Blockchains & Cryptocurrencies

Alternative Puzzles - II

Instructor: Abhishek Jain
Johns Hopkins University - Spring 2021

*Some slides based on NBFMG

Housekeeping

- HW1 Due Today
- Course Project Idea Due on 02/26

Last Time

- Mining Strategies (Default and Non-Default) & Incentives
- Puzzle Requirements and Alternative Puzzles (Motivation and Designs)

Today

- Alternative Puzzles (contd.)

Recall: Puzzle requirements

- Cheap to Verify
  - since other users have to verify solutions
- Adjustable difficulty
  - E.g., due on increase in hash rate or more users
- In PoW puzzles, chance of winning should be proportional to computing power (e.g., hash power in Bitcoin)
  - Large players get only proportional advantage
  - Even small players get proportional compensation

ASIC Resistant (PoW) Puzzles

ASIC resistance - Why? (1 of 2)

Goal: Ordinary people with idle laptops, PCs, or even mobile phones can mine!

Lower barrier to entry

Approach: Reduce the gap between custom hardware and general purpose equipment

ASIC resistance - Why? (2 of 2)

Goal: Prevent large manufacturers from dominating the game

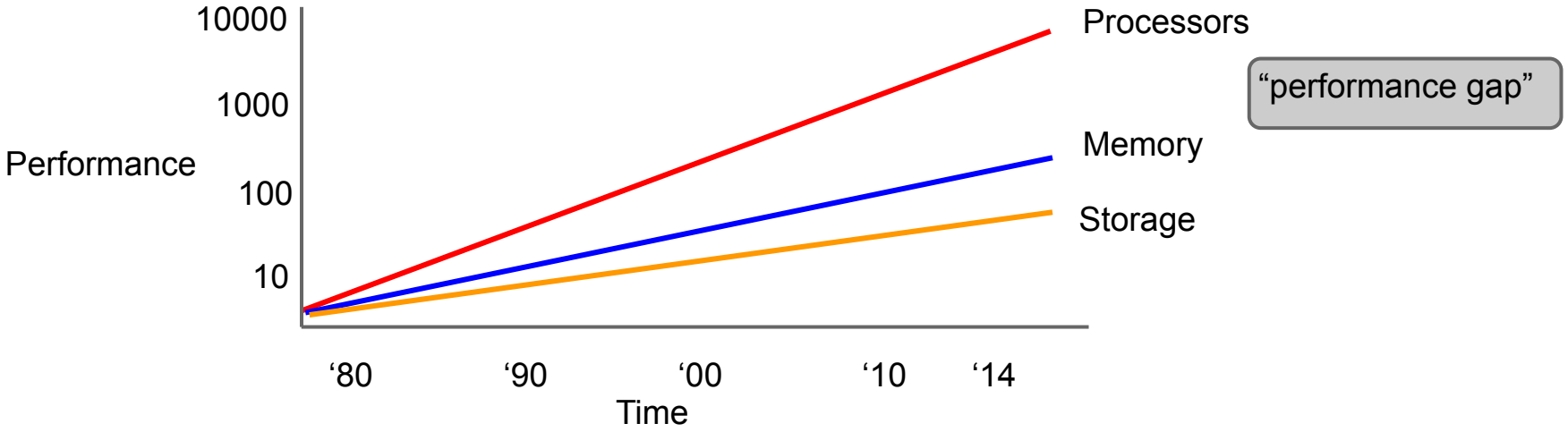
"Burn-in" advantage

In-house designs

Approach: reduce the "gap" between future hardware and the custom ASICs we already have

Memory hard puzzles

Premise: the cost and performance of memory is more stable than for processors



scrypt

Colin Percival, 2009

- Memory hard hash function
  - ***Constant time/memory tradeoff***
  - Memory consumes a large amount of on-chip area. High memory requirement => small number of hashing engines on special-purpose chips
- Widely used alternative PoW puzzle (e.g., Litecoin)
- Also used in Password-hashing

1. Fill memory with random values
2. Read from the memory in random order

scrypt - step 1 of 2 (write)

Input: **X**

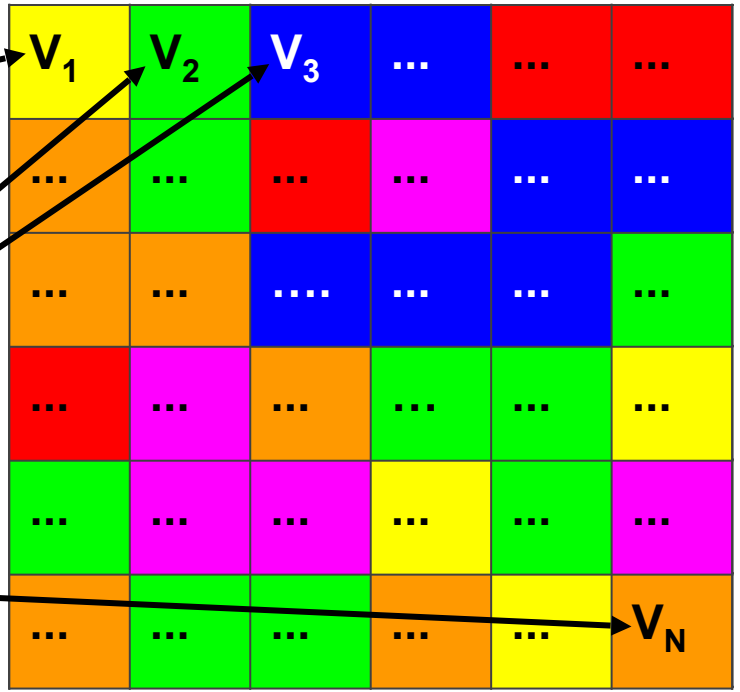
$V_1 = H(X)$

$V_2 = H(V_1) = H(H(X))$

$V_3 = H(V_2) = H^3(X)$

...

$V_N = H^N(x)$



scrypt - step 2 of 2 (read)

Input: **X**

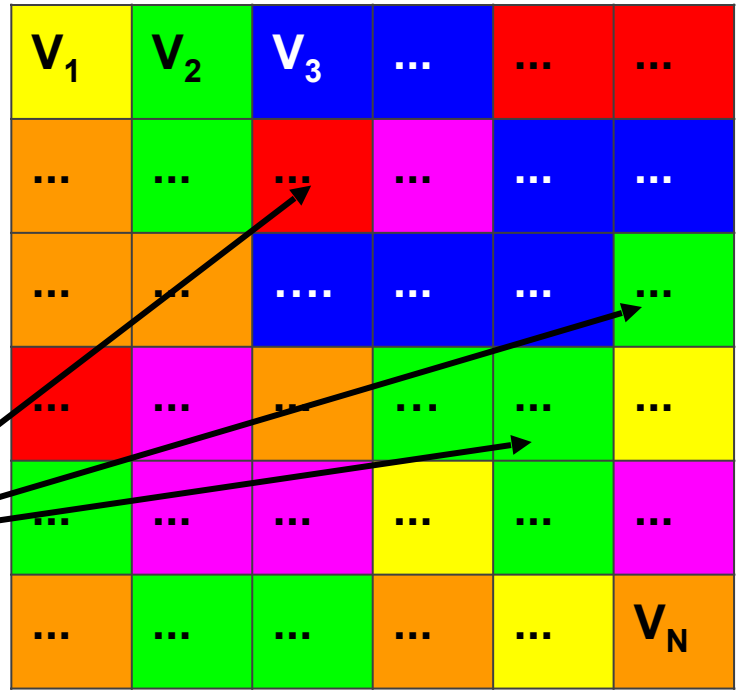
A := H^{N+1}(X)

For N iterations:

i := A mod N

A := H(A xor V_i)

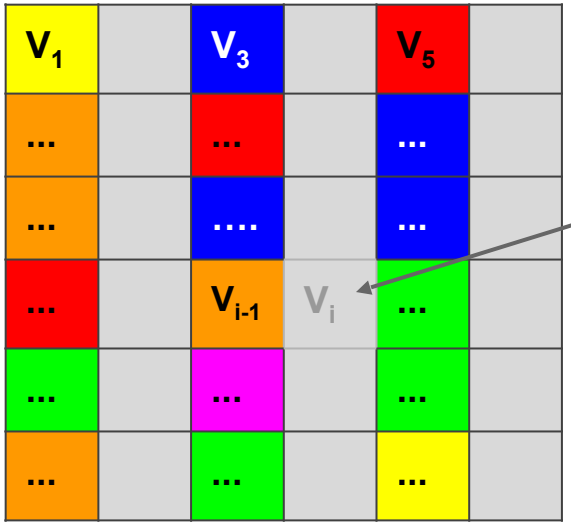
Output: A



scrypt - time/memory tradeoff

Why is this memory-hard?

Reduce memory by half, 1.5x the # steps



Need to access V_i where i is even?

Access V_{i-1}

Compute $V_i = H(V_{i-1})$

scrypt

Disadvantages: Also requires N steps, N memory to check

Is it actually ASIC resistant?

- scrypt ASICs ***are*** already available
- Exploit time-memory trade-offs, lower values of N, etc.

Academic research

- Many subsequent candidates: Argon2i (winner of PW-hashing contest), Ballon-Hashing, etc.
- Proofs of memory hardness in various models using graph pebbling techniques (see, e.g., Alwen-Serbeninko'15 and many subsequent works)

Cuckoo hash cycles

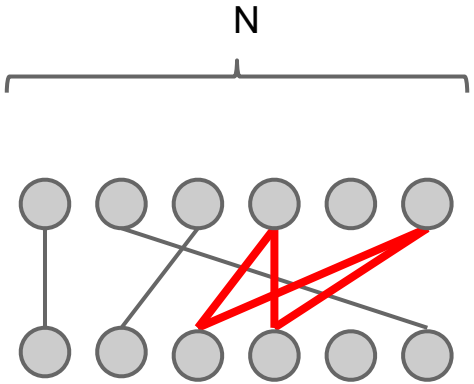
John Tromp, 2014

(Conjectured) Memory hard puzzle that's cheap to verify

Input: X

For i = 1 to E:

```
a := H0(X + i)
b := N + H1(X + i)
edge(a mod N, b mod N)
```



Is there a cycle of size K? If so, Output: X, K edges

Even more approaches

- More complicated hash functions
  - X11: 11 different hash functions combined (subsequent iterations: X13, X14, X15, X17)

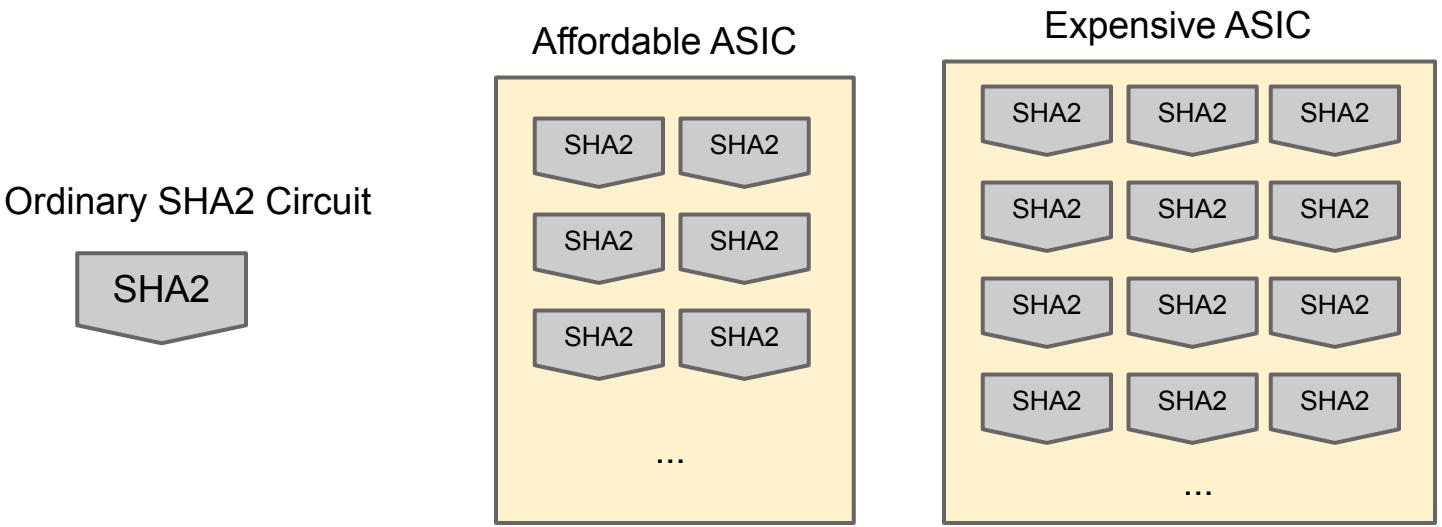
- Moving target

  Change the puzzle periodically

Counter argument: SHA2 is fine

Bitcoin Mining ASICs aren't changing much

Big ASICs only marginally more performant than small ones



Proof-of-useful-work

Recovering wasted work

Recall: power consumed by Bitcoin network in 2019
~ power consumed by Switzerland :(

Natural question:

Can we recycle this and do something useful?

Candidates - needle in a haystack

- Potential choices:
  - Protein folding (find a low energy configuration)
  - Search for aliens (find an anomalous region of a signal)

Candidates - needle in a haystack

- Potential choices:
  - Protein folding (find a low energy configuration)
  - Search for aliens (find an anomalous region of a signal)
- Challenges:
  - Randomly chosen instances must be hard
    - Who chooses the problem?
    - Verification must also be efficient

Primecoin

Sunny King, 2013

Puzzle based on finding large prime numbers

Puzzle parameters: **m, n, k**

Cunningham chain:

$p_1, p_2, \ldots p_k$ where $p_i = 2p_{i-1} + 1$

Each p_i is a large (probable) prime

p_1 is **n**-bit long and shares same leading **m** bits as

H(prev || mrkl_root || nonce)

Primecoin

- Many of the largest known Cunningham chains have come from Primecoin miners
- Hard problem? Studied by others (e.g., PrimeGrid)
- Usefulness? Some applications to crypto (e.g., Young-Yung'98)

Short Paper: The Proof is in the Pudding

Proofs of Work for Solving Discrete Logarithms

Marcella Hastings[1], Nadia Heninger[2], and Eric Wustrow[3]

[1] University of Pennsylvania
[2] University of California, San Diego
[3] University of Colorado Boulder

Abstract. We propose a proof of work protocol that computes the discrete logarithm of an element in a cyclic group. Individual provers generating proofs of work perform a distributed version of the Pollard rho algorithm. Such a protocol could capture the computational power expended to construct proof-of-work-based blockchains for a more useful purpose, as well as incentivize advances in hardware, software, or algorithms for an important cryptographic problem. We describe our proposed construction and elaborate on challenges and potential trade-offs that arise in designing a practical proof of work.

Keywords: Proofs of work, discrete log, Pollard rho

Recovering wasted hardware

Estimate: more than $100M spent on customized Bitcoin mining hardware (old data from 2014)

This hardware investment is otherwise useless

Idea: a puzzle where hardware investment is useful, even if the work is wasted?

Permacoin - Mining with storage

Miller et al., 2014

Bitcoin

Permacoin

Side effect:

Massively distributed, replicated storage system

Permacoin

Assume we have a large file **F** to store

For simplicity: **F** is chosen globally, at the beginning, by a trusted dealer

Each user stores a random subset of the file

Storage-based puzzle

1. Build a Merkle tree, where each leaf is a segment of the file

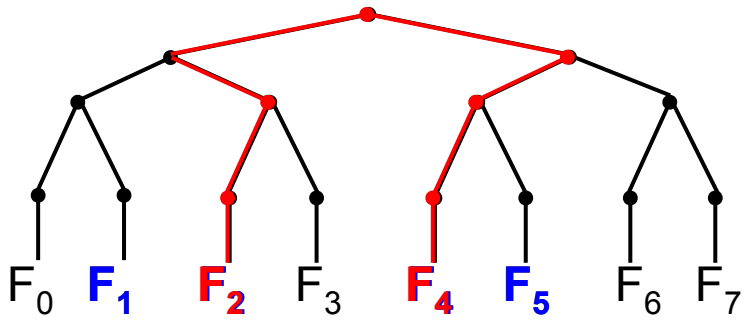
2. Generate a public signing key pk, which determines a random subset of file segments to store

3. Each mining attempt:

a) Select a random nonce
b) h1 := H(prev || mrkl_root || PK || nonce)
c) h1 selects k segments from subset
d) h2 := H(prev || mrkl_root || PK || nonce || F)
e) Winner if h2 < TARGET

F_1 F_2 F_4 F_5

F_2 F_4



Summary

- Useful proof-of-work is a natural goal

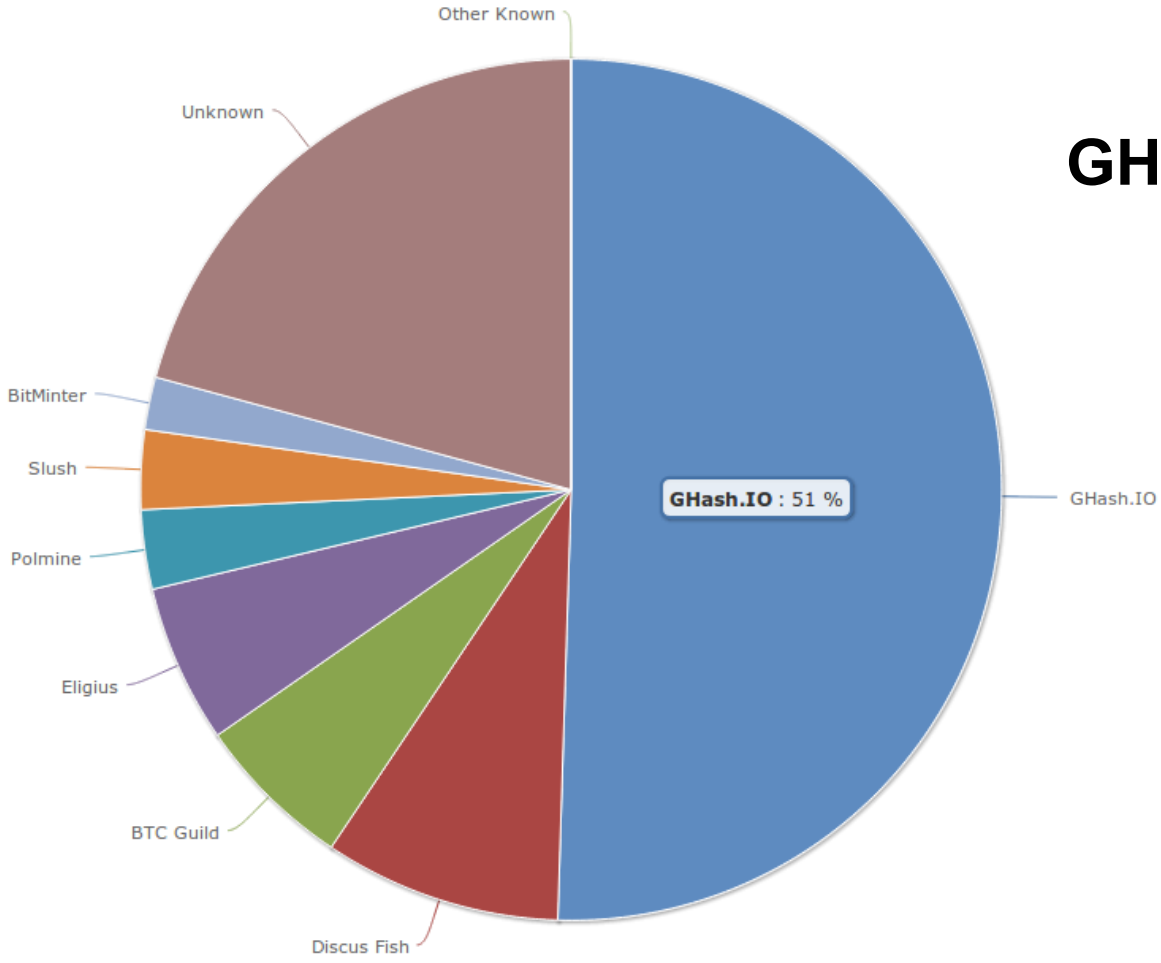
  (while maintaining security requirements)
- The benefit must be a pure public good
- Viable approaches include storage, prime-finding, others may be possible
- Realized benefit so far has been limited

Non-outsourceable Puzzles

Large mining pools are a threat

- Bitcoin's core value is decentralization
- If power is consolidated in a few large pools, the operators are targets for coercion/hacking
- Position: large pools should be discouraged

  Analogy to voting: It's illegal (in US) to sell your vote



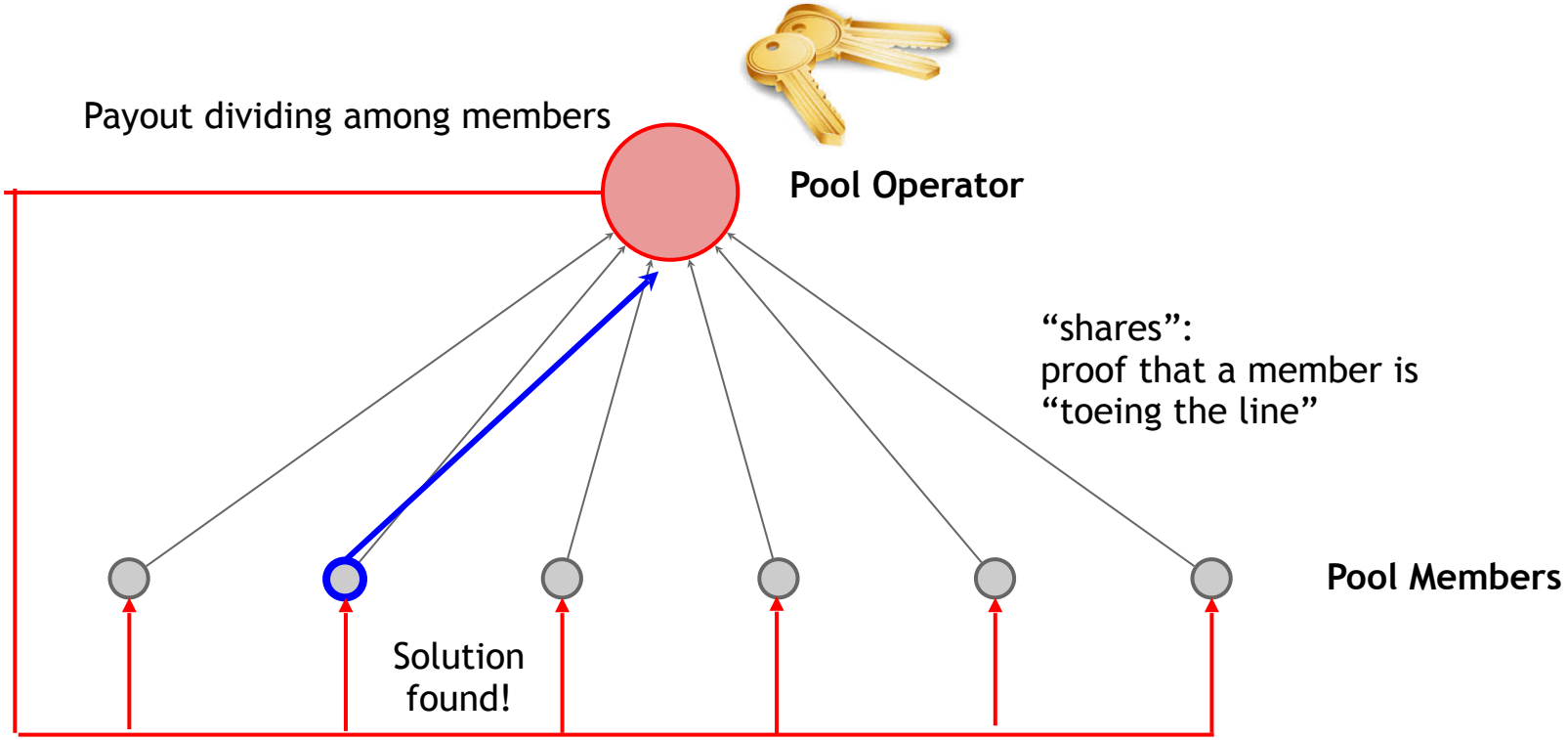
June 12, 2014
GHash.IO large mining pool crisis

Observation:

Pool participants don't trust each other

Pools only work because the "shares" protocol lets members ***prove*** cooperation

Standard Bitcoin mining pool



The Vigilante Attack

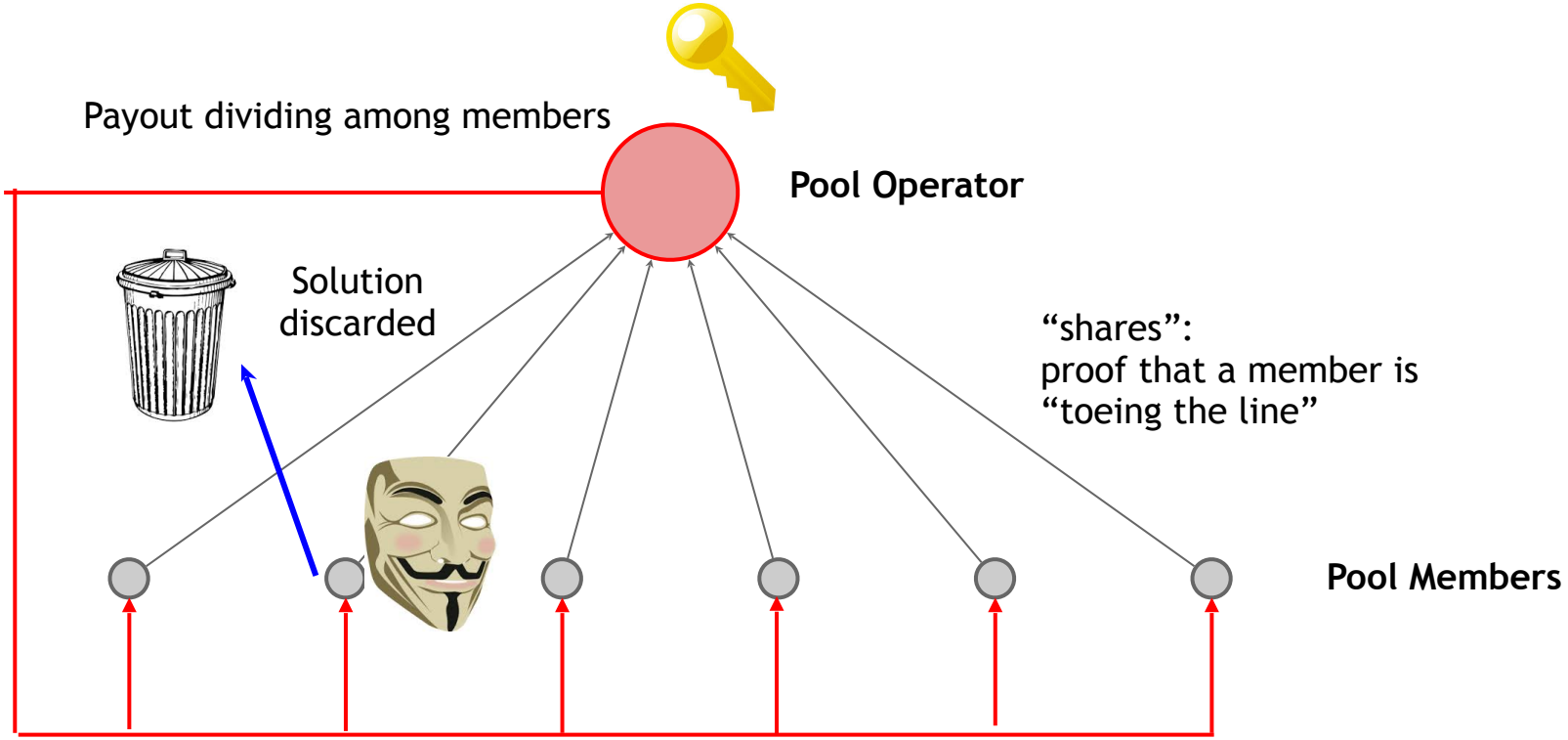
Suppose a Vigilante is angry with a large pool

He submits "shares" like normal….

… but if he finds a real solution, discards it

Pool output is reduced, Vigilante loses a little

The Vigilante Attack



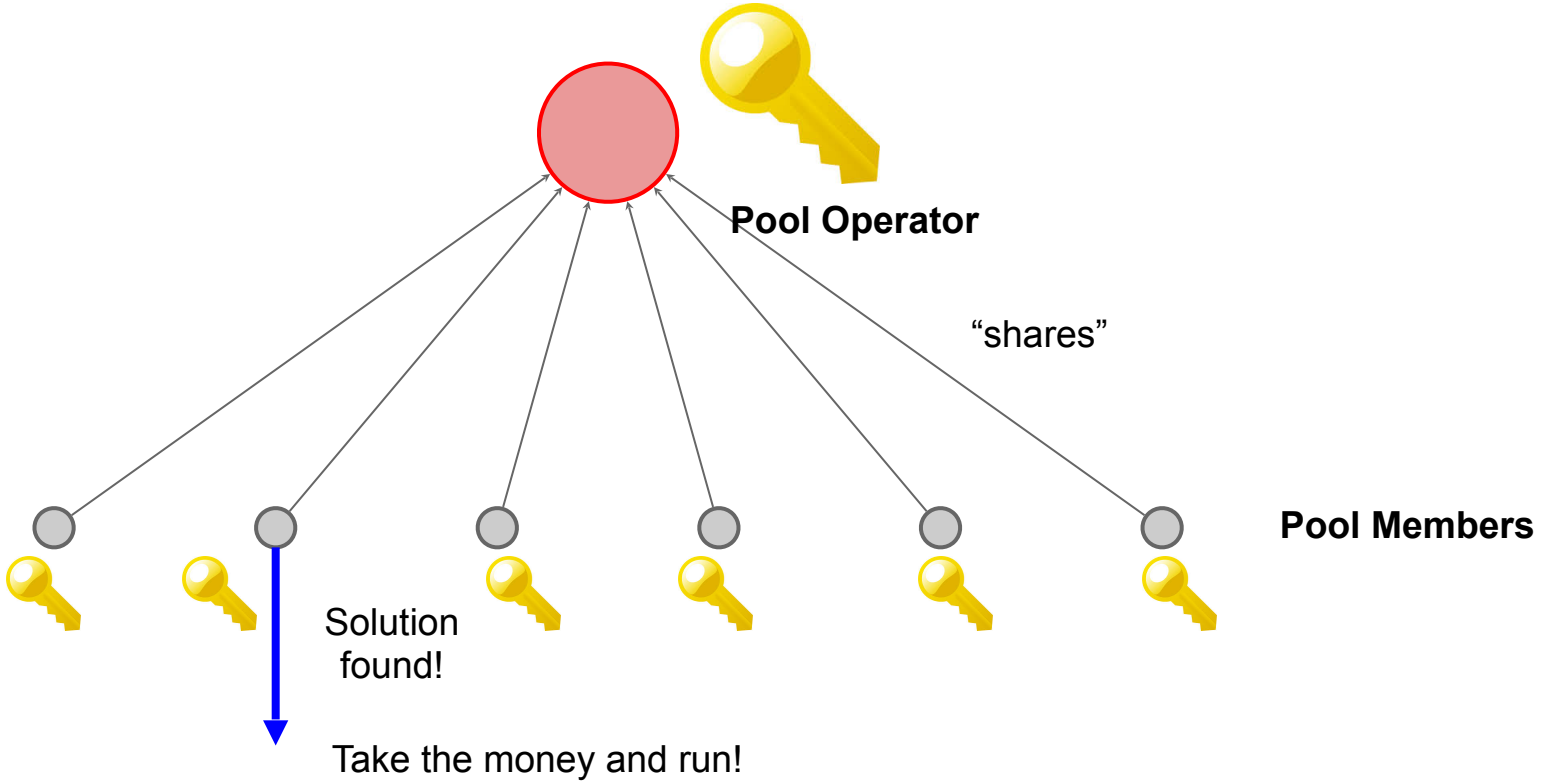
Encouraging the Vigilante

Whoever ***FINDS*** a solution spends the reward

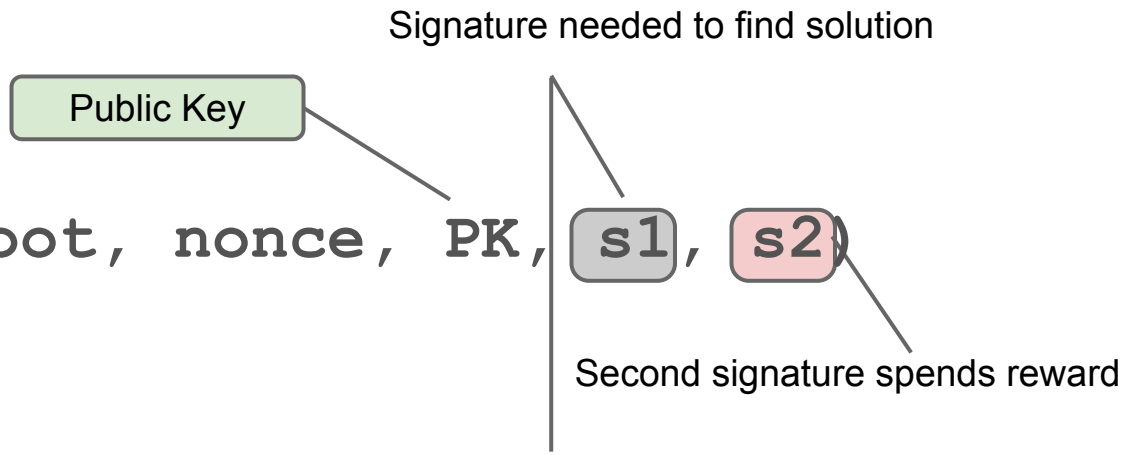
Approach:

- searching for a solution requires ***SIGNING,*** not just hashing. (Knowledge of a private key)
- Private key can be used to spend the reward

Encouraging the Vigilante



Nonoutsourceable puzzle



Solution:

```
(prev, mrkl_root, nonce, PK, s1, s2)
```

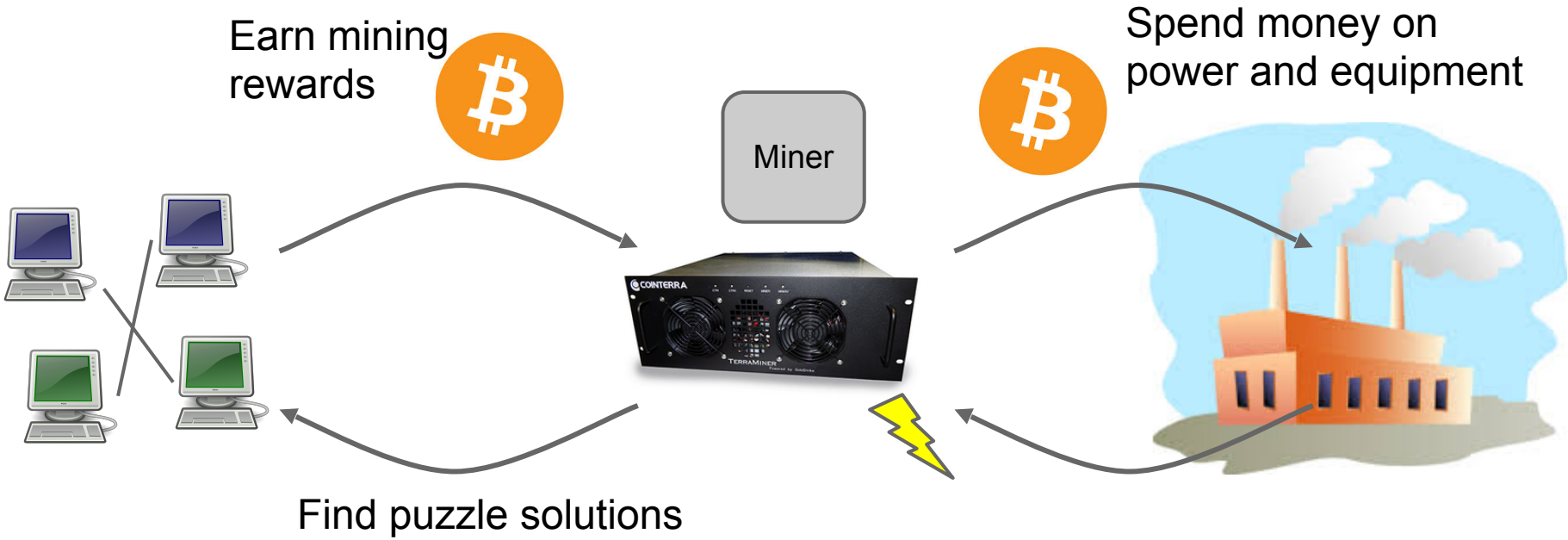
such that:

```
H(prev || PK || nonce || s1) < TARGET
VerifySig(PK, s1, prev || nonce)
VerifySig(PK, s2, prev || mrkl_root)
```

“Virtual Mining”

Bitcoin Mining has an unnecessary step

Proof-of-Work Mining:



Proof of Stake

- Creator of next block chosen at random based on amount of current "stake" in the system
- Assuming all the money owned/used by miners is in the system, this mechanism cuts the middle man (equipment manufacturer)

Potential benefits

- Lower overall costs
  - No harm to the environment
  - Savings distributed to all coin holders
- Stakeholder incentives - good stewards?
- No ASIC advantage
- 51% attack might be harder (this is debatable)

Examples of PoS based Cryptocurrencies

- Cardano
- Algorand
- Ethereum 2 (hopefully!)
- Nxt
- Neucoin
- …

Examples of secure PoS systems

- Algorand [Full version: Chen-Micali'17]
- Cardano/Ourboros [Kiayias-Russel-David-Oliynykov'17]
- Snow white [Daian-Pass-Shi'17]

Proofs of Space

- Require non-trivial disk space to solve a puzzle
  [Dziembowski et al. CRYPTO'15, Ateniese et al. SCN'14]
- More environmental-friendly than PoW
- Used in Chia, and FileCoin (combination of proof of storage/retrievability and proof of space)

Questions with Virtual Mining

Is there any security that can only be gained by consuming "real" resources?

- If so, then "waste" is the cost of security
- If not, then PoW mining may go extinct

Conclusion

- Many possible design goals for puzzles
  - Prevent ASIC miners from dominating
  - Prevent large pools from dominating
  - Intrinsic usefulness
  - Eliminate the need for mining hardware at all
- Further research required to understand the best tradeoffs
- Many competing systems already co-exist