

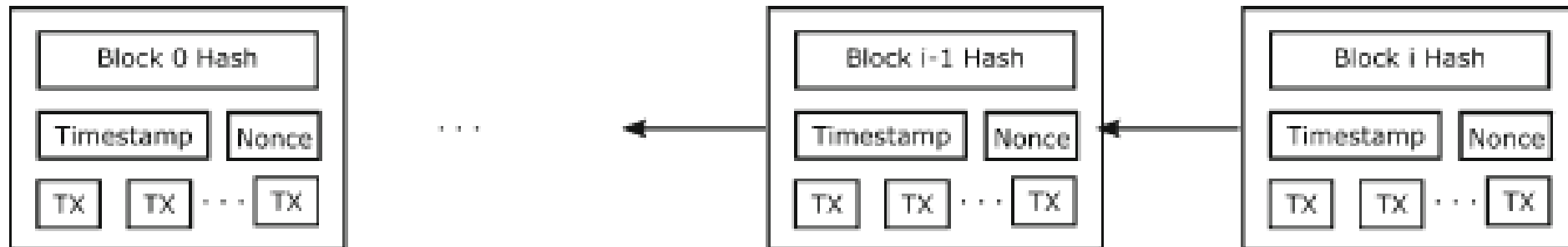
A review of quantum and hybrid quantum classical blockchain protocols

First an introduction to Blockchain

- “Blockchains are tamper evident and tamper resistant digital ledgers implemented in a distributed fashion (i.e., without a central repository) and usually without a central authority (i.e., a bank, company, or government)”[1]

The ledger

- Maintains the ownership and status of all existing coins.
- Is made up of a chain of blocks.



The ledger

- When a transaction occurs, the current ledger state is mutated by a function that takes the original state S_0 and the transaction TX , and outputs the next state S_1 or an error.

$$S_1 \text{ or } E \leftarrow \text{Apply}(S_0, TX)$$

The ledger

- In Bitcoin, a ledger's state is composed of all unspent transaction outputs (U TX O)
- Each coin has a 20-byte cryptographic public key which contains information about its owner and its denomination.
- A transaction requires references to each U TX O involved and the cryptographic signatures produced by the U TX O owners' private keys.

The consensus system

- The goal of the consensus system is to ensure that **everyone agrees** on the **validity** of the transactions that have led to the ledger's state and their order.
- There are several consensus systems that are in use today, including:
 - proof-of-work
 - proof-of-stake
 - proof-of-burn

Proof of work

- The system asks the users to do a computer resources demanding task and once they have done it the system rewards the users by giving them access or begin a communication.
- Bitcoin's consensus system is based on proof of work, and it requires that users attempt to publish their transactions constantly.
- For a block to be accepted, it's proof of work must be valid.

Proof of work

- For a block to be valid its SHA256 hash must be less than a target number specified by the system, this number is automatically varied to keep a constant rate of block generation.

```
"Hello, world!0" => 1312af178c253f84028d480a6adc1e25e81caa44c749ec81976192e2ec934c64 = 2^252.253458683
"Hello, world!1" => e9afc424b79e4f6ab42d99c81156d3a17228d6e1eef4139be78e948a9332a7d8 = 2^255.868431117
"Hello, world!2" => ae37343a357a8297591625e7134cbea22f5928be8ca2a32aa475cf05fd4266b7 = 2^255.444730341
...
"Hello, world!4248" => 6e110d98b388e77e9c6f042ac6b497cec46660deef75a55ebc7cfdcf65cc0b965 = 2^254.782233115
"Hello, world!4249" => c004190b822f1669cac8dc37e761cb73652e7832fb814565702245cf26ebb9e6 = 2^255.585082774
"Hello, world!4250" => 0000c3af42fc31103f1fdc0151fa747ff87349a4714df7cc52ea464e12dcd4e9 = 2^239.61238653
```


Proof of work - The 51% attack

- Each block needs to point to the last accepted block in the ledger.
- If an attacker wants to get coins from other owner, he would make a fake transaction block while another transaction is being made over valid coins in the ledger.
- The attacker would need to make a valid proof of work for that block, and while he is doing that other miners would publish new valid blocks, thus making the chain longer and making transactions with shorter chains invalid.

Proof of stake

- The mining power available to a miner is proportional to the amount of coins they have, their stake in the cryptocurrency system.
- The attackers would need to have the 51% of the coins in the system but by attacking it they are risking their wealth.
- This scheme incentivizes the competitions among the miners, and like evolutionary systems it rewards the “fittest” competitor

Smart contracts

- Instead of storing coins in the ledger, there are accounts stored; in Ethereum the blocks have a public-key address, nonce, balance, contract code and storage.
- There are two types of accounts, the externally owned accounts which are owned by the users and are the interface for interaction between the users and the system. And the contract accounts which are used to perform transactions.

Quantum money

Public key quantum money

- Also known as *quantum money with a classical secret*.
- Takes advantage over quantum superposition and the no-cloning theorem.
- There are multiple algorithms, but they share the basic principle, using a classical secret usually one or more arrays of length l create a quantum entangled state $|\$ \rangle$, which is shared as the public key or the quantum money.

Public key quantum money

1. Generate two random bit strings M and N of length l .
2. Prepare a quantum state $|\$ \rangle = |0 \rangle^{\otimes l}$.
3. For each bit $i < l$:
 - If $M_i = 0$ and $N_i = 0$, do nothing to the i th qubit,
 - If $M_i = 0$ and $N_i = 1$, rotate the i th qubit state to $|1 \rangle$,
 - If $M_i = 1$ and $N_i = 0$, rotate the i th qubit state to $|+ \rangle$,
 - If $M_i = 1$ and $N_i = 1$, rotate the i th qubit state to $|- \rangle$.

Public key quantum money

1. Choose a string of n uniformly random angles θ_i between 0 and 2π .
2. Using these angles, generate the state $|\varphi\rangle = \bigotimes_i |\theta_i\rangle$

$$\text{Where } |\theta_i\rangle = \cos \theta_i |0\rangle + \sin \theta_i |1\rangle$$

3. Then choose a set of n -local projectors which are all orthogonal to $|\varphi\rangle$.
4. The quantum money is the state $|\varphi\rangle$ and a classical description of the projectors.
5. Anyone can verify the money by measuring the projectors

Binding commitment

- Each transaction need to have a reference to all the U TX O and the cryptographic signatures produced by the U TX O owner's private keys.
- Using the coin owner's private key a digital signature is created and shared so that other parties can verify that the transaction was authorized by the owner of the private key and was not modified since.

Binding commitment

- Two phases:
 1. The commitment phase where, one party sends the other party some information c related to a message m .
 2. The open phase where, the sender transmits m to the receiver and proves to the receiver that m corresponds to c by providing a signature that “opens c to m .”

No algorithm A can output a commitment c and two signatures s, s' that open c to two different messages m and m' .

Binding commitment

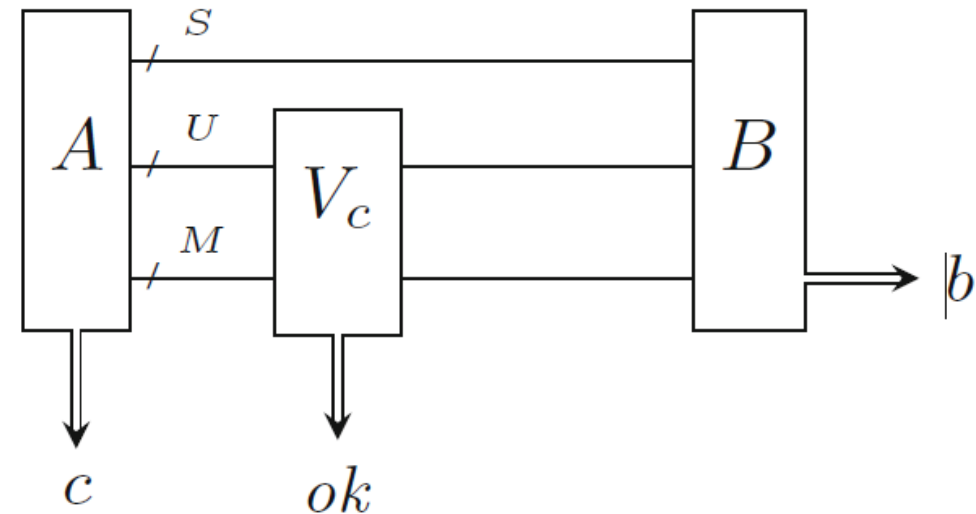
- Ambainis, Rosmanis and Unruh showed that for this binding a quantum polynomial-time algorithm A employed by an adversary could open c to any message that the adversary wished.
- Together they proposed a different type of binding that was useful in the quantum case, using two quantum games in the form of circuits.

Binding commitment

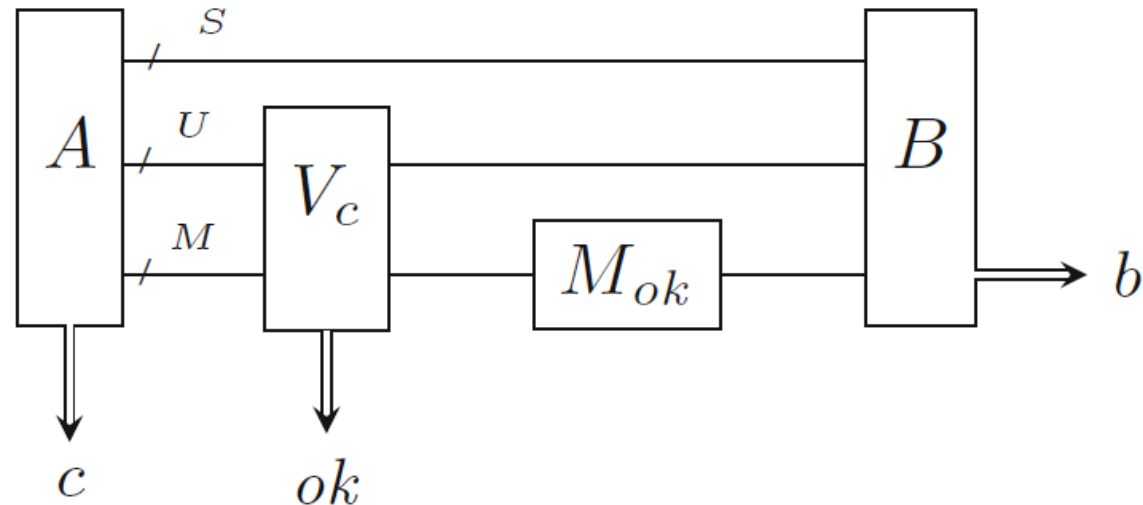
- Before defining the games is necessary to consider a perfectly-binding commitment.
- When an adversary A outputs a commitment c , there is only one possible message m_c that A can open c to.
- If the adversary A outputs a superposition of messages that he can open c to, that superposition will necessarily be in the state $|m_c\rangle$.
- When an adversary outputs a superposition of messages that he can open the commitment c to, that superposition will necessarily be a single computational basis vector

Binding commitment

- Here the adversary A outputs a commitment c (classical message).
- Additionally he outputs three quantum registers S , U , M .
- S contains his state.
- M is supposed to contain a superposition of messages.
- U a superposition of corresponding opening information.
- V_c measures whether U, M is a superposition of states $|u, m\rangle$ such that u is valid opening information for message m and commitment c .



Binding commitment



- A commitment is perfectly-binding iff for all adversaries A , the state of M after measuring $ok = 1$ is a computational basis vector A ,
- A commitment is perfectly-binding iff, for all computationally unlimited adversaries A, B , $\Pr[b = 1]$ is equal in both games where b is the output of B .

Collapsing hash functions

- H is a collapsing hash function iff no quantum polynomial-time algorithm B can distinguish between Game1 and Game2. An adversary is valid if A outputs a classical value c and a register M where $H(m) = c$.
- Mathematically, a function $H: X \rightarrow Y$ is collapsing if

$$cAdv[H](q) := \sup_{SMCU} \delta_q(M, \overline{M} | \overline{CU}) \leq \epsilon(q)$$

Collision free quantum money

- The mint or bank cannot efficiently produce two quantum coins with the same verification circuit.
- Everyone can verify a quantum coin.

Collision free quantum money

- Start with a classical set, a set of n -bit strings and a classical function L that assign a label to each element of the set.
- Produce an equal superposition of all n -bit strings.
- Then compute L into an *ancilla* register which will be measured to obtain a label l . The state left over after the measurement will be $|\varphi_l\rangle$ which is in the form:

$$|\psi_\ell\rangle = \frac{1}{\sqrt{N_\ell}} \sum_{x \text{ s.t. } L(x)=\ell} |x\rangle$$

Collision free quantum money - verification

- The verification of this scheme of quantum money is done by using rapidly mixing Markov chains.
- It is assumed that we have a Markov matrix M which, will take any distribution over bit strings that share the same label l , and rapidly mixes to the uniform distribution over those strings excluding the bit strings that don't share the same label.

Collision free quantum money - verification

- Each update over the Markov chain must consist of a uniform random choice over N update rules, where each update rule is deterministic and invertible.

$$M = \frac{1}{N} \sum_{i=1}^N P_i.$$

- The operator M is acting over the Hilbert space, (where quantum money lives), and therefore any valid quantum money state $|\varphi_l\rangle$ is a +1 eigenstate of M

$$M^T \approx \sum_l |\psi_l\rangle \langle \psi_l|$$

Collision free quantum money - verification

- The other element that is needed is an unitary quantum operator U that works over two registers and is defined as follows.

$$U = \sum_i P_i \otimes |i\rangle\langle i|.$$

Collision free quantum money - verification


1. Given some initial quantum state on n qubits, add an *ancilla* in a uniform superposition over all i (from 1 to N).
2. Then apply the unitary U
3. Measure the projector of the ancilla onto the uniform superposition
4. Discard the ancilla.

The Kraus operator sum element corresponding to the outcome 1 is


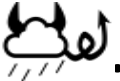
$$\begin{aligned} & \left(I \otimes \frac{1}{\sqrt{N}} \sum_{i=1}^N \langle i| \right) U \left(I \otimes \frac{1}{\sqrt{N}} \sum_{i=1}^N |i\rangle \right) \\ &= \frac{1}{N} \sum_{i=1}^N P_i \\ &= M. \end{aligned}$$

Quantum lightning



Quantum lightning

- Proposed by Mark Zhandry in 2017, it uses of non-collapsing collision-resistant hash functions.
- A quantum lightning protocol consist of two efficient quantum algorithms. A bolt generation procedure, or storm  , which takes as an input a security parameter λ , and generates a quantum state $|\text{⚡}\rangle$ on each invocation. And a **Ver** algorithm that verifies a bolt while extracts its fingerprint or serial number, this function returns this serial denoted with s or \perp when the bolt is not valid.

Quantum lightning

- **Ver** is required to: always accept bolts generated by  , does not perturb valid bolts and always outputs the same serial number on a given bolt.
- It is necessary to be computationally infeasible to produce two bolts $|\text{⚡}_0\rangle$ and $|\text{⚡}_1\rangle$ such that **Ver** accepts both and outputs identical serial numbers. This is true for even for adversarial storms .

Quantum lightning

- The system has a family F_λ of pairs ( , **Ver**) for each security parameter, there will also be a setup procedure $\text{SetupQL}(1^\lambda)$ which samples a pair ( , **Ver**) from some distribution over F_λ .

Hybrid payment system using quantum lightning

- Classical blockchain, using quantum lightning as its coins.
- The classical serial numbers and certificates of the quantum coins used as the interface between the quantum and classical elements of the system.
- The classical blockchain uses a global ledger.

```
parties = {}  
contracts = {}  
allTransactions = {}  
t = 0
```

Hybrid payment system using quantum lightning

- **Register** (id, num_coins) \rightarrow (pid): Stores the id of an user and returns their party id, used to refer to all of its information.
- **Retrieve Party** (pid) \rightarrow (id, num_coins): Retrieves a party's information
- **Pay** (pid, pid', num_coins) \rightarrow ($trid$): Used to send coins from pid to pid' , either return \perp or makes the following operation

*$*pid'.coins \leftarrow *pid'.coins + num_coins$*

*$*pid.coins \leftarrow *pid.coins - num_coins$*

$trid \leftarrow |allTransactions| + 1$

$allTransactions[trid] \leftarrow$

$(pid, pid', num_coins, time).$

Hybrid payment system using quantum lightning

- **Retrieve Transaction (trid)→(allTransactions[trid]):** Retrieve transaction details
- **Smart Contract ($pids, \{(pid, num_coins_{pid}) : pid \in pids\}, circuit, st_0$) → (cid):** Create a contract.

$cid \leftarrow |contracts| + 1$

$*cid.params \leftarrow$

$(pids, \{(pid, num_coins_{pid}) : pid \in pids\},$
 $circuit, st_0)$

$*cid.num_coins \leftarrow 0$

$contracts[cid] \leftarrow *cid.$

- **Retrieve Smart Contract (cid)→(params, coins):** Retrieve contract details

Hybrid payment system using quantum lightning

- Waits for an initialization message from each party

$$*pid.coins \leftarrow *pid.coins - num_coins_{pid}$$
$$\forall pid \in *cid.params.pids$$
$$*cid.coins \leftarrow *cid.coins + num_coins_{pid}$$
$$\forall pid \in *cid.params.pids$$
$$st \leftarrow st_0.$$

- Enters an execution phase

$$*pid.coins \leftarrow *pid.coins - num_coins$$
$$*cid.coins \leftarrow *cid.coins + num_coins$$
$$(st, result) \leftarrow circuit(pid, witness, time,$$
$$st, num_coins).$$

Hybrid payment system using quantum lightning

- **Initialize with coins (pid, cid, num_coins) → ()** : Allows a user to deposit coins into a contract.

$*pid.coins \leftarrow *pid.coins - num_coins_{pid}$

$\forall pid \in *cid.params.pids$

$*cid.coins \leftarrow *cid.coins + num_coins_{pid}$

$\forall pid \in *cid.params.pids$

$st \leftarrow st_0.$

- **Trigger (pid, cid, witness, time, st, num_coins) → (result)**: Runs the circuit associated with a contract with the given parameters.

Hybrid payment system using quantum lightning

- **Generation of valid coins**

$$|\text{⚡}\rangle \leftarrow \text{☁}$$
$$serial \leftarrow \mathbf{Ver}(|\text{⚡}\rangle).$$

- **Payment**

- P sends $|\text{⚡}\rangle$, cid , $serial$ and num_coins to P' .
- P' sends a *Retrieve Contract* message to the ledger, retrieving the contract cid .
- P' accepts the payment if $cid \in contracts$ and $\mathbf{Ver}(|\text{⚡}\rangle) = serial$.

Hybrid payment system using quantum lightning

- **Reclaim lost coins**

$$m \leftarrow A(|\text{⚡}\rangle)$$

$$|\text{⚡}\rangle \leftarrow \text{☁}$$

$$\text{serial}' \leftarrow \text{Ver}(|\text{⚡}\rangle).$$

- **Trade quantum coins for classical coins**

References

Main article: Edwards, M., Mashatan, A. & Ghose, S. A review of quantum and hybrid quantum/classical blockchain protocols. *Quantum Inf Process* 19, 184 (2020).
<https://doi.org/10.1007/s11128-020-02672-y>

1. Yaga, Dylan J., et al.: *Blockchain Technology Overview*. NIST (2018).
<https://www.nist.gov/publications/blockchain-technology-overview>.
2. Unruh, D.: Collapse-binding quantum commitments without random oracles. In: *Advances in Cryptology—ASIACRYPT 2016 Lecture Notes in Computer Science*, pp. 166–195 (2016).
https://doi.org/10.1007/978-3-662-53890-6_6
3. Lutomirski, A., et al.: Breaking and making quantum money: toward a new quantum cryptographic protocol. In: *Innovations in Computer Science—ICS 2010*, Tsinghua University, Beijing, China, January 5–7, 2010. *Proceedings*, pp. 20–31. Tsinghua University Press (2009)
4. Zhandry M. (2019) Quantum Lightning Never Strikes the Same State Twice. In: Ishai Y., Rijmen V. (eds) *Advances in Cryptology – EUROCRYPT 2019*. EUROCRYPT 2019. *Lecture Notes in Computer Science*, vol 11478. Springer, Cham. https://doi.org/10.1007/978-3-030-17659-4_14