Blockchain Systems and Smart Contracts

Agenda

- **Blockchain systems**
- **Smart contracts and applications**
- **Hyperchain**
  - Introduction
  - Architecture
  - Hyperchain Dev
- **Ethereum**
  - Introduction
  - Interesting Ethereum-based projects
  - Problems & challenges

BLOCKCHAIN SYSTEMS

What is blockchain?

- **Cryptocurrencies** have attracted a lot of attention.

- Core technological innovation powering cryptocurrencies is a **distributed ledger**.

- **Blockchain** technology provides an **open**, **decentralized** and **fault-tolerant** transaction mechanism.

Open Source Blockchain Systems

GitHub Repository	Duration(y)
bitcoin/bitcoin	5.77
ethereum/go-ethereum	2.36
ethereum/mist	1.14
dogecoin/dogecoin	2.16
ethereum/cpp-ethereum	2.68
ripple/ripple-lib	1.77
steemit/steem	0.32
AugurProject/augur	1.27




SMART CONTRACTS

Definition

A smart contract is a computer program executed in a secure environment that directly controls digital assets

A smart contract is a **computer program** executed in a secure environment that directly controls digital assets

A computer program is a collection of instructions that performs a specific task when executed by a computer. A computer requires programs to function, and typically executes the program's instructions in a central processing unit.

Wikipedia

Example: bet on an event

```
if HAS_EVENT_X_HAPPENED() is true:
   send(party_A, 1000)
else:
   send(party_B, 1000)
```

A smart contract is a computer program executed in a **secure environment** that directly controls digital assets

Properties of Secure Environments

- Correctness of execution
  - The execution is done correctly, is not tampered
- Integrity of code and data
- Optional properties
  - Confidentiality of code and data
  - Verifiability of execution
  - Availability for the programs running inside

Examples of secure environments

- Servers run by trusted parties
- Decentralized computer network (ie. blockchains)
- Quasi-decentralized computer network (ie. consortium blockchains)
- Servers secured by trusted hardware (e.g. SGX)

A smart contract is a computer program executed in a secure environment that **directly controls** digital assets

Example

- Legal contract: “I promise to send you $100 if my lecture is rated 1*”
- Smart contract: “I send $100 into a computer program executed in a secure environment which sends $100 to you if the rating of my lecture is 1*, otherwise it eventually sends $100 back to me”

A smart contract is a computer program executed in a secure environment that directly controls **digital assets**

What are digital assets?

- A broad category
  - Domain name
  - Website
  - Money
  - Anything tokenisable (e.g. gold, silver, stock share etc)
  - Game items
  - Network bandwidth, computation cycles

Example: top 5 crowdfunding campaigns in history

Rank	Project	Category	Platform	Campaign end date	Campaign target	Amount raised
1	*Star Citizen*	Video game	Kickstarter, independent	Ongoing	$500,000	$90,009,649
2	*Elio Motors*	Automotive - Low-cost, high mileage vehicle	Independent	Ongoing	-	$21,161,869
3	*Pebble Time*	Smartwatch	Kickstarter	Mar 27, 2015	$500,000	$20,338,986
4	*Ethereum*	Cryptocurrency	Bitcoin, Independent	Sep 2, 2014	-	$18,439,086
5	*Coolest Cooler*	Product Design	Kickstarter	Aug 29, 2014	$50,000	$13,285,226

Rank	Project	Category	Platform	Campaign end date	Campaign target	Amount raised
1	*Star Citizen*	Video game	Kickstarter, independent	Ongoing	$500,000	$90,009,649
2	*Elio Motors*	Automotive - Low-cost, high mileage vehicle	Independent	Ongoing	-	$21,161,869
3	*Pebble Time*	Smartwatch	Kickstarter	Mar 27, 2015	$500,000	$20,338,986
4	*Ethereum*	Cryptocurrency	Bitcoin, Independent	Sep 2, 2014	-	$18,439,086
5	*Coolest Cooler*	Product Design	Kickstarter	Aug 29, 2014	$50,000	$13,285,226

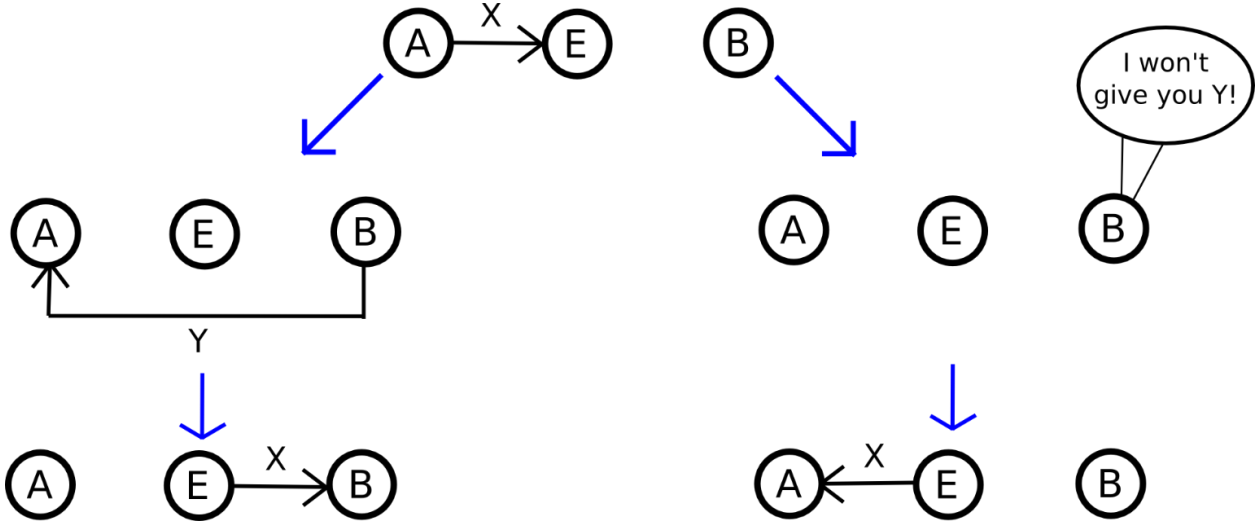
Star Citizen sold virtual spaceships in their game for $500 each

Rank	Project	Category	Platform	Campaign end date	Campaign target	Amount raised
1	*Star Citizen*	Video game	Kickstarter, independent	Ongoing	$500,000	$90,009,649
2	*Elio Motors*	Automotive - Low-cost, high mileage vehicle	Independent	Ongoing	-	$21,161,869
3	*Pebble Time*	Smartwatch	Kickstarter	Mar 27, 2015	$500,000	$20,338,986
4	*Ethereum*	Cryptocurrency	Bitcoin, Independent	Sep 2, 2014	-	$18,439,086
5	*Coolest Cooler*	Product Design	Kickstarter	Aug 29, 2014	$50,000	$13,285,226

Ethereum Foundation sold 60,102,206 digital tokens which will be useful in a decentralized network

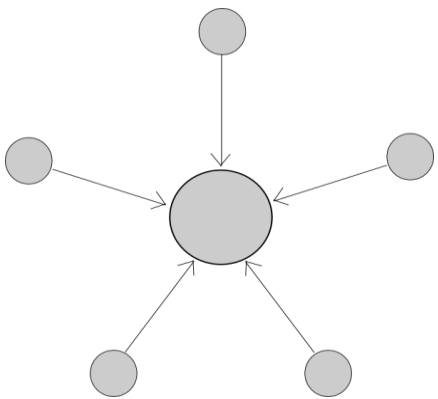
What are smart contracts’ applications?

Example: escrow service for exchange



Example: multisig

- Require M of N “owners” to agree in order for a particular digital asset to be transferred
  - Individual use cases
    - eg. two-factor authentication
  - Intra-organizational use cases



A lot more interesting applications

- Individual/intra-organizational
  - Complex access policies depending on amount, withdrawal limits, etc
  - Dead man's switch, "digital will"
    - E.g When the owner dies, transfer all assets to someone
- General
  - Prediction markets
  - Insurance
  - Micro-payments for computational services (file storage, bandwidth, computation, etc)

Why smart contracts?

- Automated processing
- Trust reduction
  - Trust the secure environments, not a very large number of contract enforcement mechanisms
- Unambiguous, terms clearly expressed in code
  - Question: how to express terms clearly in code?

HYPERCHAIN

What is Hyperchain?

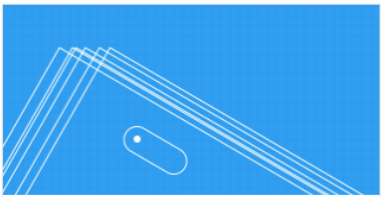
- The Hyperchain Affiliate Chain Platform addresses the blockchain technology needs of enterprises, government agencies, and industry alliances to provide enterprise-level blockchain network solutions.
- Support enterprises to rapidly deploy, expand and configure blockchain networks based on existing cloud platforms, and perform real-time visual monitoring of the operating status of blockchain networks.
- This is a blockchain core system platform that complies with ChinaLedger's technical specifications.

Hyperchain Architecture and Applications

- Hyperchain has core features such as authentication node authorization mechanism, multi-level encryption mechanism, consensus mechanism, Turing complete high-performance smart contract execution engine, and data management.
- It is a coalition-based basic technology platform with complete functions and high performance.
- In the application scenarios that are faced by enterprises and industry alliances, Hyperchain can provide high-quality underlying blockchain support platforms, convenient and reliable integrated solutions for decentralized applications such as digital asset clearing, data credible deposit storage, and intermediary transactions.

Hyperchain Dev

- Help developers rapidly develop blockchain applications



Electronic wallet

Through smart contracts, you can create your own digital assets and set the number of shares. At the same time, you can transfer assets to other users through transfer.

Digital integral circulation

A multi-institution together constitutes a points alliance. Consumers receive points issued by any institution for consumption by any institution and can be used by any institution in the coalition.

Supply chain traceability

The manufacturer registers the manufactured goods with the blockchain, and then the dealers register the goods on the chain, and finally the retailers buy it from the consumer. The blockchain records the entire supply chain process.

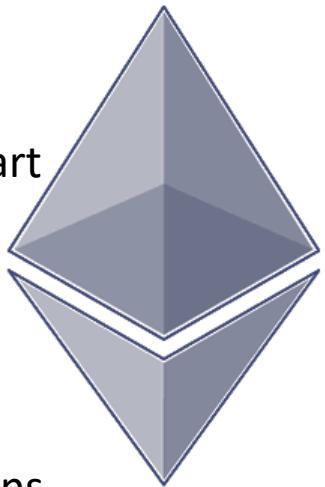
- Smart contract online IDE | API | Block Browser

https://dev.hyperchain.cn/

ETHEREUM: THE FIRST BLOCKCHAIN-BASED SMART CONTRACT PLATFORM

Ethereum

- Blockchain with expressive programming language
  - Programming language makes it ideal for smart contracts
- Why?
  - Most public blockchains are cryptocurrencies
    - Can only transfer coins between users
  - Smart contracts enable much more applications



Analogy: Most existing blockchain protocols were designed like

bitcoin

OR
THIS

MONERO

why not make a protocol that works like

OR
THIS

OR
THIS

How Ethereum Works

- Two types of account:
  - **Normal account** like in Bitcoin
    - has balance and address
  - **Smart Contract account**
    - like an object: containing (i) code, and (ii) private storage (key-value storage)
    - Code can
      - Send ETH to other accounts
      - Read/write storage
      - Call (ie. start execution in) other contracts

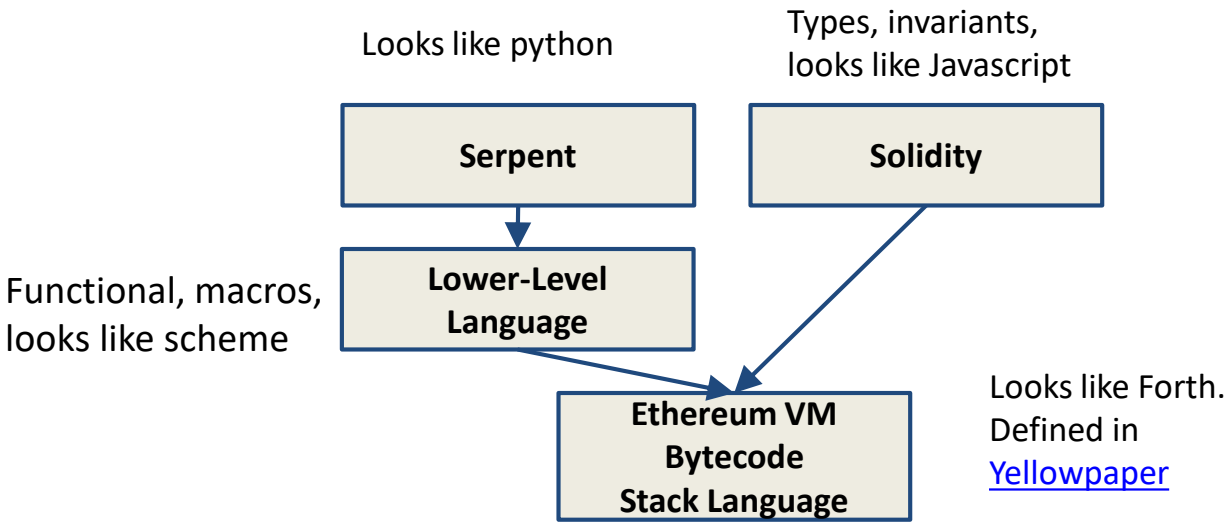
DNS: The “Hello World” of Ethereum

```
data domains[](owner, ip)

def register(addr):
    if not self.domains[addr].owner:
        self.domains[addr].owner = msg.sender

def set_ip(addr, ip):
    if self.domains[addr].owner == msg.sender:
        self.domains[addr].ip = ip
```

Private Storage

Can be invoked by other accounts

Ethereum Languages



Slide is courtesy of Andrew Miller

Example

```
1  contract Greetings {
2      string greeting;
3      function Greetings (string _greeting) public {
4          greeting = _greeting;
5      }
6
7      /* main function */
8      function greet() constant returns (string) {
9          return greeting;
10     }
11 }
```

What you write

What other see on the blockchain

```
60606040526040516102503803806102508339810160
40528.....
```

```
PUSH 60
    PUSH 40
    MSTORE
    PUSH 0
    CALLDATALOAD
.....
```

What people get from the disassembler

Transactions in Ethereum

- Normal transactions like Bitcoin transactions
  - Send tokens between accounts
- Transactions to contracts
  - like function calls to objects
  - specify which object you are talking to, which function, and what data (if possible)
- Transactions to create contracts

Transactions

- **`nonce`** (anti-replay-attack)
- **`to`** (destination address)
- **`value`** (amount of ETH to send)
- **`data`** (readable by contract code)
- **`gasprice`** (amount of ether per unit gas)
- **`startgas`** (maximum gas consumable)
- **`v, r, s`** (ECDSA signature values)

How to Create a Contract?

- Submit a transaction to the blockchain
  - **`nonce:`** previous **`nonce + 1`**
  - **`to:`** empty
  - **`value:`** value sent to the new contract
  - **`data:`** contains the code of the contract
  - **`gasprice`** (amount of ether per unit gas)
  - **`startgas`** (maximum gas consumable)
  - **`v, r, s`** (ECDSA signature values)
- If tx is successful
  - Returns the address of the new contract

How to Interact With a Contract?

- Submit a transaction to the blockchain
  - **`nonce:`** previous **`nonce + 1`**
  - **`to:`** contract address
  - **`value:`** value sent to the new contract
  - **`data:`** data supposed to be read by the contract
  - **`gasprice`** (amount of ether per unit gas)
  - **`startgas`** (maximum gas consumable)
  - **`v, r, s`** (ECDSA signature values)
- If tx is successful
  - Returns outputs from the contract (if applicable)

Blockchain State

Bitcoin's state consists of key value mapping addresses to account balance

Address	Balance (BTC)
0x123456…	10
0x1a2b3f…	1
0xab123d…	1.1

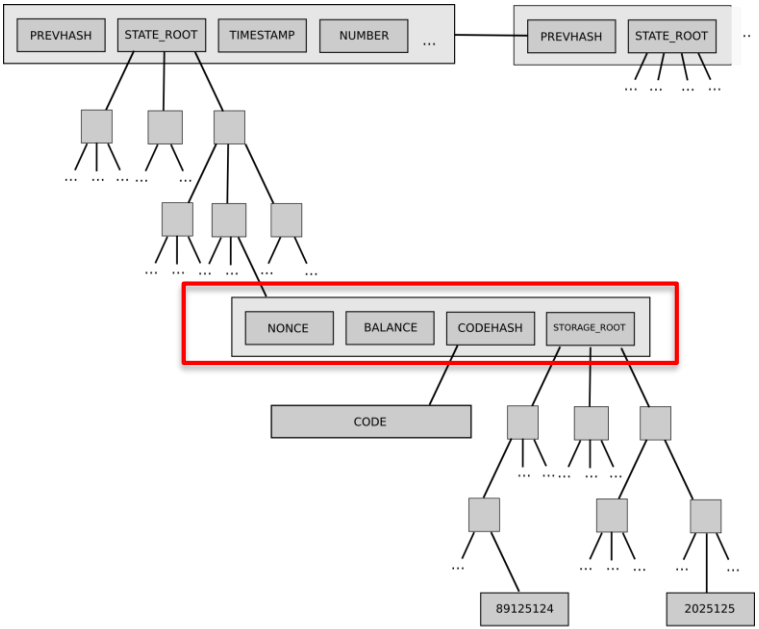
Ethereum's state consists of key value mapping addresses to account objects

Address	Object
0x123456…	X
0x1a2b3f…	Y
0xab123d…	Z

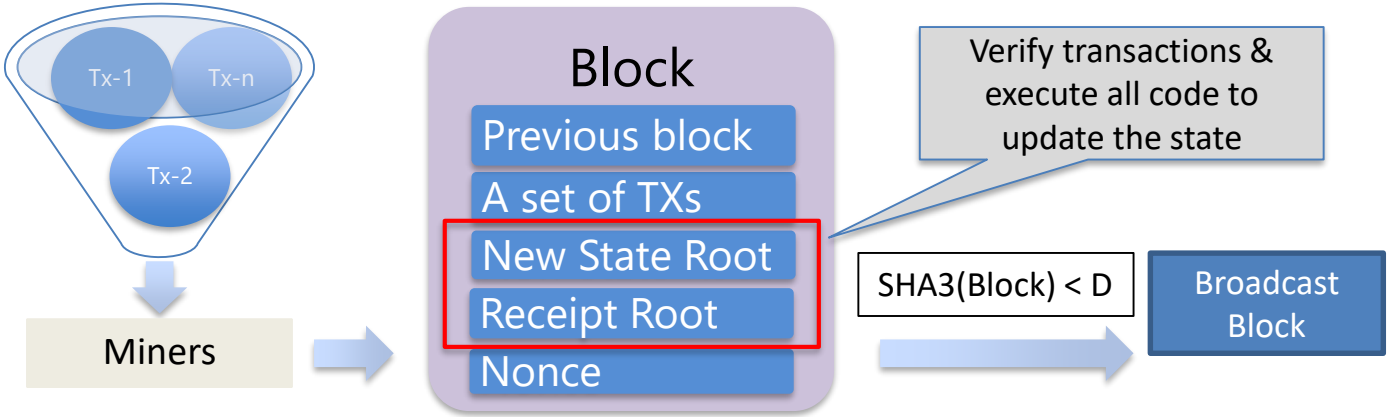
Blockchain != Blockchain State

Account Object

- Every account object contains 4 pieces of data:
  - Nonce
  - Balance
  - Code hash (code = empty string for normal accounts)
  - Storage trie root

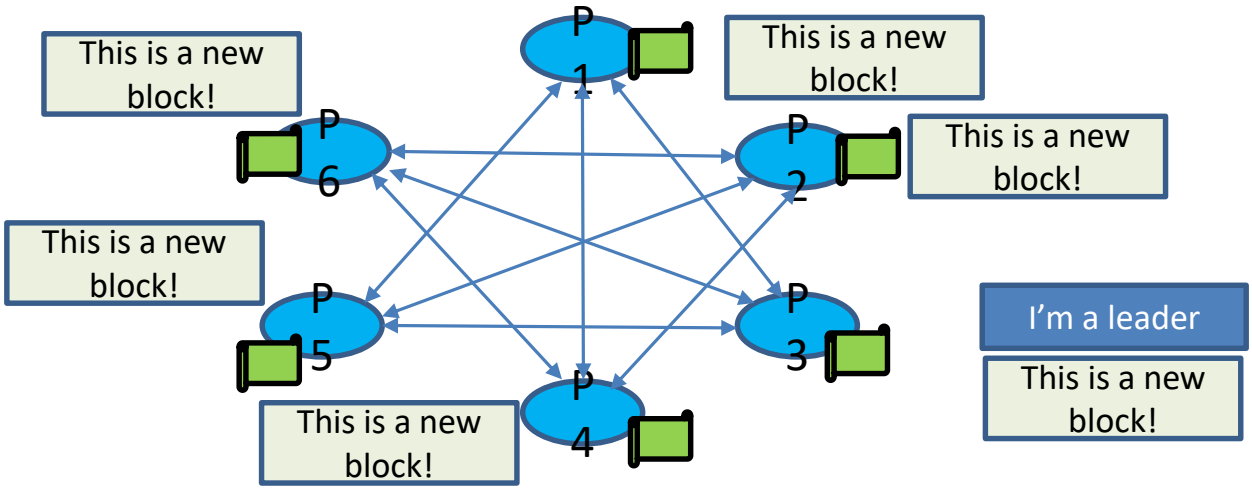


Block Mining



Code execution

- Every (full) node on the blockchain processes every transaction and stores the entire state



Dos Attack Vector

- Halting problem
  - Cannot tell whether or not a program will run infinitely
  - A malicious miner can DoS attack full nodes by including lots of computation in their txs
    - Full nodes attacked when verifying the block

```
uint i = 1;
while (i++ > 0) {
    donothing();
}
```

Solution: Gas

- Charge fee per computational step ("gas")
  - Special gas fees for operations that take up storage

Operation	Gas	GasCost
PUSH1	111741	3
PUSH1	111738	3
MSTORE	111726	12
CALLDATASIZE	111724	2
ISZERO	111721	3
PUSH2	111718	3
JUMPI	111708	10

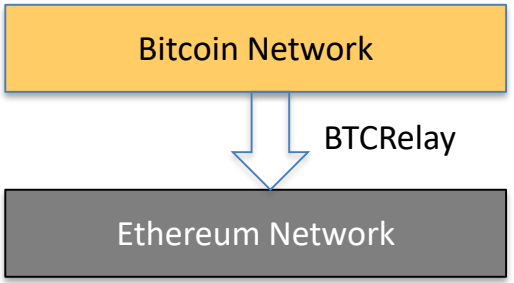
Sender has to pay for the gas

- **`gasprice`**: amount of ether per unit gas
- **`startgas`**: maximum gas consumable
  - If **startgas** is less than needed
    - Out of gas exception, revert the state as if the TX has never happened
    - Sender still pays all the gas
- TX fee = gasprice * consumedgas
- Gas limit: similar to block size limit in Bitcoin
  - Total gas spent by all transactions in a block < Gas Limit

INTERESTING ETHEREUM-BASED PROJECTS

BTCRelay

- A bridge between the Bitcoin blockchain & the Ethereum blockchain
  - Allow to verify Bitcoin transactions within Ethereum network
  - Allow Ethereum contracts to read information from Bitcoin blockchain

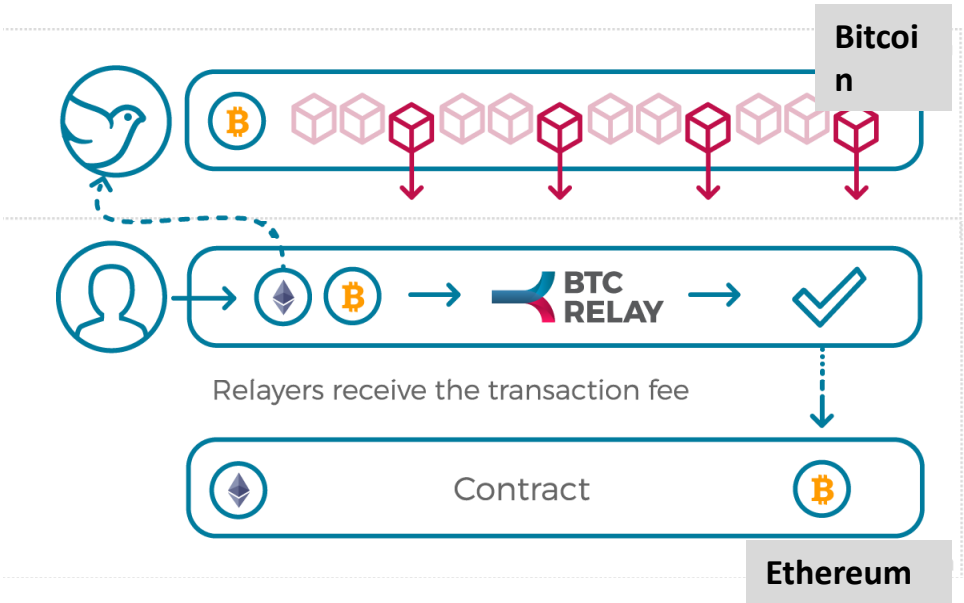


BTCRelay – How it works

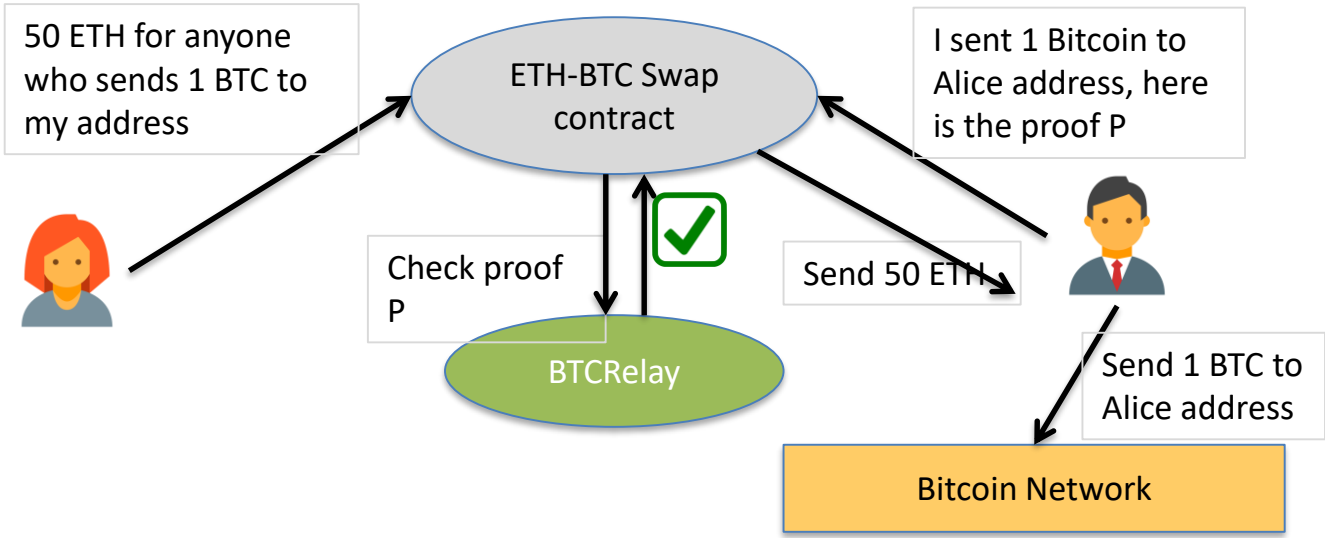
Relayers constantly submit Bitcoin block headers

A Bitcoin transaction is submitted, BTCRelay verifies TX based on the block header

The verified Bitcoin transaction is relayed to the smart contract



BTCRelay Application: ETH-BTC atomic swaps



BTCRelay Application: Contracts can read information of Bitcoin blockchain

E.g. betting on the outcomes of events on Bitcoin blockchain

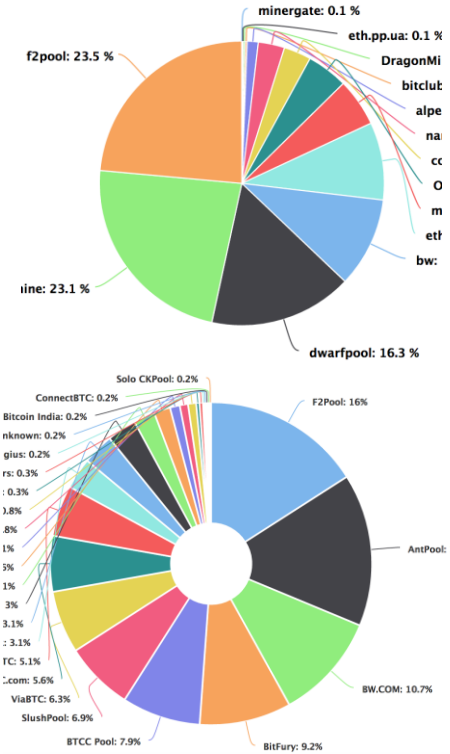
Other Work-in-progress Relays

- Project Alchemy
  - Zcash relay
- Dogecoin/ Litecoin Relay
  - Dogecoin light client on Ethereum by Vitalik
  - Interactive verification for Scrypt pow by Christian

Question: can we build a decentralized exchange between cryptocurrencies using all the relays?

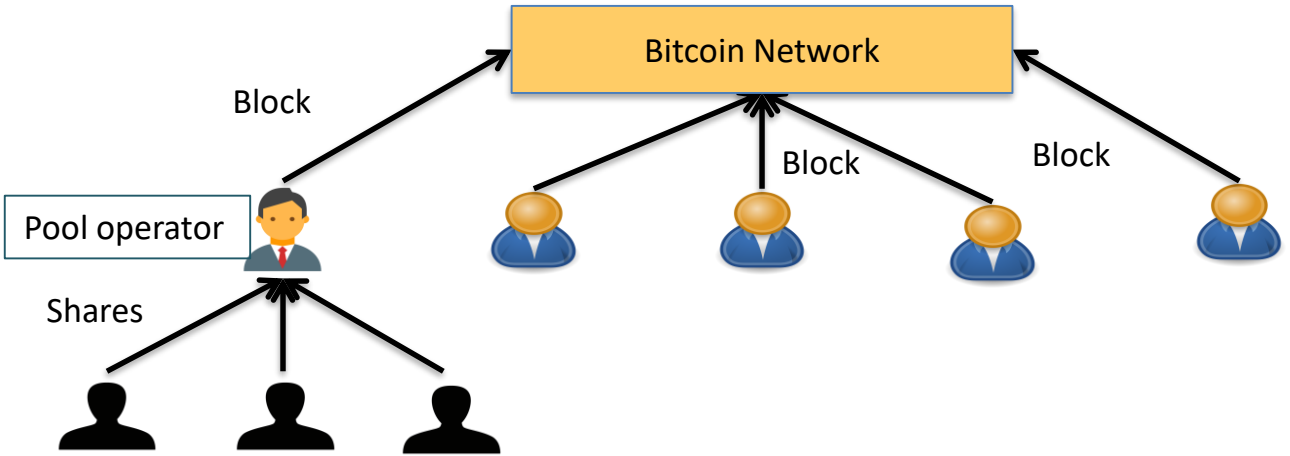
SmartPool

- Decentralized Mining Pools using Smart Contracts
- Problem: mining centralization
  - Miners go to mining pools for stable and frequent rewards
  - Decentralized platforms are secured by centralized entities
    - Transaction censorships
    - Single point of failures



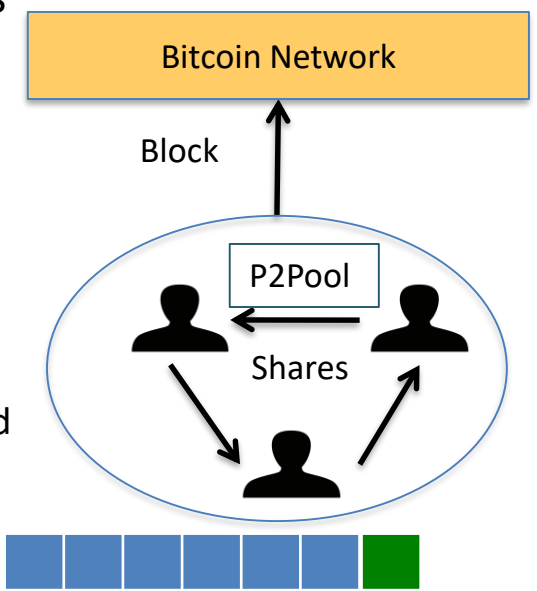
Pooled mining

- Pools track miners' contribution by using shares
  - A share is similar to a block, but required less work to find



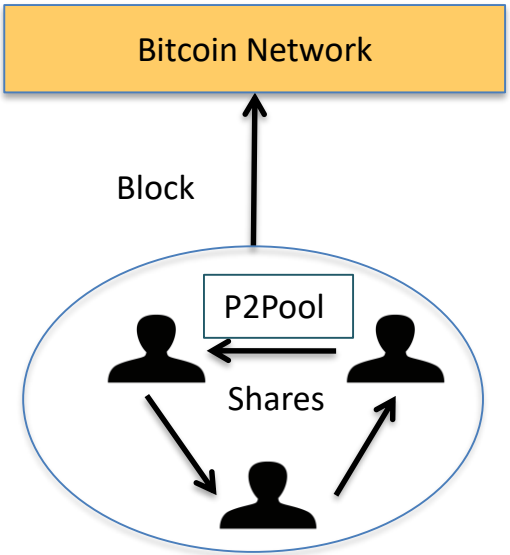
P2Pool: decentralized mining pool

- Miners maintain the pool's contributions by themselves
  - Maintain a share-chain within the pool (just like the blockchain)
  - Pay miners in proportional to their contributions
    - Done in the coinbase transaction
- When a miner finds a share
  - Broadcast to all miners
  - Check if the coinbase tx is correct and extend the share-chain



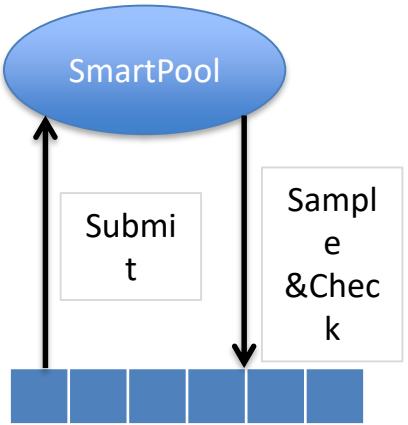
Why P2Pool is Inefficient and not scalable?

- Millions of messages per block (each per share)
  - Expensive to everyone
- Reducing the number of shares?
  - No, will increase the variance of reward



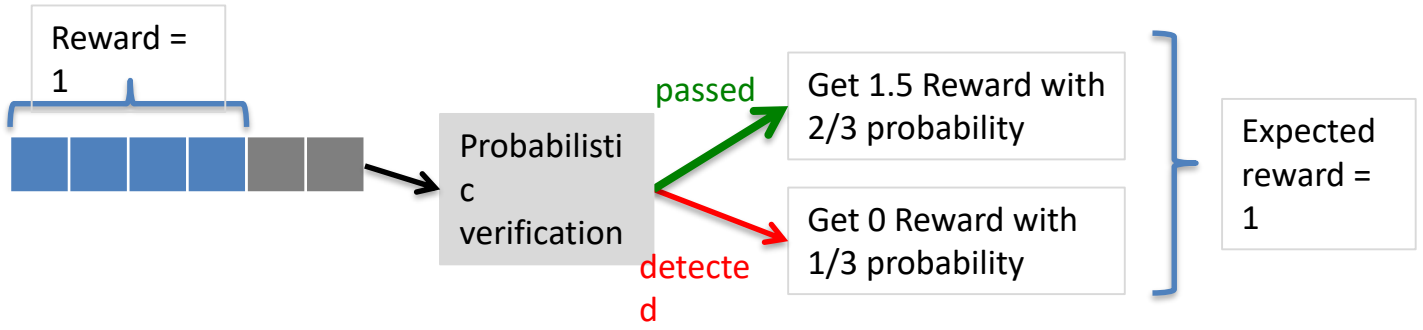
SmartPool: Efficient P2Pool using SmartContract

- Track miners' contributions to the pool in a contract
- Allows batch submissions, e.g. billions of shares in a claim
  - Reduce number of messages (txs) to the contract significantly
- Use probabilistic verification to check a submission
  - Randomly verify only one share per submission
  - Probability of cheating being detected is proportional to the amount of cheating



SmartPool: Disincentivize cheating

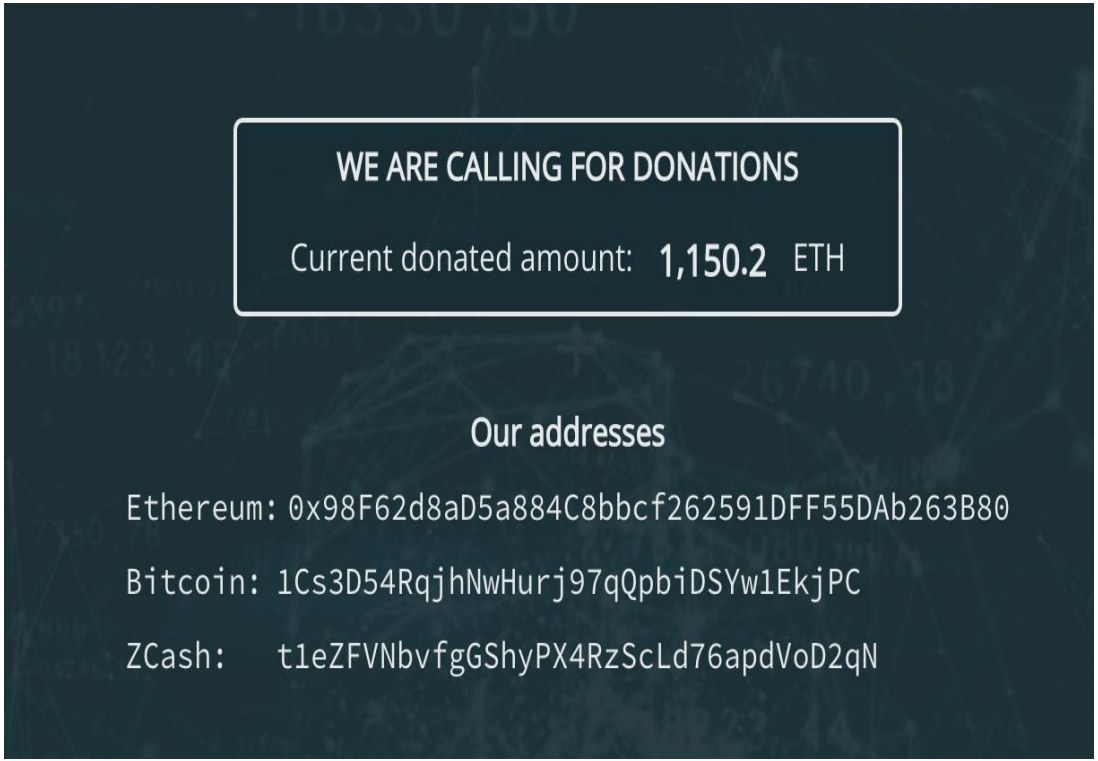
- Payment scheme: pay 0 for a submission if cheating detected
  - Expected reward is the same whether cheating or not
  - Miners have no incentive to cheat



More in the paper

- How to prevent miners from stealing others' shares?
- How to prevent claiming a share multiple times
  - Within a submission
  - Across submissions
- How to verify Ethash PoW?
  - Require huge memory and storage

SmartPool.io is calling for donation



A lot more interesting apps

- TownCrier and Oraclize
  - allow contracts to fetch external data from real websites
  - Enable a lots of applications: betting, insurance, bounty based on real world event
- Augur and Gnosis
  - Prediction market: predict the outcome of real world event to get reward
- Many others: theDao, iConomi, Golem, etc

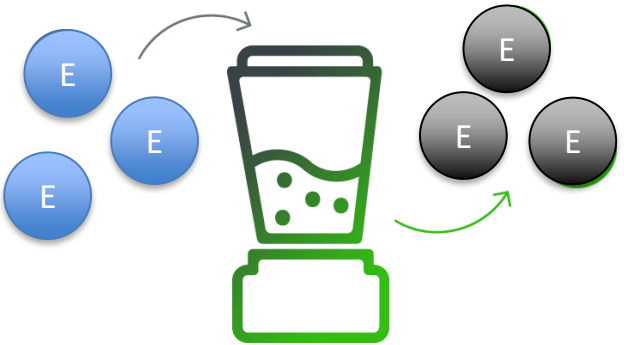
PROBLEMS/ CHALLENGES

Privacy

- Ethereum blockchain guarantees correctness and availability, not privacy for smart contracts
  - Everything on the Ethereum blockchain is public
    - Cannot execute on private data (e.g. death will remains secret until the owner dies)
- Transactions are traceable
  - Analysing transaction graph [IMC'13]

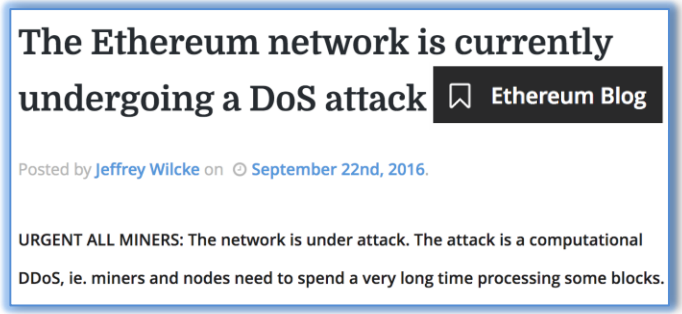
Privacy Solution

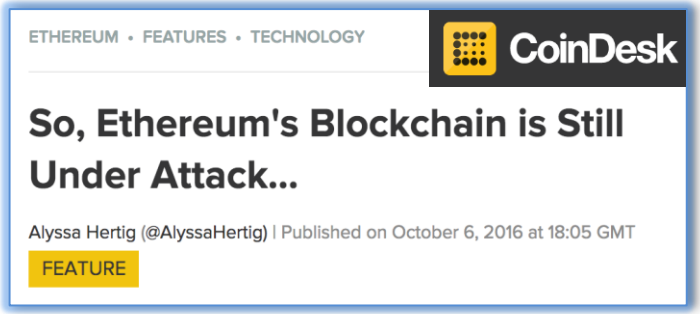
- Hawk (Kosba et al. IEEE S&P'16)
  - Privacy-Preserving Smart Contracts
  - Execute confidential, fair, multiparty protocols
- ZeroCash over Ethereum, Ring signatures on Ethereum
  - Mixing coins with others



Scalability

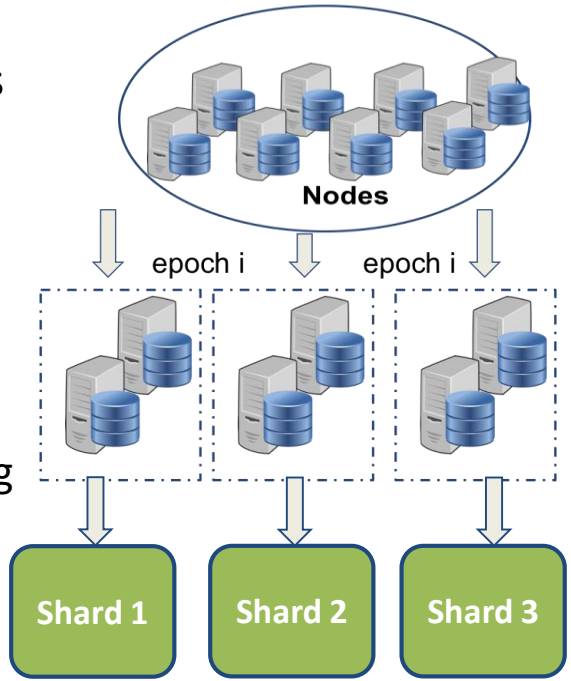
- Resources on blockchain are expensive
  - Full nodes perform the same on-chain computations
  - Full nodes store the same data
- Gas-limit is relatively small
  - Can't run an OS on blockchain
  - Can't increase gas-limit: DoS vector





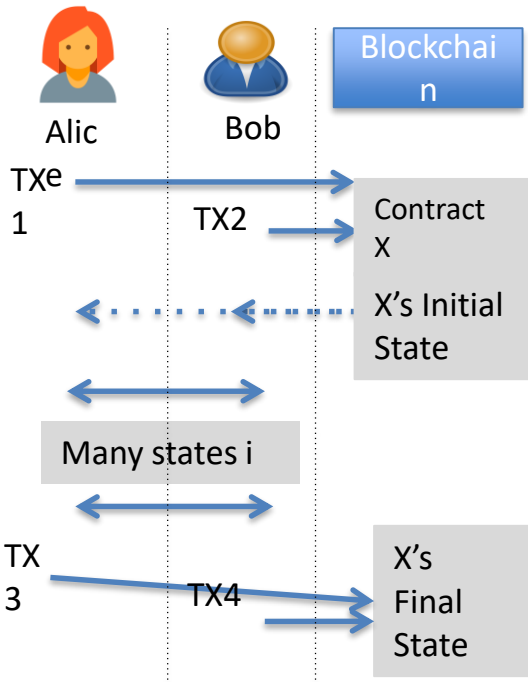
Scalability Solution 1: Sharding

- Divide the network into sub-networks
  - each stores and manages a fraction of the blockchain (a shard)
  - Allow scaling up as the network grows
- There is a catch
  - May affect usability or performance
  - May not be compatible with all existing applications



Scalability Solution 2: State Channel

- Similar to payment channel (e.g. lightning network) but for states
  - Scaling by using off-chain transactions
  - Can update the state multiple times
  - Only settlement transactions are on-chain
- Challenges
  - Cannot create state channel for all applications
  - Still early research, more work needed



Scalability Solutions: Other approaches

- Storage rental
  - Problem: data fee is charged once
  - Idea: Charge more fees if store data longer
    - Similar to resource tax
    - Incentivize users to remove unnecessary data
- Hardware-rooted trust
  - Using SGX to build state channel?

  (Inspired by teechan protocol)

Security Flaws

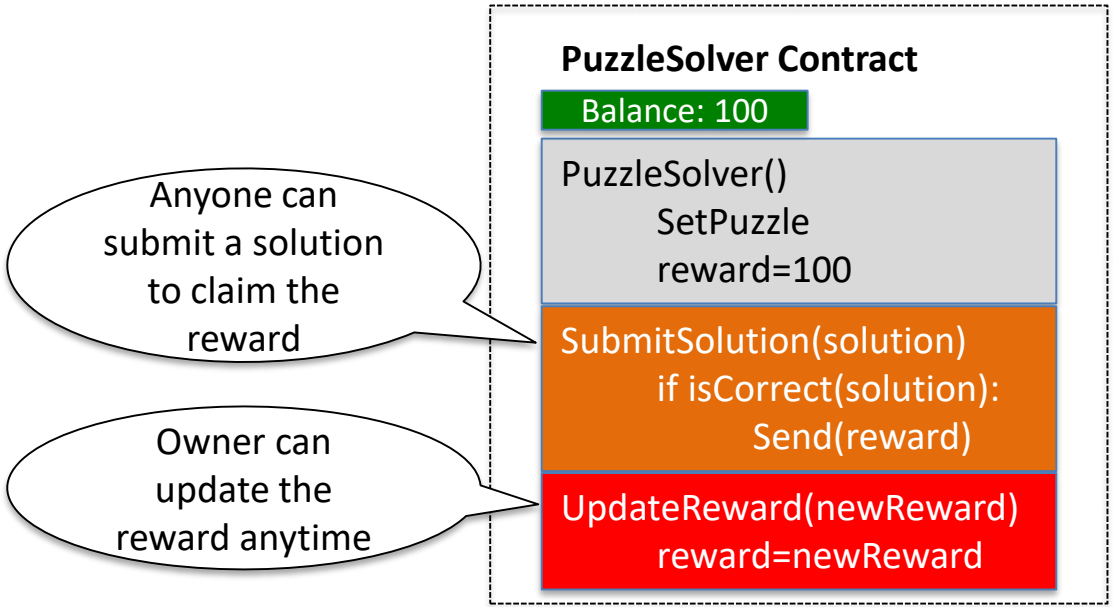
- Due to abstraction of semantic
  - Transaction ordering dependence
  - Reentrancy bug
    - Which exploited the DAO
- Obscure VM rules
  - Maximum stack depth is 1024: not many devs know
  - Inconsistent Exception Handling in EVM

The DAO Attacked: Code Issue Leads to $60 Million Ether Theft

Michael del Castillo (@DelRayMan) | Published on June 17, 2016 at 14:00 GMT

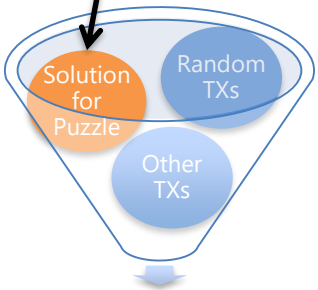
NEWS

Example 1: Transaction Ordering Dependence



Scenario 1: SubmitSolution is trigerred

+100



Miners

Block

Random TXs

SubmitSolution

Other TXs

PuzzleSolver Contract

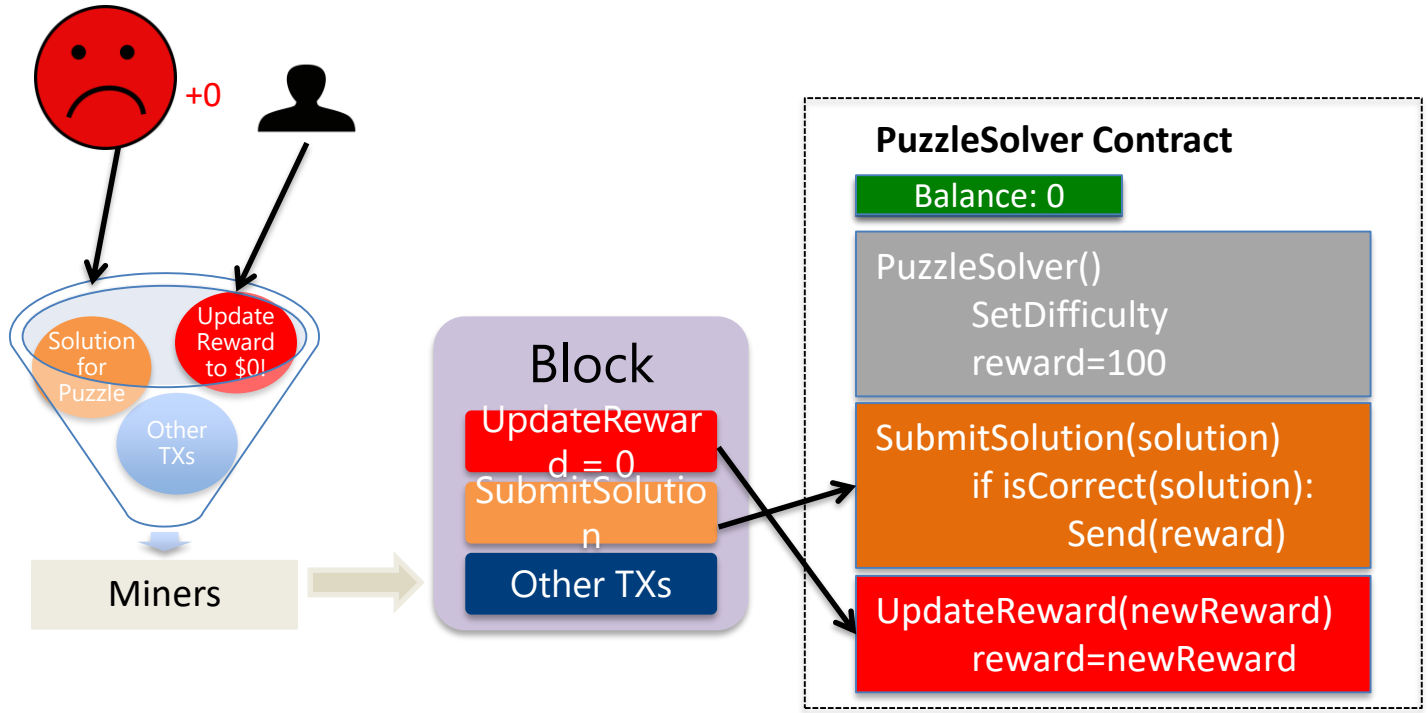
Balance: 0

```
PuzzleSolver()
        SetDifficulty
        reward=100
```

```
SubmitSolution(solution)
        if isCorrect(solution):
                Send(reward)
```

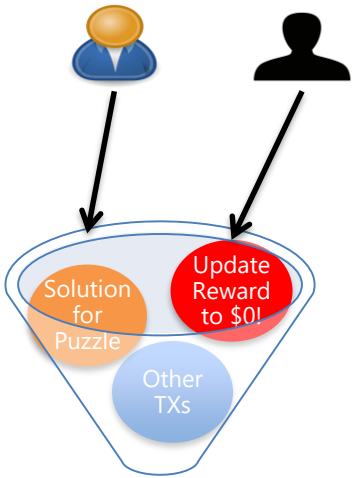
```
UpdateReward(newReward)
        reward=newReward
```

Scenario 2: Both SubmitSolution and UpdateReward are triggered



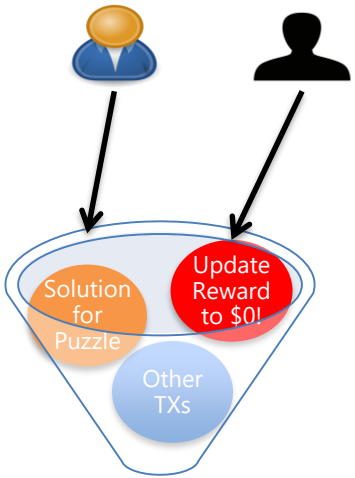
Transaction Ordering Dependence

- Observed state != execution state
  - Transactions do not have atomicity property
- Can be coincidence
  - Two transactions happen at the same time



Transaction Ordering Dependence

- Observed state != execution state
  - Transactions do not have atomicity property
- Can be coincidence
  - Two transactions happen at the same time
- Can be a malicious intention
  - Saw the targeted TX from the victim
  - Submit the second TX to update the reward
  - Both TXs enter the race

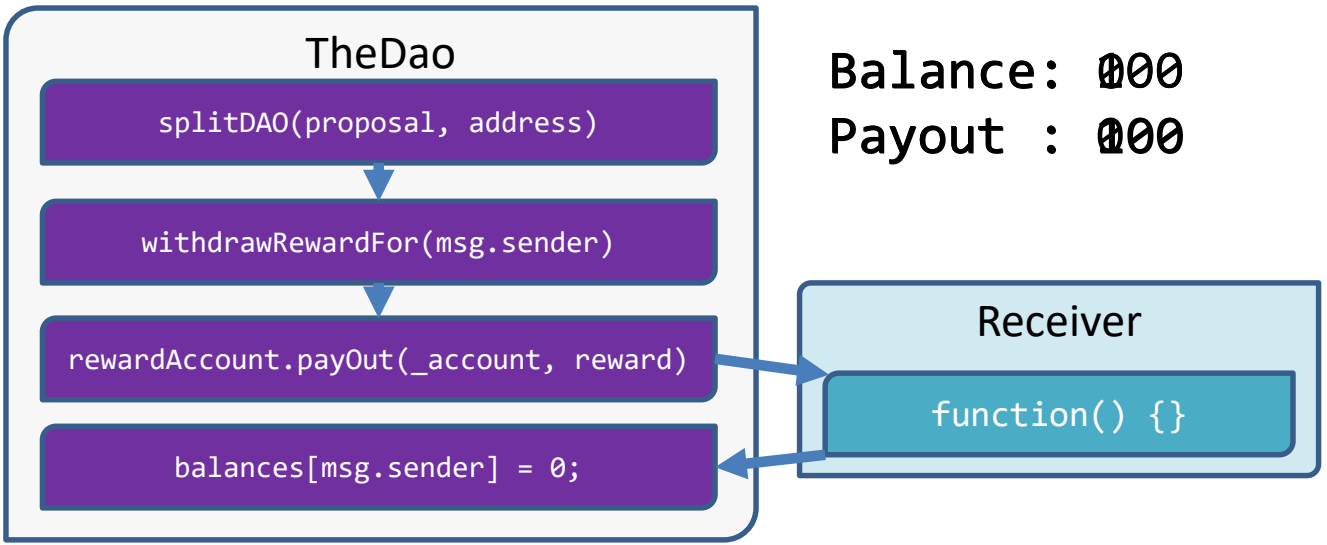


Example 2: Reentrancy Bug --- TheDAO Bug

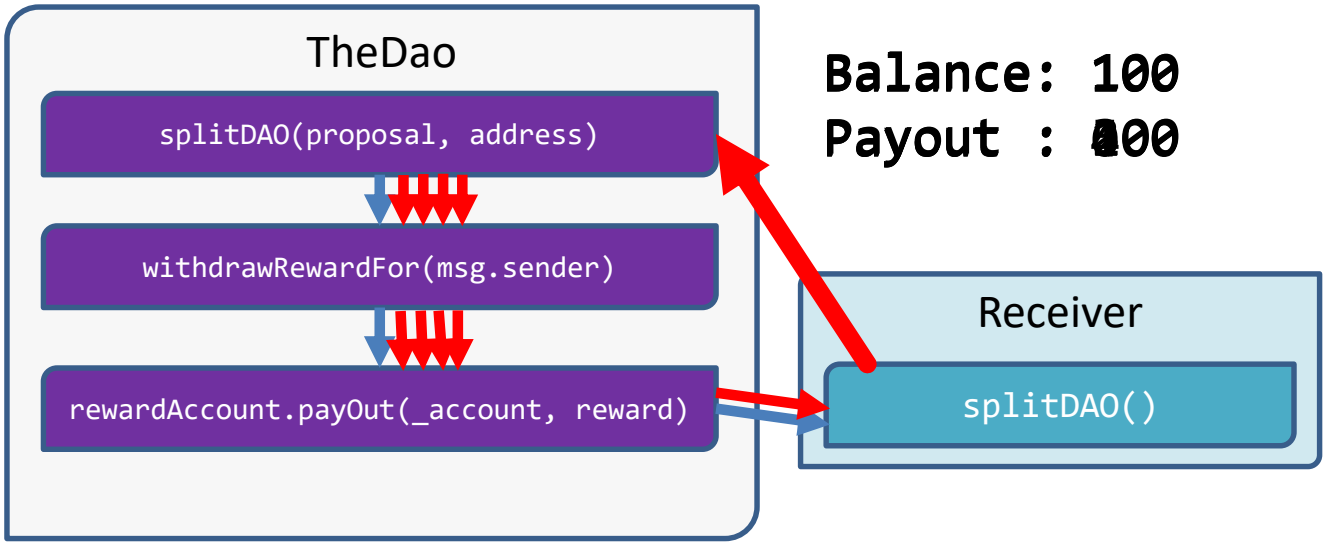
- Reentrancy vulnerability
  - Most expensive vulnerability to date
- Call before balance update

```
...
// Burn DAO Tokens
if (balances[msg.sender] == 0)
      throw;
withdrawRewardFor(msg.sender);
totalSupply -= balances[msg.sender];
balances[msg.sender] = 0;
paidOut[msg.sender] = 0;
return true;
```

TheDAO Bug: Honest Secenario



TheDAO Bug: Attack Scenario



Solutions to Resolve Security Flaws

- Create developer tools
  - Smart contract analyser based on symbolic exec: Oyente
  - Testing and deployment framework: truffle
  - Formal verification for smart contracts: eth-isabelle, why3
- Design better semantic [CCS'16]
- Educate users
- Idea
  - Create security certificates for smart contracts?

Closing thought

Ethereum and Smart contract are awesome, build your own Dapp today!

- Pay more attention to security