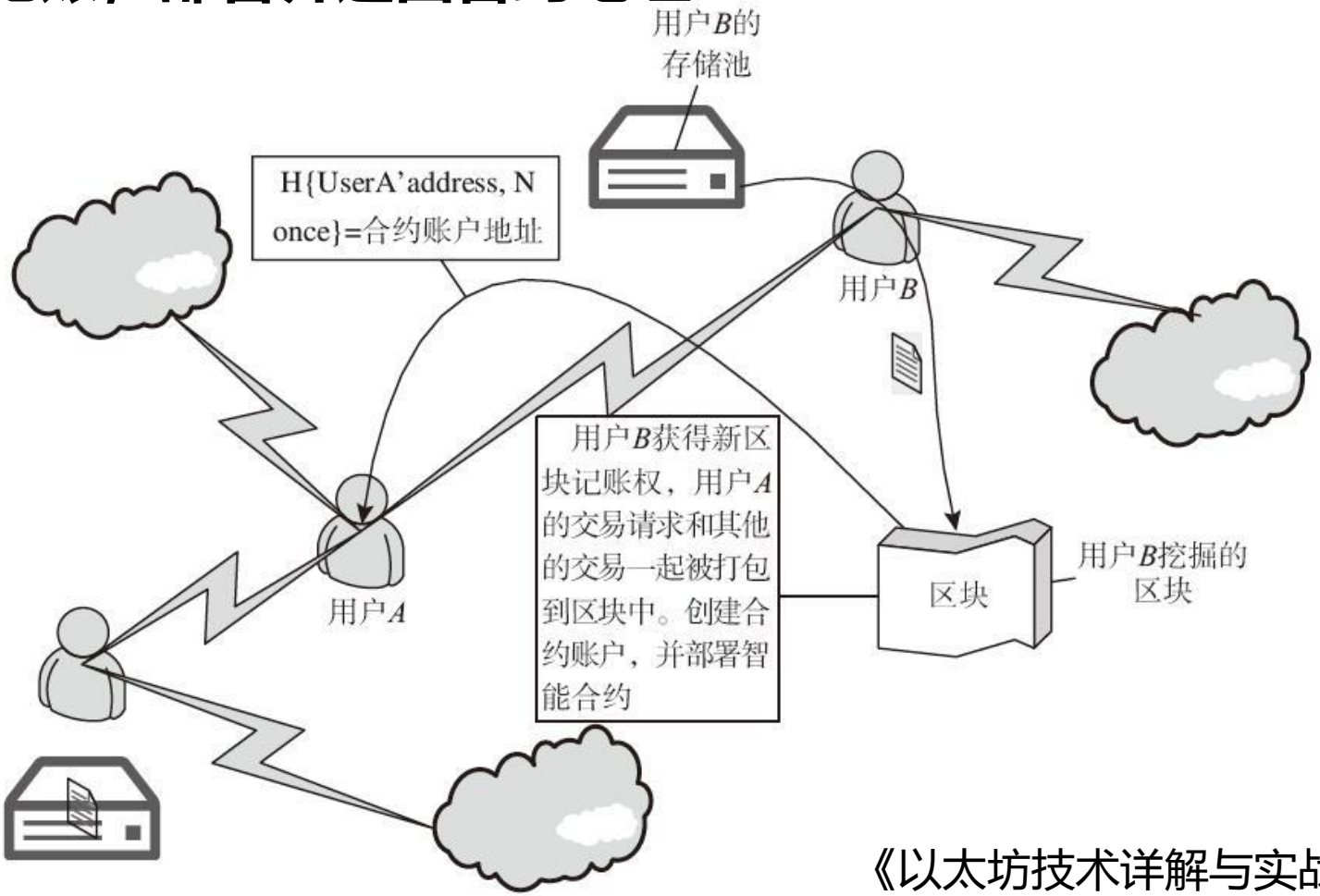
```
pragma solidity ^0.4.0;

contract SimpleStorage {
  string storedData;
  event stored(address from,string s)
  function set(string s) {
    storedData = s;
    stored (msg.sender,s);
  }
  function get() constant returns (string)
{
    return storedData;
  }
}
```

以太坊基本操作及原理

以太坊交易(事务)结构

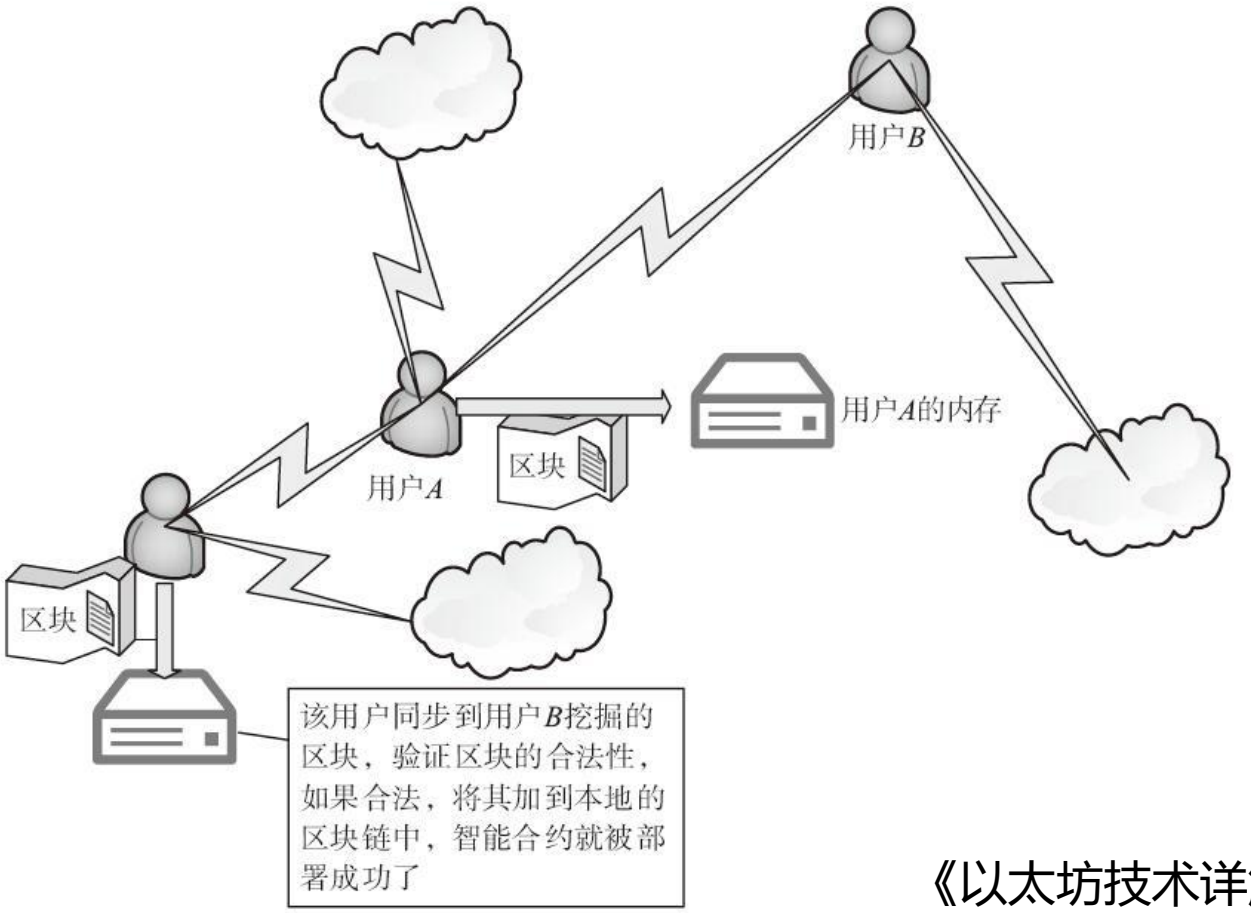
- 矿工记账，部署并返回合约地址



以太坊基本操作及原理

以太坊交易(事务)结构

- 完成智能合约部署



《以太坊技术详解与实战》

以太坊基本操作及原理

以太坊合约代码

获取合约地址

```
> web3.eth.getTransactionReceipt("0xc3df4b16dcc80785241a913059ee8142656ec38a748a38a35cfea791820bfaaf")
{blockHash: "0x2f89bbebdc680a37ce8d4f71594793dc01b66a6e3ba29ad99225e7fd9a23f97a", blockNumber: 5240655, contractAddress: "0xc86bdf9661c62646194ef29b1b8f5fe226e8c97e", cumulativeGasUsed: 6175103, from: "0xa450fcdb1079cdacfeff221087c00536e97e365a", …}
  blockHash: "0x2f89bbebdc680a37ce8d4f71594793dc01b66a6e3ba29ad99225e7fd9a23f97a
  blockNumber: 5240655
  contractAddress: "0xc86bdf9661c62646194ef29b1b8f5fe226e8c97e"
  cumulativeGasUsed: 6175103
  from: "0xa450fcdb1079cdacfeff221087c00536e97e365a"
  gasUsed: 1132894
▶ logs: [{…}]
  logsBloom: "0x00000000000000000000000000000000000000000000000000000000000000
  status: "0x1"
  to: null
  transactionHash: "0xc3df4b16dcc80785241a913059ee8142656ec38a748a38a35cfea79182
  transactionIndex: 65
▶ __proto__ : Object
```

以太坊基本操作及原理

以太坊合约代码

获取合约代码

```
> web3.eth.getCode("0xc86bdf9661c62646194ef29b1b8f5fe226e8c97e")
<· "0x6060604052600436106100785760003 57c0100000000000000000000000000000000000000000000
000000000000000000900463ffffffff16806306661abd1461007d5780632e323f5d146100a657806
37e56d25f146101155780638 2ed13d1146101c9578063b1a867d51461031f578063c2ff0f151461
0382575b600080fd5b341561008857600080fd5b61009061042d565b60405180828152602001915
05060405180910390f35b34156100b157600080fd5b61011360048080359060200190919080359 0
6020019091908035906020019082018035906020019080806 01f0160208091040260200160405 19
0810160405280939291908181526020018383808284378201915050505050509190505061043356
5b005b341561012057600080fd5b6101c7600480803590602001909190803590602001908201803
5906020019080806 01f01602080910402602001604051908101604052809392919081815260200 1
838380828437820191505050505050919080351515906020019091908035906020019082018035 9
06020019080806 01f016020809104026020016040519081016040528093929190818152602001 83
83808284378201915050505050509190505061 05c5565b005b34156101d457600080fd5b6101f36
0048080359060200190919080359060200190919050506107 5b565b6040518086 73ffffffffffff
ffffffffffffffffffffffffffff1673ffffffffffffffffffffffffffffffffffffffff1681526
020018060200185815260200184151515158152602001806020018381038352878181518152602 0
01915080519060200190808383600 05b838110156102795780820151818401526020810190506 10
25e565b505050509050908101906 01f1680156102a65780820380516001836020036101000a0319
168152602001915 05b508381038252848181518152602001915080519060200190808383600 05b8
38110156102df5780820151818401526020810190506102c4565b50505050905090810190601f16
801561030c57808203805160018360200361 01000a031916815260200191505b50975050505050 5
```

以太坊基本操作及原理

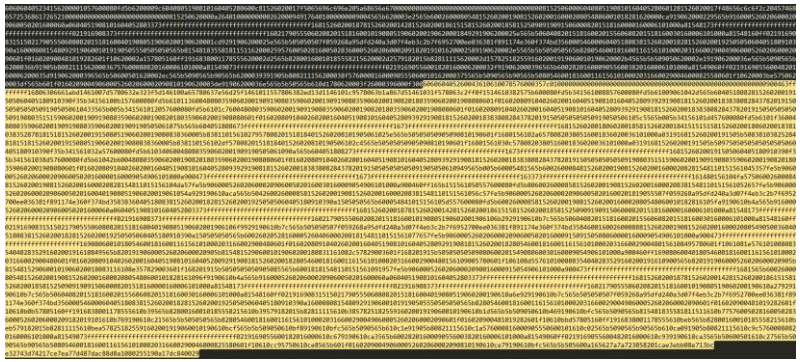
以太坊合约代码

问题：交易的input和getCode为什么不一样?

Contract Code

存在对应地址的代码

eth.getCode()



Creation Code

交易的input

getTrasaction().input

包含与被包含
本质用处不同

以太坊基本操作及原理

运行节点进行开发

- 为保证演示和同学们同步, windows可以从该网页下载geth.
- geth -datadir shuju init genesis.json
- geth -datadir shuju -networkid 2018 -rpc -rpcaddr 你的IP -rpccorsdomain "*" console
- personal.newAccount("");
- personal.unlockAccount(eth.account[0],"");
- http://remix.ethereum.org
- miner.start(); admin.sleepBlocks(1); miner.stop();